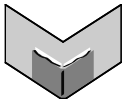
**The Journal of Machine Learning Research**
Volume 14
Print-Archive Edition

Pages 1285–2518



**Microtome Publishing**
Brookline, Massachusetts
www.mtome.com

**The Journal of Machine Learning Research**
Volume 14
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 2013.

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/.

Collection copyright © 2013 The Journal of Machine Learning Research, Inc. and Microtome Publishing. Copyright of individual articles remains with their respective authors.

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

# JMLR Editorial Board

Editor-in-Chief
**Bernhard Schölkopf**, MPI for Intelligent Systems, Germany

Editor-in-Chief
**Kevin Murphy**, Google Research, USA

Managing Editor
**Aron Culotta**, Southeastern Louisiana University, USA

Production Editor
**Rich Maclin**, University of Minnesota, Duluth, USA

JMLR Web Master
**Chiyuan Zhang**, Massachusetts Institute of Technology, USA

JMLR Action Editors
**Peter Auer**, University of Leoben, Austria **Francis Bach**, INRIA, France **David Barber**, University College London, UK **Mikhail Belkin**, Ohio State University, USA **Yoshua Bengio**, Université de Montréal, Canada **Samy Bengio**, Google Research, USA **Jeff Bilmes**, University of Washington, USA **David Blei**, Princeton University, USA **Karsten Borgwardt**, MPI For Intelligent Systems, Germany **Léon Bottou**, Microsoft Research, USA **Lawrence Carin**, Duke University, USA **Francois Caron**, University of Bordeaux, France **David Maxwell Chickering**, Microsoft Research, USA **Andreas Christman**, University of Bayreuth, Germany **Alexander Clark**, King's College London, UK **William W. Cohen**, Carnegie-Mellon University, USA **Corinna Cortes**, Google Research, USA **Koby Crammer**, Technion, Israel **Sanjoy Dasgupta**, University of California, San Diego, USA **Peter Dayan**, University College, London, UK **Rina Dechter**, University of California, Irvine, USA **Inderjit S. Dhillon**, University of Texas, Austin, USA **David Dunson**, Duke University, USA **Charles Elkan**, University of California at San Diego, USA **Yoav Freund**, University of California at San Diego, USA **Kenji Fukumizu**, The Institute of Statistical Mathematics, Japan **Sara van de Geer**, ETH Zurich, Switzerland **Amir Globerson**, The Hebrew University of Jerusalem, Israel **Moises Goldszmidt**, Microsoft Research, USA **Russ Greiner**, University of Alberta, Canada **Arthur Gretton**, University College London, UK **Maya Gupta**, Google Research, USA **Isabelle Guyon**, ClopiNet, USA **Matthias Hein**, Saarland University, Germany **Aapo Hyvärinen**, University of Helsinki, Finland **Alex Ihler**, University of California, Irvine, USA **Tommi Jaakkola**, Massachusetts Institute of Technology, USA **Tony Jebara**, Columbia University, USA **Sathiya Keerthi**, Microsoft Research, USA **John Lafferty**, University of Chicago, USA **Christoph Lampert**, Institute of Science and Technology, Austria **Gert Lanckriet**, University of California, San Diego, USA **John Langford**, Microsoft Research, USA **Pavel Laskov**, University of Tübingen, Germany **Neil Lawrence**, University of Manchester, UK **Guy Lebanon**, Amazon, USA **Daniel Lee**, University of Pennsylvania, USA **Jure Leskovec**, Stanford University, USA **Gábor Lugosi**, Pompeu Fabra University, Spain **Ulrike von Luxburg**, University of Hamburg, Germany **Sridhar Mahadevan**, University of Massachusetts, Amherst, USA **Shie Mannor**, Technion, Israel **Chris Meek**, Microsoft Research, USA **Marina Meila**, University of Washington, USA **Nicolai Meinshausen**, University of Oxford, UK **Vahab Mirrokni**, Google Research, USA **Mehryar Mohri**, New York University, USA **Sebastian Nowozin**, Microsoft Research, Cambridge, UK **Manfred Opper**, Technical University of Berlin, Germany **Una-May O'Reilly**, Massachusetts Institute of Technology, USA **Ronald Parr**, Duke University, USA **Martin Pelikan**, Google Inc, USA **Jie Peng**, University of California, Davis, USA **Jan Peters**, Technische Universität Darmstadt, Germany **Avi Pfeffer**, Charles River Analytis, USA **Joelle Pineau**, McGill University, Canada **Massimiliano Pontil**, University College London, UK **Yuan (Alan) Qi**, Purdue University, USA **Luc de**

**Raedt**, Katholieke Universiteit Leuven, Belgium **Alexander Rakhlin**, University of Pennsylvania, USA **Ben Recht**, University of Wisconsin, Madison, USA **Saharon Rosset**, Tel Aviv University, Israel **Ruslan Salakhutdinov**, University of Toronto, Canada **Marc Schoenauer**, INRIA Saclay, France **Matthias Seeger**, Ecole Polytechnique Federale de Lausanne, Switzerland **John Shawe-Taylor**, University College London, UK **Xiaotong Shen**, University of Minnesota, USA **Yoram Singer**, Google Research, USA **Peter Spirtes**, Carnegie Mellon University, USA **Nathan Srebro**, Toyota Technical Institute at Chicago, USA **Ingo Steinwart**, University of Stuttgart, Germany **Ben Taskar**, University of Washington, USA **Yee Whye Teh**, University of Oxford, UK **Ivan Titov**, Saarland University, Germany **Koji Tsuda**, National Institute of Advanced Industrial Science and Technology, Japan **Zhuowen Tu**, University of California San Diego, USA **Nicolas Vayatis**, Ecole Normale Supérieure de Cachan, France **S V N Vishwanathan**, Purdue University, USA **Martin J. Wainwright**, University of California at Berkeley, USA **Manfred Warmuth**, University of California at Santa Cruz, USA **Stefan Wrobel**, Fraunhofer IAIS and University of Bonn, Germany **Eric Xing**, Carnegie Mellon University, USA **Bin Yu**, University of California at Berkeley, USA **Tong Zhang**, Rutgers University, USA **Hui Zou**, University of Minnesota, USA

JMLR-MLOSS Editors

**Mikio L. Braun**, Technical University of Berlin, Germany **Geoffrey Holmes**, University of Waikato, New Zealand **Antti Honkela**, University of Helsinki, Finland **Balázs Kégl**, University of Paris-Sud, France **Cheng Soon Ong**, University of Melbourne, Australia **Mark Reid**, Australian National University, Australia

JMLR Editorial Board

**Naoki Abe**, IBM TJ Watson Research Center, USA **Yasemin Altun**, Google Inc, Switzerland **Jean-Yves Audibert**, CERTIS, France **Jonathan Baxter**, Australia National University, Australia **Richard K. Belew**, University of California at San Diego, USA **Kristin Bennett**, Rensselaer Polytechnic Institute, USA **Christopher M. Bishop**, Microsoft Research, Cambridge, UK **Lashon Booker**, The Mitre Corporation, USA **Henrik Boström**, Stockholm University/KTH, Sweden **Craig Boutilier**, University of Toronto, Canada **Nello Cristianini**, University of Bristol, UK **Dennis DeCoste**, eBay Research, USA **Thomas Dietterich**, Oregon State University, USA **Jennifer Dy**, Northeastern University, USA **Saso Dzeroski**, Jozef Stefan Institute, Slovenia **Ran El-Yaniv**, Technion, Israel **Peter Flach**, Bristol University, UK **Emily Fox**, University of Washington, USA **Dan Geiger**, Technion, Israel **Claudio Gentile**, Università dell'Insubria, Italy **Sally Goldman**, Google Research, USA **Tom Griffiths**, University of California at Berkeley, USA **Carlos Guestrin**, University of Washington, USA **Stefan Harmeling**, University of Düsseldorf, Germany **David Heckerman**, Microsoft Research, USA **Katherine Heller**, Duke University, USA **Philipp Hennig**, MPI for Intelligent Systems, Germany **Larry Hunter**, University of Colorado, USA **Risi Kondor**, University of Chicago, USA **Aryeh Kontorovich**, Ben-Gurion University of the Negev, Israel **Andreas Krause**, ETH Zurich, Switzerland **Erik Learned-Miller**, University of Massachusetts, Amherst, USA **Fei Fei Li**, Stanford University, USA **Yi Lin**, University of Wisconsin, USA **Wei-Yin Loh**, University of Wisconsin, USA **Vikash Mansingkha**, Massachusetts Institute of Technology, USA **Yishay Mansour**, Tel-Aviv University, Israel **Jon McAuliffe**, University of California, Berkeley, USA **Andrew McCallum**, University of Massachusetts, Amherst, USA **Joris Mooij**, Radboud University Nijmegen, Netherlands **Raymond J. Mooney**, University of Texas, Austin, USA **Klaus-Robert Muller**, Technical University of Berlin, Germany **Guillaume Obozinski**, Ecole des Ponts - ParisTech, France **Pascal Poupart**, University of Waterloo, Canada **Cynthia Rudin**, Massachusetts Institute of Technology, USA **Robert Schapire**, Princeton University, USA **Fei Sha**, University of Southern California, USA **Shai Shalev-Shwartz**, Hebrew University of Jerusalem, Israel **Padhraic Smyth**, University of California, Irvine, USA **Le Song**, Georgia Institute of Technology, USA **Alexander Statnikov**, New York University, USA **Csaba Szepesvari**, University of Alberta, Canada **Jean-Philippe Vert**, Mines ParisTech, France **Chris Watkins**, Royal Holloway, University of London, UK **Kilian Weinberger**, Washington University, St Louis, USA **Max Welling**, University of

Amsterdam, Netherlands **Chris Williams**, University of Edinburgh, UK **David Wipf**, Microsoft Research Asia, China **Alice Zheng**, Microsoft Research Redmond, USA

JMLR Advisory Board

**Shun-Ichi Amari**, RIKEN Brain Science Institute, Japan **Andrew Barto**, University of Massachusetts at Amherst, USA **Thomas Dietterich**, Oregon State University, USA **Jerome Friedman**, Stanford University, USA **Stuart Geman**, Brown University, USA **Geoffrey Hinton**, University of Toronto, Canada **Michael Jordan**, University of California at Berkeley, USA **Leslie Pack Kaelbling**, Massachusetts Institute of Technology, USA **Michael Kearns**, University of Pennsylvania, USA **Steven Minton**, InferLink, USA **Tom Mitchell**, Carnegie Mellon University, USA **Stephen Muggleton**, Imperial College London, UK **Nils Nilsson**, Stanford University, USA **Tomaso Poggio**, Massachusetts Institute of Technology, USA **Ross Quinlan**, Rulequest Research Pty Ltd, Australia **Stuart Russell**, University of California at Berkeley, USA **Lawrence Saul**, University of California at San Diego, USA **Terrence Sejnowski**, Salk Institute for Biological Studies, USA **Richard Sutton**, University of Alberta, Canada **Leslie Valiant**, Harvard University, USA

# Journal of Machine Learning Research

Volume 14, 2013

Journal of Machine Learning Research 14 (2013) 1285-1302 Submitted 2/12; Revised 9/12; Published 5/13

# Regularization-Free Principal Curve Estimation

**Samuel Gerber** SGERBER@MATH.DUKE.EDU
*Mathematics Department*
*Duke University*
*Durham, NC 27708*

**Ross Whitaker** WHITAKER@CS.UTAH.EDU
*Scientific Computing and Imaging Institute*
*University of Utah*
*Salt Lake City, UT 84112*

**Editor:** Sridhar Mahadevan

## Abstract

Principal curves and manifolds provide a framework to formulate manifold learning within a statistical context. Principal curves define the notion of a curve passing through the middle of a distribution. While the intuition is clear, the formal definition leads to some technical and practical difficulties. In particular, principal curves are saddle points of the mean-squared projection distance, which poses severe challenges for estimation and model selection. This paper demonstrates that the difficulties in model selection associated with the saddle point property of principal curves are intrinsically tied to the minimization of the mean-squared projection distance. We introduce a new objective function, facilitated through a modification of the principal curve estimation approach, for which all critical points are principal curves and minima. Thus, the new formulation removes the fundamental issue for model selection in principal curve estimation. A gradient-descent-based estimator demonstrates the effectiveness of the new formulation for controlling model complexity on numerical experiments with synthetic and real data.

**Keywords:** principal curve, manifold estimation, unsupervised learning, model complexity, model selection

## 1. Introduction

Manifold learning is emerging as an important tool in many data analysis applications. In manifold learning, the assumption is that a given sample is drawn from a low-dimensional manifold embedded in a higher dimensional space, possibly corrupted by noise. However, the manifold is typically unknown and only observations from the ambient high-dimensional space are available. Manifold learning refers to the task of uncovering the low-dimensional manifold given the high-dimensional observations. Recent work in the machine learning community, such as Laplacian eigenmaps (Belkin and Niyogi, 2003), isomap (Tenenbaum et al., 2000), and locally linear embedding (Roweis and Saul, 2000) build directly on the manifold assumption and typically formulate the manifold estimation in terms of a graph embedding problem. The focus of these methods is to recover a low-dimensional description without modeling the manifold.

Hastie and Stuetzle (1989), motivated by the statistical properties of principal component analysis, introduced the notion of a principal curve passing through the *middle* of a distribution. The principal curve formulation casts manifold learning as a statistical fitting problem. Several ap-

©2013 Samuel Gerber and Ross Whitaker.

proaches that fit a manifold model, such as autoencoder neural networks (Hinton and Salakhutdinov, 2006), self-organizing maps (Kohonen, 1988) and unsupervised kernel regression (Meinicke et al., 2005), can be implicitly or explicitly related to the principal curve formulation. The principal curve approach to manifold learning can be advantageous over alternatives that exclusively estimate descriptive parameters. For instance, the expected distance of the data to the manifold provides a quantitative measure of the geometric fit to the data. Additionally, the formulation provides a consistent approach to project and reconstruct unseen data points which can be important in practical applications.

While several methods have been proposed to estimate principal curves, a systematic approach to model selection remains an open question. To facilitate the discussion, recall the formal definition of principal curves. Let $Y$ be a random variable with a smooth density $p$ such that the support $\Omega = \{y : p(y) > 0\}$ is a compact, connected region with a smooth boundary.

**Definition 1 (Principal Curve, Hastie and Stuetzle, 1989)** *Let* $g : \Lambda \to \mathbf{R}^n$, $\Lambda \subset \mathbf{R}$ *and* $\lambda_g : \Omega \to \Lambda$ *with* projection index $\lambda_g(y) = \max_s\{s : \|y - g(s)\| = \inf_{\tilde{s}}\|y - g(\tilde{s})\|\}$. *The principal curves of* $Y$ *are the set* $\mathcal{G}$ *of smooth functions* $g$ *that fulfill the self consistency property* $E[Y|\lambda_g(Y) = s] = g(s)$.

Hastie and Stuetzle (1989) showed that principal curves are critical points of the expected projection distance $d(g,Y)^2 = E[\|g(\lambda_g(Y)) - Y\|^2]$, that is, for $g$ a principal curve the (Fréchet) derivative of $d(g,Y)^2$ with respect to $g$ is zero. Estimators for principal curves typically minimizes $d(g,Y)^2$ directly over a suitable representation of $g$ (e.g., kernel smoother, piece-wise linear). However, Duchamp and Stuetzle (1996) showed, for principal curves in the plane, that all critical points are saddles. Thus, there are always adjacent (nonprincipal) curves with lower projection distance.

In fact, without additional regularization, principal curve estimation results in space-filling curves, which have both lower training and test projection distance than curves that provide a more *meaningful* summary of the distribution. Thus, all estimation approaches impose, implicit or explicit, regularity constraints on the curves (Tibshirani, 1992; Kégl et al., 2000; Chang and Ghosh, 2001; Smola et al., 2001; Meinicke et al., 2005) which, due to the saddle point property, will lead to estimates that lie on the constraint boundary and typically do not satisfy the conditions for principal curves. Using only the data, there is no well-defined, *correct* amount of regularization, and a regularization strategy would necessarily depend on extraneous considerations (e.g., the particular properties of the application). Thus, Duchamp and Stuetzle (1996) noted: "To our knowledge, nobody has yet suggested a reasonably motivated, automatic method for choice of model complexity in the context of manifold estimation or nonparametric orthogonal distance regression. This remains an important open problem". In this paper we propose a solution to address the model complexity challenge in principal curve estimation.

This paper contends that the model selection problem is intrinsically tied to the mean-squared-error objective function of the principal curve formulation. In the supervised setting, for example, for regression, the mean squared error provides an adequate measure for model selection (Härdle and Marron, 1985). In the unsupervised setting, for manifold estimation, the missing information on variable relationships renders distance measures inadequate. To be more precise, in regression for two observations with the same predictor measurements, the differences in predicted value is necessarily due to noise. Manifold estimation faces an inherent ambiguity in whether differences between observations should be treated as variation *orthogonal* to the manifold or as variation *along* the manifold, as illustrated in Figure 1.

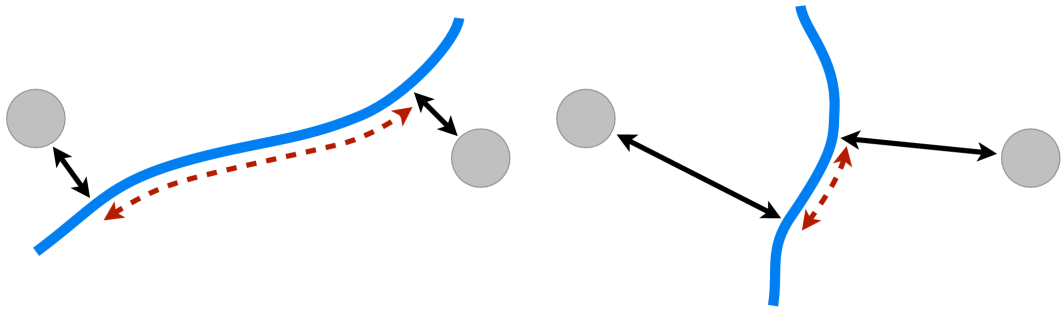


Figure 1: Two curves (solid lines) with a different trade off between variation along (dashed arrows) and orthogonal to (solid arrows) the curve for two data points in identical positions. A direct minimization of mean squared projection distance $d(g,Y)^2 = E[\|g(\lambda(Y)) - Y\|^2]$ will always favor curves passing close to the data points and not necessarily move towards a principal curve.

Informally, this indicates why principal curves are not minima of the distance function; trading off variation orthogonal to the manifold with variation along the curve leads to curves with smaller projection distances. This becomes evident in cross-validation, where a test example has no fixed assigned manifold coordinate (as compared to the fixed predictor variables in regression) but has manifold coordinates assigned based on the current fit. This leaves the flexibility to adapt the manifold coordinates of the test data to minimize projection distance, and leads, almost always, to simultaneous decreases in train and test error. This would not pose a significant problem if the principal curves would be minima of the mean squared projection distance—one would rely on the local minimum provided by the optimization. However, due to the saddle point property, the optimization moves towards a curve that passes through all the training data and cross-validation does not provide any reliable indication for early stopping. Thus, principal curve estimation based on minimizing the projection distance is not a viable path for automatic model selection.

Recent research produced some sophisticated approaches to automatically select the regularization parameters, such as the minimum description length approach by Wang and Lee (2006) and the oracle inequality approach by Biau and Fischer (2012). However, the fundamental problem persists; a regularized principal curve, is estimated which will always force the principal curve conditions to be violated. To expand on this point, consider this comparison to the regression setting. For nonparametric least squares regression, the regularization is a mechanism to cope with finite sample sizes. If the complete density is known, the regularization is unnecessary. The regression curve that minimizes the least squares objective function, the conditional expectation given the predictor, is the desired solution. Thus, with increasing sample size the amount of regularization required decreases to zero, for example, the bandwidth in kernel regression. For principal curves, the regularization is required even if the full density is available; the regularization serves as a crutch to fix a shortcoming in the objective function. This paper proposes an approach to principal curve estimation that fixes this shortcoming in the objective function. The proposed alternative objective function, for which

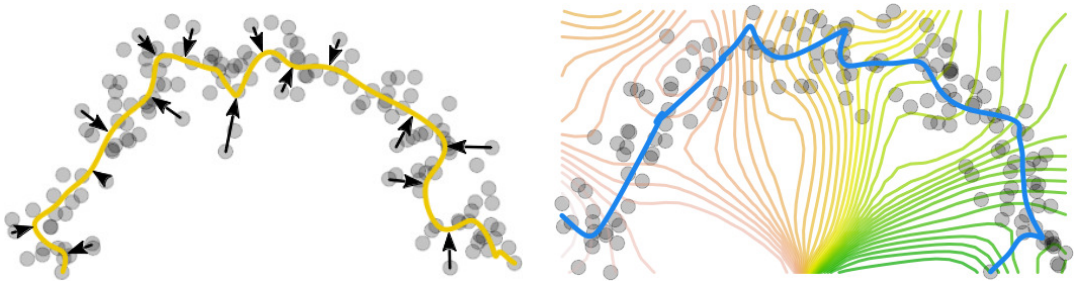


Figure 2: Illustration of the traditional approach compared to the proposed method for estimating principal curves. (left) The traditional approach specifies a curve $g$ (yellow) and defines $\lambda_g$ as the coordinate corresponding to the closest point on the curve. (right) The proposed formulation specifies and arbitrary $\lambda$—here a few level sets of $\lambda$ are shown—which in turn defines the curve $g = E[Y|\lambda(Y)]$ (blue). In the traditional approach $g$ is optimized to find a principal curve. In the new formulation the optimization is over $\lambda$.

principal curves emerge naturally as minima, alleviates the need for regularization or changes of the original principal curve definition.

The proposed formulation rests on the observation that minimizing $d(g,Y)^2$ allows principal curves $g$ to move away from saddle points and improve their fit by violating the self consistency property. This leads to the idea of turning the optimization problem around. Instead of having the projection index $\lambda_g$ controlled by the curve $g$, we fix $g \equiv E[Y|\lambda(Y)]$ to the conditional expectation given $\lambda$ and let $\lambda$ be an arbitrary function $\lambda : \Omega \mapsto \mathbf{R}$ (with mild conditions, to be specified in Section 2). To distinguish from Hastie and Stuetzle (1989), with $\lambda_g$ fixed given $g$, we call $\lambda$ the *coordinate mapping* and $E[Y|\lambda(Y)]$ the *conditional expectation curve* of $\lambda$. The difference between the two formulations is illustrated in Figure 2. Now, the strategy is to optimize over the coordinate mapping $\lambda$ rather than $g$. Minimizing $d(\lambda,Y) = E\left[\|g(\lambda(Y)) - Y\|^2\right]$ with respect to $\lambda$ leads to critical points that are, indeed, principal curves, but unfortunately they are still saddle points. In this case, escaping from saddle points is possible by violating the orthogonal projection condition. This inspires the design of an objective function, facilitated through this level-set based formulation, that penalizes nonorthogonality of the coordinate mapping $\lambda$

$$q(\lambda,Y)^2 = E\left[\left\langle (g(\lambda(Y)) - Y), \frac{d}{ds}g(s)\Big|_{s=\lambda(Y)}\right\rangle^2\right].$$

Here, $g$ is by definition self-consistent and the orthogonal projection of $\lambda$ is enforced by the objective function. In this formulation, all critical points $\lambda^*$ are minima with $q(\lambda^*,Y)^2 = 0$. Hence, the conditional expectation curve at a minima $g \equiv E[Y|\lambda^*(Y)]$ corresponds to a principal curve (see Section 2).

The proposed formulation leads to a practical estimator, that does not impose any model complexity penalty in its estimate. This regularization-free estimation scheme has the remarkable property that a minimum on training data typically corresponds to a desirable solution (see Sections 3 and 4).

## 2. Conditional Expectation Curves

Let $Y$ be a random variable with a smooth density $p$ with support $\tilde{\Omega} = \{y : p(y) > 0\} \subset \mathbf{R}^n$ such that the closure $\Omega = \mathrm{cl}(\tilde{\Omega})$ is a compact, connected region with smooth boundary. This includes densities $p$ that smoothly go to 0 at the boundary with an open support as well as densities with a compact support. Let $\lambda : \Omega \to \Lambda$ be Lipschitz with $\Lambda$ a connected closed subset of $\mathbf{R}$. To shorten notation, let $\mu$ be the measure such that $p$ is the Radon-Nikodym derivative of $\mu$ with respect to the Lebesgue measure and write $d\mu = d\mu(y) = \mu(dy) = p(y)dy$.

As for principal curve setting, the conditional expectation $E[Y|\lambda(Y)]$ is defined over a set of measure zero and is not a priori well defined. To formally define the conditional expectation let

$$g_\sigma(s) = \frac{\int_\Omega K_\sigma(s-\lambda(y))y d\mu}{\int_\Omega K_\sigma(s-\lambda(y))d\mu}$$

with $K_\sigma(s)$ a mollifier. Now we define the conditional expectation as the limit

$$\begin{aligned} E[Y|\lambda(Y)=s] &= \lim_{\sigma\to 0} g_\sigma(s) = \frac{\int_\Omega \delta(\lambda(y)-s)y d\mu}{\int_\Omega \delta(\lambda(y)-s)d\mu} \\ &= \frac{\int_{\lambda^{-1}(s)} y p(y) J_\lambda(y)^{-1} d\mathcal{H}_{n-1}(y)}{\int_{\lambda^{-1}(s)} p(y) J_\lambda(y)^{-1} d\mathcal{H}_{n-1}(y)} \,. \end{aligned} \tag{1}$$

with $J_\lambda(y)$ the 1-dimensional Jacobian of $\lambda$ and $\mathcal{H}_{n-1}$ dimensional Hausdorff measure. The second equality follows from the uniform convergence of the mollifier to the Dirac delta distribution. The last equality invokes the coarea formula (Federer, 1969) by changing to integration over the level sets of $\lambda$. This definition is, for $\lambda$ an orthogonal projection, equivalent to the conditional expectation as understood in Hastie and Stuetzle (1989) and Duchamp and Stuetzle (1996). This formulation also directly extends to define conditional expectation manifolds with $\Lambda \subset \mathbf{R}^m$.

Depending on $\lambda$ and $\Omega$ the conditional expectation (1) can be discontinuous, for example, for $\lambda$ with constant regions or for $\lambda$ and $\Omega$ that result in *abrupt* topological or geometrical changes in the level sets. However, additional conditions on $\lambda$ and $\Omega$, ensure that $g$ is a differentiable.

**Lemma 2** *For $\lambda \in C^2(\Omega)$, $g$ is differentiable $\Lambda$-almost everywhere.*

**Proof** The derivative of $g$ is

$$\frac{d}{ds}g(s) = \frac{\frac{d}{ds}m(s)}{n(s)} - g(s)\frac{\frac{d}{ds}n(s)}{n(s)} \,.$$

with

$$\begin{aligned} m(s) &= \int_{\lambda^{-1}(s)} y p(y) J_\lambda(y)^{-1} d\mathcal{H}_{n-1}(y) \,, \\ n(s) &= \int_{\lambda^{-1}(s)} p(y) J_\lambda(y)^{-1} d\mathcal{H}_{n-1}(y) \,. \end{aligned}$$

By Sard's theorem, the set of critical values of $\lambda$ has measure zero in $\Lambda$, and by the implicit function theorem, $\lambda^{-1}(s)$ is a Riemannian manifold $\Lambda$-almost everywhere. Thus, integrating with respect to

the Hausdorff measure is equivalent to integration over the manifold $\lambda^{-1}(s)$, $\Lambda$-almost everywhere. Then, by the general Leibniz rule (Flanders, 1973), the derivatives $\frac{d}{ds}m(s)$ and $\frac{d}{ds}n(s)$ are

$$\frac{d}{ds}\int_{\lambda^{-1}(s)}\omega = \int_{\lambda^{-1}(s)} i(\mathbf{v}, d_y\omega) + \int_{\partial\lambda^{-1}(s)} i(\mathbf{v},\omega) + \int_{\lambda^{-1}(s)}\frac{d}{ds}\omega \tag{2}$$

with $i(\cdot,\cdot)$ the interior product, $d_y$ the exterior derivative with respect to $y$ and $\omega$ the differential $(n-1)$-form $yp(y)J_\lambda(y)^{-1}d\lambda^{-1}(s)$ for $\frac{d}{ds}m(s)$ and $p(y)J_\lambda(y)^{-1}d\lambda^{-1}(s)$ for $\frac{d}{ds}n(s)$, respectively. The vector field $\mathbf{v}$ is the change of the manifold $\lambda^{-1}(s)$ with respect to $s$. For $\Omega$ with smooth boundary $\partial\Omega$, the restriction $\lambda|_{\partial\Omega}$ is also in $C^2(\partial\Omega)$. Thus, by the implicit function theorem, $\mathbf{v}$ exist $\Lambda$-almost everywhere for both $\lambda$ and $\lambda|_{\partial\Omega}$ and it follows that $\frac{d}{ds}g(s)$ exists $\Lambda$-almost everywhere. ■

**Corollary 3** *If $\lambda \in C^2(\Omega)$, $J_\lambda(y) > 0$ $\Omega$-almost everywhere and $J_{\lambda|_{\partial\Omega}}(y) > 0$, $\partial\Omega$-almost everywhere then $g$ is differentiable everywhere.*

The conditions in Corollary 3 ensure that the integrals in Equation (2) are well defined for all $s \in \Lambda$ and hence, the derivative of $g$ is defined everywhere as well. As for the conditional expectation (1), Lemma 2 and Corollary 3 do not rest on a scalar $\lambda$ and directly extends to conditional expectation manifolds.

With the definition of $g \equiv E[Y|\lambda(Y)]$, the conditional expectation given a coordinate mapping $\lambda$, one is tempted to minimize the distance function

$$d(\lambda, Y)^2 = E\left[\|g(\lambda(Y)) - Y\|^2\right]$$

with respect to $\lambda$. This formulation does, indeed, lead to conditional expectations at critical points that are principal curves and the self consistency cannot be violated by definition. Unfortunately, critical points are still saddles. In this case, $\lambda$ can move towards conditional expectation curves with lower objective by violating the orthogonal projection requirement (rather than violating the self consistency requirement, as happens in the original formulation). However, the switch from optimization of curves $g$ to optimization of coordinate mappings $\lambda$ permits an alternative objective function that penalizes nonorthogonal $\lambda$:

$$q(\lambda, Y)^2 = E\left[\left\langle g(\lambda(Y)) - Y, \frac{d}{ds}g(s)\Big|_{s=\lambda(Y)}\right\rangle^2\right]. \tag{3}$$

For $q(\lambda, Y)$ to be well defined $\frac{d}{ds}g(s)$ has to exist $\Omega$-almost everywhere and thus, requires the conditions of Corollary 3.

The next Section shows that all critical points of (3) are minima and that the corresponding conditional expectation curves are principal curves in the sense of Hastie and Stuetzle (1989).

### 2.1 Properties of Critical Points

The following definition introduces a slightly weaker version of principal curves. For principal curves which have no ambiguity points, that is, all $y \in \Omega$ have a unique closest point on the curve, the definition is equivalent to principal curves.

**Definition 4 (Weak Principal Curves)** *Let $g : \Lambda \to \mathbf{R}^n$ and $\lambda : \Omega \to \Lambda$. The weak principal curves of $Y$ are the set $\mathcal{G}_w$ of functions $g$ that fulfill the self consistency property $E[Y|\lambda(Y) = s] = g(s)$ with $\lambda$ satisfying $\left\langle y - g(\lambda(y)), \frac{d}{ds} g(s)\big|_{s=\lambda(y)} \right\rangle = 0 \, \forall y \in \Omega$.*

Weak principal curves do include all principal curves but can potentially also include additional curves for $\lambda_g$ with discontinuities, that is, curves with ambiguity points. However, under restriction to continuous $\lambda$, which excludes principal curves with ambiguity points, the set of principal curves and weak principal curves are identical under this restriction. Furthermore, in practical applications principal curves with ambiguity points are typically not of interest.

**Theorem 5** *The conditional expectation curves of critical points of $q(\lambda, Y)^2$ are weak principal curves.*

**Proof** The self-consistency property follows immediately from the definition of $g$ as conditional expectation given $\lambda$. Thus, to show that critical points of $q(\lambda, Y)^2$ are principal curves, requires $\frac{d}{ds} g(s)$ to be orthogonal to $g(s) - y$ almost everywhere for $y$ in the level set $\lambda^{-1}(s)$.

Let $\tau$ be an admissible variation of $\lambda$, that is, $\tau \in C^3$ and satisfies the conditions of Corollary (3). Taking the Gâteaux derivative of $q(\lambda, Y)^2$ with respect to $\tau$ yields

$$\frac{d}{d\varepsilon} q(\lambda + \varepsilon\tau, Y)^2 \Big|_{\varepsilon=0} = \int_\Omega \left\langle (g(\lambda(y)) - y), \frac{d}{ds} g(s) \Big|_{s=\lambda(y)} \right\rangle \frac{d}{d\varepsilon} \left\langle (g(\lambda(y) + \varepsilon\tau(y)) - y), \frac{d}{ds} g(s) \Big|_{s=\lambda(y+\varepsilon\tau(y))} \right\rangle \Big|_{\varepsilon=0} d\mu .$$

It immediately follows $\lambda$ is critical if it is orthogonal to its conditional expectation curve $\Omega$-almost everywhere.

To exclude other possible critical points requires that there is no $\lambda$ for which

$$\left\langle \frac{d}{d\varepsilon} g(\lambda(y) + \varepsilon\tau(y)) \Big|_{\varepsilon=0}, \frac{d}{ds} g(s) \Big|_{s=\lambda(y)} \right\rangle + \left\langle (g(\lambda(y)) - y), \frac{d}{d\varepsilon} \frac{d}{ds} g(s) \Big|_{s=\lambda(y+\varepsilon\tau(y))} \Big|_{\varepsilon=0} \right\rangle = 0$$

for all admissible $\tau$.

For the variation of $g(\lambda(y))$ take the Gâteaux derivative of the mollified expectation

$$\frac{d}{d\varepsilon} g_\sigma(\lambda(y) + \varepsilon\tau(y)) \Big|_{\varepsilon=0} = \frac{\int_\Omega K'_\sigma(\lambda(y) - \lambda(\tilde{y}))(\tau(y) - \tau(\tilde{y}))\big(\tilde{y} - g_\sigma(\lambda(y))\big) d\mu(\tilde{y})}{\int_\Omega K_\sigma(\lambda(y) - \lambda(\tilde{y})) d\mu(\tilde{y})}$$

with $K'_\sigma(s - \lambda(\tilde{y})) = \frac{d}{ds} K_\sigma(s - \lambda(\tilde{y}))$. Separating the terms into $\tau(y)$ and $\tau(\tilde{y})$ yields

$$\frac{d}{d\varepsilon} g_\sigma(\lambda(y) + \varepsilon\tau(y)) \Big|_{\varepsilon=0} = \tau(\lambda(y)) \frac{d}{ds} g(s) \Big|_{s=\lambda(y)} + \alpha_\sigma(\lambda(y), \tau)$$

with

$$\alpha_\sigma(s, \tau) = \frac{\int_\Omega K'_\sigma(s - \lambda(\tilde{y}))\tau(\tilde{y})\,(\tilde{y} - g_\sigma(s))\, d\mu}{\int_\Omega K_\sigma(s - \lambda(\tilde{y})) d\mu}$$

The same procedure for $\frac{d}{ds}g(s)$ yields

$$\begin{aligned}\frac{d}{d\varepsilon}\left.\frac{dg_\sigma(s)}{ds}\right|_{s=\lambda(y)+\varepsilon\tau(y)}\Bigg|_{\varepsilon=0} =& \frac{\int_\Omega K''_\sigma(\lambda(y)-\lambda(\tilde{y}))(\tau(y)-\tau(\tilde{y}))\,(y-g_\sigma(\lambda(y)))\,d\mu(\tilde{y})}{\int_\Omega K_\sigma(\lambda(y)-\lambda(\tilde{y}))d\mu(\tilde{y})}\\ &- \frac{\int_\Omega K'_\sigma(\lambda(y)-\lambda(\tilde{y}))(\tau(y)-\tau(\tilde{y}))d\mu}{\int_\Omega K_\sigma(\lambda(y)-\lambda(\tilde{y}))d\mu(\tilde{y})}\left.\frac{d}{ds}g_\sigma(s)\right|_{s=\lambda(y)}\\ &- \left.\frac{d}{ds}g_\sigma(s)\right|_{s=\lambda(y)}\left.\frac{d}{d\varepsilon}g(\lambda(y)+\varepsilon\tau(y))\right|_{\varepsilon=0}\end{aligned}$$

and by separating the terms

$$\frac{d}{d\varepsilon}\left.\frac{dg_\sigma(s)}{ds}\right|_{s=\lambda(y)+\varepsilon\tau(y)}\Bigg|_{\varepsilon=0} = \tau(y)\left.\frac{d^2}{ds^2}g_\sigma(s)\right|_{s=\lambda(y)} + \beta_\sigma(\lambda(y),\tau)$$

with

$$\begin{aligned}\beta_\sigma(s,\tau) =& -\frac{\int_\Omega K''_\sigma(s-\lambda(\tilde{y}))\tau(\tilde{y})\,(y-g_\sigma(s))\,d\mu}{\int_\Omega K_\sigma(s-\lambda(\tilde{y}))d\mu}\\ &+\frac{\int_\Omega K'_\sigma(s-\lambda(\tilde{y}))\tau(\tilde{y})d\mu}{\int_\Omega K_\sigma(s-\lambda(\tilde{y}))d\mu}\frac{d}{ds}g_\sigma(s) - \frac{d}{ds}g_\sigma(s)\alpha_\sigma(s,\tau)\,.\end{aligned}$$

Terms of the form $\int_\Omega K'_\sigma(s-\lambda(\tilde{y})f(\tilde{y})d\tilde{y}$ converge uniformly to $\int_\Omega \delta'(s-\lambda(\tilde{y})f(\tilde{y})d\tilde{y} = \frac{d}{ds}\int_{\lambda^{-1}(s)} f(\tilde{y})d\tilde{y}$ which under the conditions of Lemma (2) exist. Thus, $\frac{d}{ds}g_\sigma(s)$ converges uniformly and with the uniform convergence of $g_\sigma \to g$ yields the limit $\lim_{\sigma\to 0}\frac{d}{ds}g_\sigma(s) = \frac{d}{ds}g(s)$. With $\alpha(s,\tau) = \lim_{\sigma\to 0}\alpha_\sigma(s,\tau)$ and $\beta(s,\tau) = \lim_{\sigma\to 0}\beta_\sigma(s,\tau)$ and taking the limit as $\sigma\to 0$ yields

$$\begin{aligned}&\frac{d}{d\varepsilon}\left\langle (g(\lambda(y)+\varepsilon\tau(y))-y)\,,\,\left.\frac{d}{ds}g(s)\right|_{s=\lambda(y+\varepsilon\tau(y))}\right\rangle\Bigg|_{\varepsilon=0}\\ =&\tau(y)\left\|\left.\frac{d}{ds}g(s)\right|_{s=\lambda(y)}\right\|^2 + \left\langle \left.\frac{d}{ds}g(s)\right|_{s=\lambda(y)}, \alpha(\lambda(y),\tau)\right\rangle\\ &+\left\langle g(\lambda(y))-y, \left(\tau(y)\left.\frac{d^2}{ds^2}g(s)\right|_{s=\lambda(y)} + \beta(\lambda(y),\tau)\right)\right\rangle.\end{aligned}$$

Note that

$$\int_{\lambda^{-1}(s)}(g(s)-y)\,d\mu = g(s)\int_{\lambda^{-1}(s)}d\mu - \int_{\lambda^{-1}(s)} y d\mu = 0\,,$$

and thus the term $\left\langle \alpha(\lambda(y),\tau),\, \frac{d}{ds}g(s)\big|_{s=\lambda(y)}\right\rangle$ with constant value along the level sets of $\lambda$ does not contribute to the variation. Therefore, a critical point for which

$$\left\langle g(\lambda(y))-y,\, \left.\frac{d}{ds}g(s)\right|_{s=\lambda(y)}\right\rangle \neq 0$$

$\Omega$-almost everywhere, there must be a $\lambda$ for which

$$\left\|\left.\frac{d}{ds}g(s)\right|_{s=\lambda(y)}\right\|^2 + \left\langle g(\lambda(y))-y,\, \left.\frac{d^2}{ds^2}g(s)\right|_{s=\lambda(y)} + \beta(\lambda(y),\tau)\right\rangle = 0$$

$\Omega$-almost everywhere. Since $\beta(\lambda(y),\tau)$, $\frac{d}{ds}g(s)\big|_{s=\lambda(y)}$ and $\frac{d^2}{ds^2}g(s)\Big|_{s=\lambda(y)}$ are constant along the level set of $\lambda$ requires that $\left\| \frac{d}{ds}g(s)\big|_{s=\lambda(y)} \right\|^2 = 0$ and either $g(\lambda(y)) - y = 0$ or

$$\left\langle g(\lambda(y)) - y, \frac{d^2}{ds^2}g(s)\Big|_{s=\lambda(y)} + \beta(\lambda(y),\tau) \right\rangle = 0\,,$$

$\Omega$-almost everywhere. It follows that for all critical points

$$\left\langle g(\lambda(y)) - y, \frac{d}{ds}g(s)\Big|_{s=\lambda(y)} \right\rangle = 0\,.$$

■

**Corollary 6** *All critical points of $q(\lambda,Y)^2$ are minima.*

**Proof** By definition $q(\lambda,Y)^2 \geq 0$ and by Theorem 5 $q(\lambda,Y)^2 = 0$ for all critical points. ■

**Corollary 7** *The coordinate mapping $\lambda$ is a minima of $q(\lambda,Y)$ if and only if $E[Y|\lambda(Y)]$ is a weak principal curve.*

**Proof** If $\lambda$ is minimal it follows from Theorem 5 that $E[Y|\lambda(Y)]$ is a weak principal curve. If $E[Y|\lambda(Y)]$ is a principal curve $\lambda$ has to be an orthogonal projection and thus $q(\lambda,Y)^2 = 0$ is minimal. ■

### 2.2 Remarks

The minima include *pathological* conditional expectation sets such as single points, curves with discontinuities and space-filling curves. However, if $\Omega$ and $p$ admit a minimal $\lambda$ that satisfies the condition in Lemma 2, the conditional expectation is a well behaved curve $\Lambda$-almost everywhere.

In principle, $q(\lambda,Y)^2$ can be minimized by shrinking the tangent $\frac{d}{ds}g(s)$, that is, through a reparametrization of $g$. If $\lambda$ is constrained so that the resulting conditional expectation has a fixed-length tangent, for example, a unit parametrization, or if the objective function is changed to

$$q_n(\lambda,Y)^2 = E\left[ \frac{\left\langle g(\lambda(Y)) - Y, \frac{d}{ds}g(s)\big|_{s=\lambda(Y)} \right\rangle^2}{\left\| \frac{d}{ds}g(s)\big|_{s=\lambda(Y)} \right\|^2} \right],$$

then shortening the tangent does not change the objective. However, in practice, using the approach described in Section 3, this problem did not arise and the empirical differences between minimizing $q_n$ and $q$ were negligible.

At critical points (local minima), the length of the tangent $\frac{d}{ds}g(s)$ has no effect on either the objective function or the solution. Changing the coordinate mapping to $\tilde{\lambda}(y) = \eta(\lambda(y))$ with $\eta$ :

$\Lambda \mapsto \mathbf{R}$ differentiable and monotonic results in a re-parametrization of the conditional expectation curve, thus, the minimal sets consist of an infinite number of curves. Only for a re-parametrization $\eta(s) = c$, that is, a constant $\lambda$, is the geometry of the curve changed to a single point. Thus, the set of local minima of this objective form a single connected set, with many re-parametrizations and different curves connected at the solution $\lambda = c$. In terms of the objective function $q(\lambda, Y)$, these are all equally good solutions. Similar to principal components, choosing among the possible principal curves an alternate criterion, possibly application dependent, is required. An obvious choice is the mean squared projection distance and exclude the trivial space filling solution with zero mean squared projection distance.

## 3. An Algorithm for Data Sets

Gerber et al. (2009) proposed an algorithm to minimize $d(\lambda, Y)^2$ using a radial basis function expansion for $\lambda$ and a kernel regression for the conditional expectation $g$. For a data set $Y = \{y_i\}_1^n$, the radial basis function expansion is

$$\lambda(y) = \frac{\sum_{j=1}^n K(y - y_j) z_j}{\sum_{j=1}^n K(y - y_j)}$$

with $K$ a Gaussian kernel with bandwidth $\sigma_\lambda$ and $Z = \{z_i\}_1^n \in \mathbf{R}$ a set of parameters. The parameters $Z$ are a set of locations in $\Lambda$, the range of the coordinate mapping $\lambda$. Each point in $\Omega$ is thus mapped to a weighted average of the parameters $Z$. The expected squared projection distance $d(\lambda, Y)^2$ is estimated with the sample mean-squared projection distance $\hat{d}(\lambda, Y)^2 = \frac{1}{n} \sum_{i=1}^n \| g(\lambda(y_i)) - y_i \|^2$. Gerber et al. (2009) employ a gradient descent on $Z$ to minimize $\hat{d}(\lambda, Y)^2$. Here, the same approach is used for minimizing

$$\hat{q}(\lambda, Y) = \frac{1}{n} \sum_{i=1}^n \left\langle g(\lambda(y_i)) - y_i, \frac{d}{ds} g(s) \Big|_{s=\lambda(y_i)} \right\rangle^2 \tag{4}$$

but instead of a Nadaraya-Watson kernel regression (Nadaraya, 1964; Watson, 1964) for $g$ we use a locally linear regression (Cleveland, 1979), for better behavior for data near the boundaries of $\Lambda$. The algorithm is implemented in R in the package *cems* (Gerber, 2011).

The gradient of (4) is

$$\begin{aligned} \frac{d}{dz_k} \hat{q}(\lambda, Y)^2 = \sum_{i=1}^n & \left\langle \left( g(\lambda(y_i)) - y_i \right), \frac{d}{ds} g(s) \Big|_{s=\lambda(y_i)} \right\rangle \\ & \left[ \left\langle \frac{d}{dz_k} g(\lambda(y_i)), \frac{d}{ds} g(s) \Big|_{s=\lambda(y_i)} \right\rangle \right. \\ & \left. + \left\langle \left( g(\lambda(y_i)) - y_i \right), \frac{d}{dz_k} \frac{d}{ds} g(s) \Big|_{s=\lambda(y_i)} \right\rangle \right]. \end{aligned} \tag{5}$$

This has a similar form as the first variation of $q(\lambda, Y)^2$ and for $\lambda$ orthogonal to $g$ $\hat{q}(\lambda, Y)$ is also minimal. However, there may exist additional local minima for which $\lambda$ is not an orthogonal projection onto $g$.

Because this is a local optimization, one must consider the initialization. The parameters $Z$ can be initialized by principal components, a spectral manifold learning method (Tenenbaum et al., 2000; Roweis and Saul, 2000; Belkin and Niyogi, 2003), or by some heuristic that is particular to the application. The algorithm generalizes in a straightforward manner to $m$-dimensional manifolds by initializing $Z$ to points in $\mathbf{R}^m$ and replacing dot products with the appropriate matrix-vector multiplications.

### 3.1 Model Complexity through Automatic Bandwidth Selection

Gerber et al. (2009) reported good results with cross-validation for minimizing the mean-squared projection distance $\hat{d}(\lambda, Y)^2$ starting from a $\lambda$ that results in a smooth estimate $g$. However, the optimization was based on fixed bandwidths for the kernel regression and radial basis function expansion and thus imposed implicitly a regularization on the estimate.

In the proposed computational scheme, there are two bandwidth selection problems $\sigma_\lambda$ for $\lambda$ and $\sigma_g$ for $g$ that control model complexity. The bandwidth for $\lambda$ determines a degree of smoothness in the coordinate mapping and thereby provides a form of regularization. For sufficiently small $\lambda$, the regularization effect is negligible but can negatively affect generalization performance. The bandwidth of $g$ determines the smoothness of the conditional expectation estimate. By shrinking $\sigma_g$ towards zero the estimate will be forced to pass through the training data.

For automatically deducing model complexity the bandwidth selection is incorporated into the optimization. The optimization alternates between gradient descent steps of the spatial locations $Z$ (5) of the radial basis function expansion $\lambda$ and gradient descent steps of the bandwidths

$$\frac{d}{d\sigma}\hat{q}(\lambda, Y)^2 = \left(\frac{\partial}{\partial\sigma_g}\hat{q}(\lambda, Y)^2, \frac{\partial}{\partial\sigma_\lambda}\hat{q}(\lambda, Y)^2\right)$$

with stopping based on held out data.

The empirical results bear out the theoretical arguments in Section 2: cross-validation for bandwidth selection for $g$ and $\lambda$ is not feasible when minimizing projection distance. The minimization of $\hat{d}(\lambda, Y)^2$ with bandwidth selection tends towards curves with increased curvature, possibly moving away from a *close by* principal curve that is smoother than the initial curve (see Figures 3 and 5). Even starting from smooth curves cross-validation is not a reliable indicator for estimating a good, approximating principal curve (see Figure 4). This, empirically, demonstrates the saddle point nature of the critical points. Minimizing $\hat{q}(\lambda, Y)$ mitigates these issues and shows a desirable objective-function landscape (see Figures 3, 4 and 5).

For minimizing $\hat{q}(\lambda, Y)^2$ stopping based on cross-validation data was unnecessary in all numerical experiments—the test error follows closely in step with the training error. This nice property is not unexpected and hinges directly on the theoretical results of the new formulation to principal curves. Over-fitting in the case of minimizing $\hat{q}(\lambda, Y)^2$ requires a $\lambda$ which is close to orthogonal for training data but not on the test data. However, the objective function $\hat{q}(\lambda, Y)^2$ indicates no preference towards such a particular orthogonal $\lambda$ on the training data and does not express a desire to increase the length/complexity of the curve in order to achieve a minima. Furthermore, a minima of $q(\lambda, Y)^2$ given the complete density $p$ is also a minima for the empirical objective $\hat{q}(\lambda, Y)^2$ on a sample of $Y$. This is not true for the objective function $d(\lambda, Y)^2$ or even for least squares regression in the supervised setting (however, for regression cross-validation is a viable option as mentioned in the discussion in Section 1). These two observations provide an explanation of the remarkable

property of the proposed estimation scheme that a regularization-free minimum on the training data is typically a desirable principal curve.

### 3.2 Computational Considerations

The observation on the properties of the regularization suggest the optimization strategy of $Z, \sigma_\lambda$ and $\sigma_g$ in Algorithm 1, which proves to be effective in practice (see Section 4).

**Data**: $Y$
**Result**: Conditional expectation curve
Initialize $Z$ from $Y$ ( isomap, PCA, LLE, ... ) ;
Initialize $\sigma_\lambda$ to average $k$-nearest neighbor distance of $Y$;
Initialize $\sigma_g$ to average $k$-nearest neighbor distance of $\lambda(Y)$ ;
**while** $\hat{q}(\lambda, Y)$ *is improving* **do**
  **while** $\hat{q}(\lambda, Y)$ *is improving* **do**
    Gradient descent step in $\sigma_\lambda$ ;
  **end**
  Gradient descent step in $\sigma_g$ ;
  Gradient descent step in $Z$ ;
**end**

**Algorithm 1:** Conditional expectation curve optimization strategy

Various alternatives and extensions for the algorithm proposed here are possible, such as the choice of the conditional expectation estimator and the modeling of $\lambda$. A naive implementation of the algorithm has a complexity of $O(n^3md)$ per gradient descent step with $n$ the number of points, $m$ the intrinsic dimensionality of the manifold and $d$ the ambient dimensionality. This can be reduced to $O(nc^2md)$ by taking advantage of local nature of the kernel based formulation for both $g$ an $\lambda$. For any $y_i$ and $z_i$ only approximately $c$ data points, with $c$ depending on the bandwidth parameter, are effectively contributing to the mapping. A detailed analysis of these run time complexity results is in Gerber et al. (2009). For further speed up one can consider reducing the number of parameters $Z$ in the mapping $\lambda$. The mapping complexity of $\lambda$ only depends on the complexity of the principal curve and can be independent of the number of data points. Since the complexity can conceivably vary spatially, one should consider an adaptive algorithm that introduces more or less *control points* $Z$ in $\lambda$ as required and varies the bandwidth spatially as well. While the spatial varying bandwidth is essentially included in the optimization through spreading the parameters $Z$ differently, an implementation incorporating variable bandwidths is potentially more effective in practice.

## 4. Numerical Experiments

The numerical experiments illustrate the theoretical results on the critical point properties of $q(\lambda, Y)^2$ and $d(\lambda, Y)^2$ using the estimation strategy proposed in Section 3. Gerber et al. (2009) report extensive results on the optimization of $\hat{d}(\lambda, Y)^2$ with equal or improved performance compared to several principal curve algorithms in various settings. Thus, the comparison is representative to a typical principal curve estimation scheme based on minimizing the squared projection distance. The results demonstrate that automatic bandwidth selection leads to good results for $\hat{q}(\lambda, Y)^2$, while minimizing $\hat{d}(\lambda, Y)^2$ shows the expected behavior of tending towards curves with high curvature that pass close to the training data, even with early stopping based on held-out data.

### 4.1 Conditional Expectation Curves in the Plane

Here we demonstrate the observations on optimization and bandwidth selection on a 2D data set generated by $a(\phi) = (\sin(\phi), \cos(\phi))^t(1+\varepsilon)$. A 180 degree arc of radius one and $\varepsilon \approx \mathcal{N}(0, \eta^2)$ normal distributed noise added orthogonal to the arc. To generate $n$ data points $\phi$ is sampled uniformly on $[0, \pi]$.

Figure 3 shows optimization results for $n = 150$ and $\eta = 0.15$ but with $Z = \sin(\phi)$ initialized close to a principal component. Starting from bandwidths $\sigma_g = 0.2$ and $\sigma_\lambda = 0.1$, the minimization of $\hat{d}(\lambda, Y)$ moves away from the principal component while $\hat{q}(\lambda, Y)$ approximately recovers the principal component. These observations also hold for other bandwidth initializations.

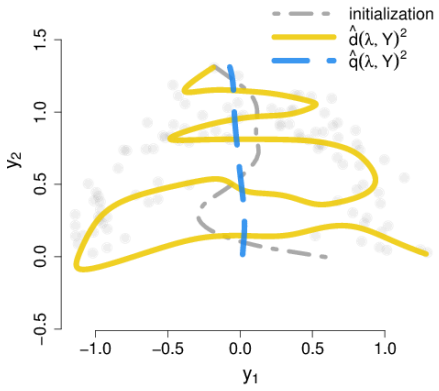



Figure 3: Minimization of $\hat{d}(\lambda, Y)^2$ and $\hat{q}(\lambda, Y)^2$ starting from a curve close to a principal component with bandwidths initialized to $\sigma_g = 0.2$ and $\sigma_\lambda = 0.1$.

Figures 4 and 5 show optimization results for $n = 150$, $\eta = 0.25$ and $Z = \phi$ initialized to the ground truth starting from different bandwidth initialization. Once again, minimizing $\hat{d}(\lambda, Y)^2$ favors curves with high curvature. The minimization of $\hat{q}(\lambda, Y)^2$ moves towards the ground truth, even for a small initial bandwidth.

The theoretical results give no guarantee/bounds on the radius of convergence around solutions. However,in all numerical experiments the optimization moved towards the anticipated solution.

### 4.2 Conditional Expectation Surfaces in 3D

The numerical results on algorithm behavior generalize to conditional expectation surfaces. This section replicates the results from Section 4.1 on the synthetic *Swissroll* data, a 2D-manifold in 3D-space (illustrated in Figures 6 and 7), commonly used to validate manifold learning techniques. The Swissroll is a parametric surface $s(h, \phi) = (\cos(\phi)\phi, h, \sin(\phi)\phi)$. The data for the experiments are generated by drawing observations $(h_i, \phi_i) \in H \times \Phi$, with $H = [0, 4]$ and $\Phi = [\pi, 3\pi]$ and adding normal $\mathcal{N}(0, \frac{1}{2})$ distributed noise orthogonal to $s(h_i, \phi_i)$. Note, this induces a nonuniform sample of the Swissroll, where the density in the ambient space decreases with increasing radius.

Figure 6 shows results for minimizing $\hat{d}(\lambda, Y)^2$ and $\hat{q}(\lambda, Y)^2$ on a training sample with $n = 1000$ drawn uniformly on $H \times \Phi$. The optimization of $Z$ is initialized by a 2D isomap embedding using 15-nearest neighbors. Bandwidths are initialized by the average 75-nearest neighbor distance of $\lambda(Y)$ for $\sigma_g$ and the average 25-nearest neighbor distance of $Y$ for $\sigma_\lambda$. The same behavior as for the

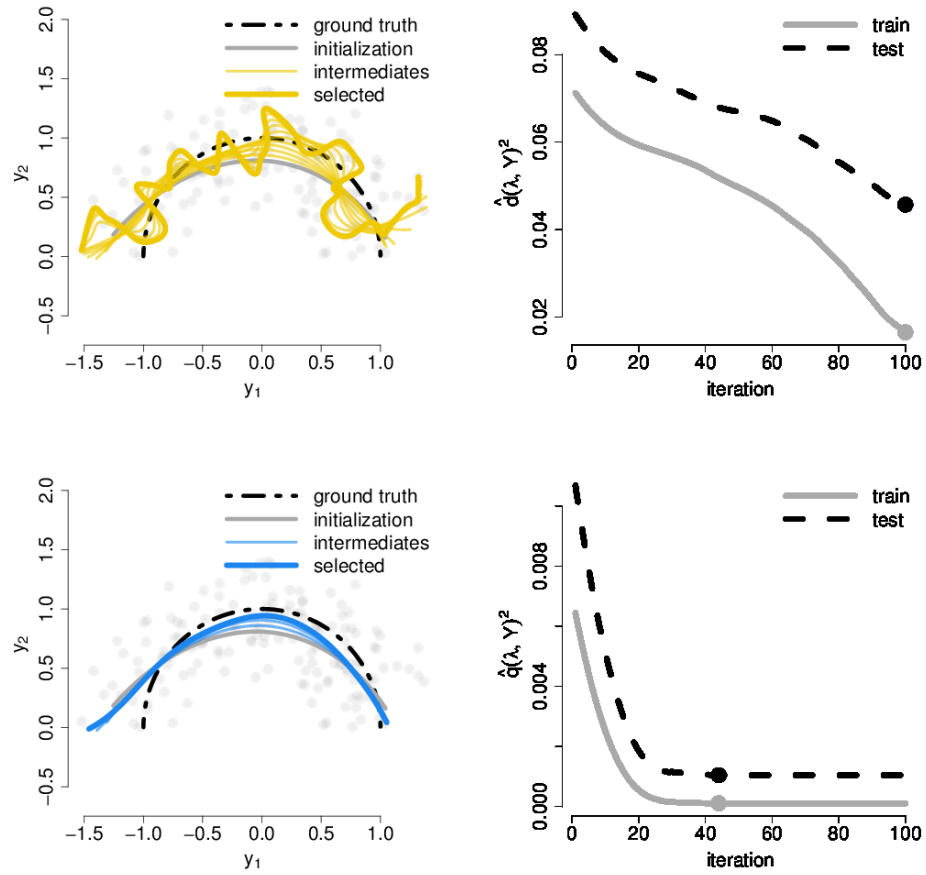



Figure 4: Minimization of (top) $\hat{d}(\lambda, Y)^2$ and (bottom) $\hat{q}(\lambda, Y)^2$ with automatic bandwidth selection starting from $\sigma_g = 1$ and $\sigma_\lambda = 0.1$. (left) Fitted curve with optimization path and (right) train and test error with points indicating minimal train and test error, respectively.

1D case in Section 4.1 holds. To qualitatively inspect the optimization, test data is created using points from a regular grid on $H \times \Phi$, again with normal $\mathcal{N}(0, \frac{1}{2})$ noise orthogonal to the surface added at each grid location.

Figure 7 shows optimization results starting from a surface close to a principal plane. Bandwidths are initialized by the average 75-nearest neighbor distance of $\lambda(Y)$ for $\sigma_g$ and the average 25-nearest neighbor distance of $Y$ for $\sigma_\lambda$. As in the 1D case $\hat{q}(\lambda, Y)^2$ moves towards the principal plane, while $\hat{d}(\lambda, Y)^2$ results in a highly curved fit that begins to fill the space.

### 4.3 Conditional Expectation Manifolds in Higher Dimensions

The final experiment reports results of fitting a 3D manifold on an image data set. The data set consists of 1965 images of a face with different facial expression with varying orientation. The images are $20 \times 28$ pixels with intensities in $[0, 255]$ and treated as points in a 560-dimensional Euclidean space. To test the quality of automatic bandwidth selection normally-distributed ($\mathcal{N}(0, 20)$) noise is added to each pixel. The data set is randomly split into 1000 training and 965 testing images. Then

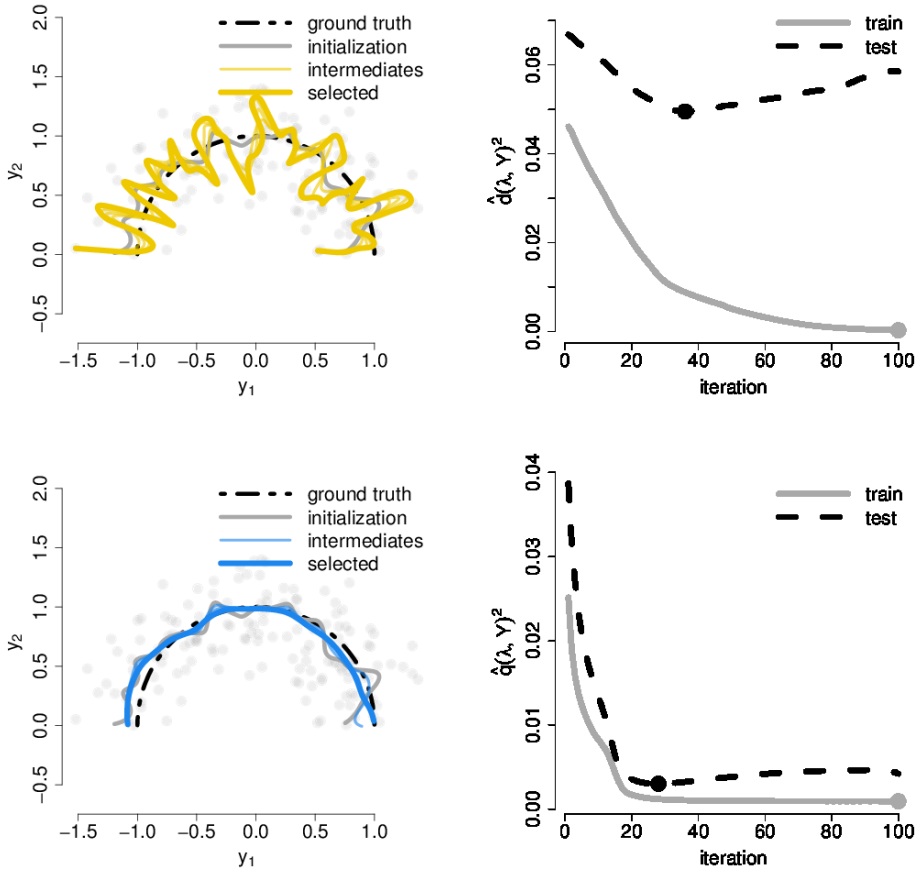
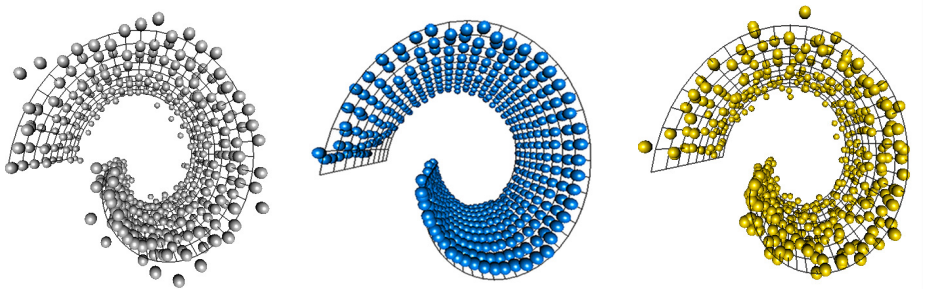



Figure 5: Minimization of (top) $\hat{d}(\lambda, Y)^2$ and (bottom) $\hat{q}(\lambda, Y)^2$ with automatic bandwidth selection starting from $\sigma_g = 0.1$ and $\sigma_\lambda = 0.1$. (left) Fitted curve with optimization path and (right) train and test error with points indicating minimal train and test error, respectively.

$\hat{q}^2(\lambda, Y)$ and $\hat{d}^2(\lambda, Y)$ is optimized on the training data with $Z$ initialized to a 3D isomap embedding. The optimization is stopped based on the objective function value of the test data. Figure 8 shows the original and noisy images and the projection on the conditional expectation manifold for each optimization. While the mean squared projection distance is similar for both solutions, the optimization $\hat{q}^2(\lambda, Y)$ leads to qualitatively much better results. This underscores the observations in the introduction: mean squared error in itself is not a good indicator of model quality for nonparametric manifold estimation.

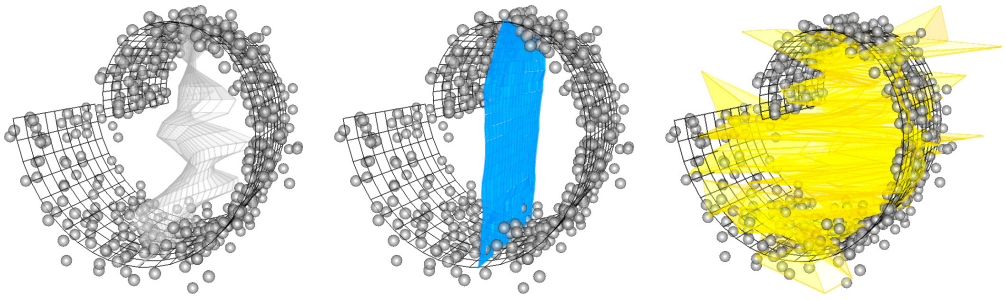
## 5. Conclusion

The conditional expectation curve formulation together with the introduction of a new objective function solves the model complexity problem in principal curve estimation. The resulting algorithm for finite data has the remarkable property that a regularization-free minimum on training data is typically a desired solution. The proposed formulation appears to transform the ill-posed



(a) (b) (c)

Figure 6: Optimization results on the Swissroll (wire frame). Projection $g(\lambda(y))$ of (a) test data after minimization (on a separate training sample shown in Figure 7) of (b) $\hat{q}(\lambda, Y)^2$ and (c) $\hat{d}(\lambda, Y)^2$. A good estimate projects the noisy test data points, located orthogonal to the wire frame, close to the intersections.



(a) (b) (c)

Figure 7: Minimization of (b) $\hat{q}(\lambda, Y)^2$ and (c) $\hat{d}(\lambda, Y)^2$ starting from an (a) initial surface close to a principal plane.

problem of principal curve estimation into a well-posed, or at least stable, problem. A formal investigation of the stability as defined in the framework of Mukherjee et al. (2006) could provide insight towards possible adaptations to other ill-posed problems.

Several important questions regarding the interplay between noise, curvature and sample size remain open. Of particular interest are the convergence rates of $\hat{q}(\lambda, Y)$ to $q(\lambda, Y)$ or error bounds on the corresponding conditional expectation curves. An analysis of the requirements imposed by projection distance, curvature and sample size on $\sigma_\lambda$ and implications on alternative formulations for modelling $\lambda$ is an interesting direction for future research.

## Acknowledgments

The authors thank Thomas Fletcher for the helpful discussions on Lemma 2. This work was supported by the NSF grant CCF-073222.

Figure 8: Optimization results on (1st row) face data set. Test data (2nd row) with $\mathcal{N}(0,20)$ noise added, projected onto the conditional expectation manifold after minimization of (3rd row) $\hat{d}(\lambda,Y)^2$ and (4th row) $\hat{q}(\lambda,Y)^2$ on training data also with $\mathcal{N}(0,20)$ noise added.

## References

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

G. Biau and A. Fischer. Parameter selection for principal curves. *IEEE Transactions On Information Theory*, 58(3):1924 –1939, 2012.

K-Y. Chang and J. Ghosh. A unified model for probabilistic principal surfaces. *IEEE Transaction On Pattern Analysis And Machine Intelligence*, 23(1):22–41, 2001.

W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74(368):829–836, 1979.

T. Duchamp and W. Stuetzle. Extremal properties of principal curves in the plane. *The Annals of Statistics*, 24(4):1511–1520, 1996.

H. Federer. *Geometric Measure Theory*. Springer, NY, 1969.

H. Flanders. Differentiation under the integral sign. *The American Mathematical Monthly*, 80(6): 615–627, 1973.

S. Gerber. *Cems: Conditional Expectation Manifolds*, 2011. URL `http://CRAN.R-project.org/package=cems`. R package version 0.1.

S. Gerber, T. Tasdizen, and R. Whitaker. Dimensionality reduction and principal surfaces via kernel map manifolds. In *IEEE 12th International Conference on Computer Vision*, pages 529–536, 2009.

W. Härdle and J. S. Marron. Optimal bandwidth selection in nonparametric regression function estimation. *Annals of Statistics*, 13(4), 1985.

T. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84 (406):502–516, 1989.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504 – 507, 2006.

B. Kégl, A. Krzyzak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Transaction On Pattern Analysis Machine Intelligence*, 22(3):281–297, 2000.

T. Kohonen. *Self-organized Formation of Topologically Correct Feature Maps*, pages 509–521. MIT Press, Cambridge, MA, USA, 1988.

P. Meinicke, S. Klanke, R. Memisevic, and H. Ritter. Principal surfaces from unsupervised kernel regression. *IEEE Transactions on Pattern Analysis And Machine Intelligence*, 27(9):1379–1391, 2005. ISSN 0162-8828.

S. Mukherjee, P. Niyogi, T. Poggio, and R. Rifkin. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25:161–193, 2006.

E. A. Nadaraya. On estimating regression. *Theory of Probability and its Applications*, 9(1):141–142, 1964.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(550), 2000.

A. J. Smola, S. Mika, B. Schölkopf, and R. C. Williamson. Regularized principal manifolds. *Journal Of Machine Learning Research*, 1:179–209, 2001.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(550):2319–2323, 2000.

R. Tibshirani. Principal curves revisited. *Statistics and Computing*, 2:183–190, 1992.

H. Wang and T. C. M. Lee. Automatic parameter selection for a k-segments algorithm for computing principal curves. *Pattern Recognition Letteerss*, 27(10):1142–1150, 2006.

G. Watson. Smooth regression analysis. *Sankhya, Series*, A(26):359–372, 1964.

Journal of Machine Learning Research 14 (2013) 1303-1347 Submitted 6/12; Published 5/13

# Stochastic Variational Inference

**Matthew D. Hoffman** MATHOFFM@ADOBE.COM
*Adobe Research*
*Adobe Systems Incorporated*
*601 Townsend Street*
*San Francisco, CA 94103, USA*

**David M. Blei** BLEI@CS.PRINCETON.EDU
*Department of Computer Science*
*Princeton University*
*35 Olden Street*
*Princeton, NJ 08540, USA*

**Chong Wang** CHONGW@CS.CMU.EDU
*Machine Learning Department*
*Carnegie Mellon University*
*Gates Hillman Centers, 8110*
*5000 Forbes Avenue*
*Pittsburgh, PA 15213, USA*

**John Paisley** JPAISLEY@BERKELEY.EDU
*Computer Science Division*
*University of California*
*Berkeley, CA 94720-1776, USA*

**Editor:** Tommi Jaakkola

## Abstract

We develop stochastic variational inference, a scalable algorithm for approximating posterior distributions. We develop this technique for a large class of probabilistic models and we demonstrate it with two probabilistic topic models, latent Dirichlet allocation and the hierarchical Dirichlet process topic model. Using stochastic variational inference, we analyze several large collections of documents: 300K articles from *Nature*, 1.8M articles from *The New York Times*, and 3.8M articles from *Wikipedia*. Stochastic inference can easily handle data sets of this size and outperforms traditional variational inference, which can only handle a smaller subset. (We also show that the Bayesian nonparametric topic model outperforms its parametric counterpart.) Stochastic variational inference lets us apply complex Bayesian models to massive data sets.

**Keywords:** Bayesian inference, variational inference, stochastic optimization, topic models, Bayesian nonparametrics

## 1. Introduction

Modern data analysis requires computation with massive data. As examples, consider the following. (1) We have an archive of the raw text of two million books, scanned and stored online. We want to discover the themes in the texts, organize the books by subject, and build a navigator for users

©2013 Matthew D. Hoffman, David M. Blei, Chong Wang and John Paisley.

to explore our collection. (2) We have data from an online shopping website containing millions of users' purchase histories as well as descriptions of each item in the catalog. We want to recommend items to users based on this information. (3) We are continuously collecting data from an online feed of photographs. We want to build a classifier from these data. (4) We have measured the gene sequences of millions of people. We want to make hypotheses about connections between observed genes and other traits.

These problems illustrate some of the challenges to modern data analysis. Our data are complex and high-dimensional; we have assumptions to make—from science, intuition, or other data analyses—that involve structures we believe exist in the data but that we cannot directly observe; and finally our data sets are large, possibly even arriving in a never-ending stream.

Statistical machine learning research has addressed some of these challenges by developing the field of probabilistic modeling, a field that provides an elegant approach to developing new methods for analyzing data (Pearl, 1988; Jordan, 1999; Bishop, 2006; Koller and Friedman, 2009; Murphy, 2012). In particular, *probabilistic graphical models* give us a visual language for expressing assumptions about data and its hidden structure. The corresponding *posterior inference algorithms* let us analyze data under those assumptions, inferring the hidden structure that best explains our observations.

In descriptive tasks, like problems #1 and #4 above, graphical models help us explore the data—the organization of books or the connections between genes and traits—with the hidden structure probabilistically "filled in." In predictive tasks, like problems #2 and #3, we use models to form predictions about new observations. For example, we can make recommendations to users or predict the class labels of new images. With graphical models, we enjoy a powerful suite of probability models to connect and combine; and we have general-purpose computational strategies for connecting models to data and estimating the quantities needed to use them.

The problem we face is scale. Inference algorithms of the 1990s and 2000s used to be considered scalable, but they cannot easily handle the amount of data that we described in the four examples above. This is the problem we address here. We present an approach to computing with graphical models that is appropriate for massive data sets, data that might not fit in memory or even be stored locally. Our method does not require clusters of computers or specialized hardware, though it can be further sped up with these amenities.

As an example of this approach to data analysis, consider topic models. Topic models are probabilistic models of text used to uncover the hidden thematic structure in a collection of documents (Blei, 2012). The main idea in a topic model is that there are a set of topics that describe the collection and each document exhibits those topics with different degrees. As a probabilistic model, the topics and how they relate to the documents are hidden structure and the main computational problem is to infer this hidden structure from an observed collection. Figure 1 illustrates the results of our algorithm on a probabilistic topic model. These are two sets of topics, weighted distributions over the vocabulary, found in 1.8M articles from the *New York Times* and 300,000 articles from *Nature*. Topic models are motivated by applications that require analyzing massive collections of documents like this, but traditional algorithms for topic model inference do not easily scale collections of this size.

Our algorithm builds on variational inference, a method that transforms complex inference problems into high-dimensional optimization problems (Jordan et al., 1999; Wainwright and Jordan, 2008). Traditionally, the optimization is solved with a coordinate ascent algorithm, iterating between re-analyzing every data point in the data set and re-estimating its hidden structure. This

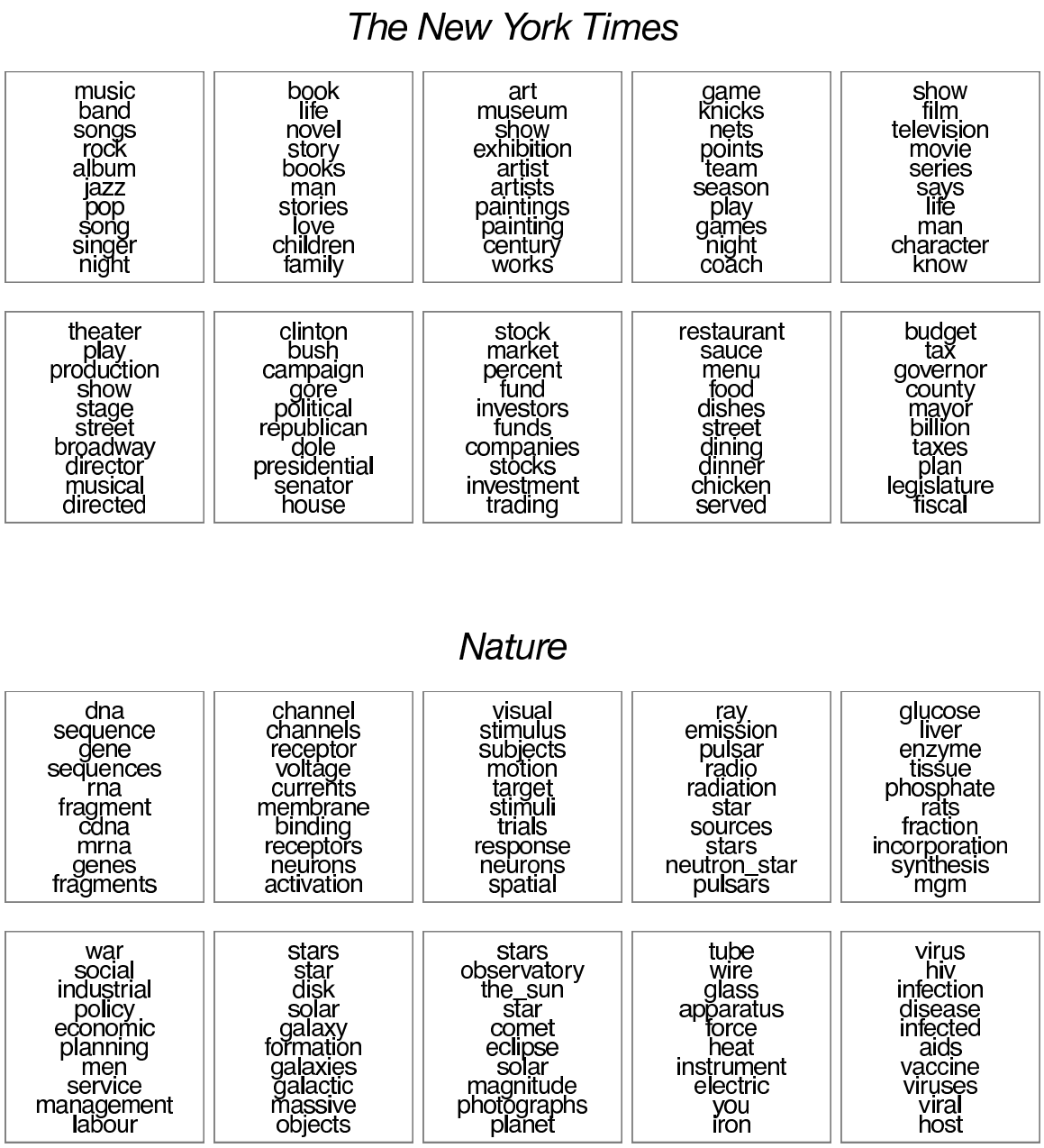



Figure 1: Posterior topics from the hierarchical Dirichlet process topic model on two large data sets. These posteriors were approximated using stochastic variational inference with 1.8M articles from the *New York Times* (top) and 350K articles from *Nature* (bottom). (See Section 3.3 for the modeling details behind the hierarchical Dirichlet process and Section 4 for details about the empirical study.) Each topic is a weighted distribution over the vocabulary and each topic's plot illustrates its most frequent words.

is inefficient for large data sets, however, because it requires a full pass through the data at each iteration.

In this paper we derive a more efficient algorithm by using stochastic optimization (Robbins and Monro, 1951), a technique that follows noisy estimates of the gradient of the objective. When used in variational inference, we show that this gives an algorithm which iterates between subsampling the data and adjusting the hidden structure based only on the subsample. This is much more efficient than traditional variational inference. We call our method *stochastic variational inference*.

We will derive stochastic variational inference for a large class of graphical models. We will study its performance on two kinds of probabilistic topic models. In particular, we demonstrate stochastic variational inference on latent Dirichlet allocation (Blei et al., 2003), a simple topic model, and the hierarchical Dirichlet process topic model (Teh et al., 2006a), a more flexible model where the number of discovered topics grows with the data. (This latter application demonstrates how to use stochastic variational inference in a variety of Bayesian nonparametric settings.) Stochastic variational inference can efficiently analyze massive data sets with complex probabilistic models.

*Technical summary.* We now turn to the technical context of our method. In probabilistic modeling, we use hidden variables to encode hidden structure in observed data; we articulate the relationship between the hidden and observed variables with a factorized probability distribution (i.e., a graphical model); and we use inference algorithms to estimate the posterior distribution, the conditional distribution of the hidden structure given the observations.

Consider a graphical model of hidden and observed random variables for which we want to compute the posterior. For many models of interest, this posterior is not tractable to compute and we must appeal to approximate methods. The two most prominent strategies in statistics and machine learning are Markov chain Monte Carlo (MCMC) sampling and variational inference. In MCMC sampling, we construct a Markov chain over the hidden variables whose stationary distribution is the posterior of interest (Metropolis et al., 1953; Hastings, 1970; Geman and Geman, 1984; Gelfand and Smith, 1990; Robert and Casella, 2004). We run the chain until it has (hopefully) reached equilibrium and collect samples to approximate the posterior. In variational inference, we define a flexible family of distributions over the hidden variables, indexed by free parameters (Jordan et al., 1999; Wainwright and Jordan, 2008). We then find the setting of the parameters (i.e., the member of the family) that is closest to the posterior. Thus we solve the inference problem by solving an optimization problem.

Neither MCMC nor variational inference scales easily to the kinds of settings described in the first paragraph. Researchers have proposed speed-ups of both approaches, but these usually are tailored to specific models or compromise the correctness of the algorithm (or both). Here, we develop a general variational method that scales.

As we mentioned above, the main idea in this work is to use stochastic optimization (Robbins and Monro, 1951; Spall, 2003). In stochastic optimization, we find the maximum of an objective function by following noisy (but unbiased) estimates of its gradient. Under the right conditions, stochastic optimization algorithms provably converge to an optimum of the objective. Stochastic optimization is particularly attractive when the objective (and therefore its gradient) is a sum of many terms that can be computed independently. In that setting, we can cheaply compute noisy gradients by subsampling only a few of these terms.

Variational inference is amenable to stochastic optimization because the variational objective decomposes into a sum of terms, one for each data point in the analysis. We can cheaply obtain noisy estimates of the gradient by subsampling the data and computing a scaled gradient on the

subsample. If we sample independently then the expectation of this noisy gradient is equal to the true gradient. With one more detail—the idea of a natural gradient (Amari, 1998)—stochastic variational inference has an attractive form:

1. Subsample one or more data points from the data.
2. Analyze the subsample using the current variational parameters.
3. Implement a closed-form update of the variational parameters.
4. Repeat.

While traditional algorithms require repeatedly analyzing the whole data set before updating the variational parameters, this algorithm only requires that we analyze randomly sampled subsets. We will show how to use this algorithm for a large class of graphical models.

*Related work.* Variational inference for probabilistic models was pioneered in the mid-1990s. In Michael Jordan's lab, the seminal papers of Saul et al. (1996); Saul and Jordan (1996) and Jaakkola (1997) grew out of reading the statistical physics literature (Peterson and Anderson, 1987; Parisi, 1988). In parallel, the mean-field methods explained in Neal and Hinton (1999) (originally published in 1993) and Hinton and Van Camp (1993) led to variational algorithms for mixtures of experts (Waterhouse et al., 1996).

In subsequent years, researchers began to understand the potential for variational inference in more general settings and developed generic algorithms for conjugate exponential-family models (Attias, 1999, 2000; Wiegerinck, 2000; Ghahramani and Beal, 2001; Xing et al., 2003). These innovations led to automated variational inference, allowing a practitioner to write down a model and immediately use variational inference to estimate its posterior (Bishop et al., 2003). For good reviews of variational inference see Jordan et al. (1999) and Wainwright and Jordan (2008).

In this paper, we develop scalable methods for generic Bayesian inference by solving the variational inference problem with stochastic optimization (Robbins and Monro, 1951). Our algorithm builds on the earlier approach of Sato (2001), whose algorithm only applies to the limited set of models that can be fit with the EM algorithm (Dempster et al., 1977). Specifically, we generalize his approach to the much wider set of probabilistic models that are amenable to closed-form coordinate ascent inference. Further, in the sense that EM itself is a mean-field method (Neal and Hinton, 1999), our algorithm builds on the stochastic optimization approach to EM (Cappé and Moulines, 2009). Finally, we note that stochastic optimization was also used with variational inference in Platt et al. (2008) for fast approximate inference in a specific model of web service activity.

For approximate inference, the main alternative to variational methods is Markov chain Monte Carlo (MCMC) (Robert and Casella, 2004). Despite its popularity in Bayesian inference, relatively little work has focused on developing MCMC algorithms that can scale to very large data sets. One exception is sequential Monte Carlo, although these typically lack strong convergence guarantees (Doucet et al., 2001). Another is the stochastic gradient Langevin method of Welling and Teh (2011), which enjoys asymptotic convergence guarantees and also takes advantage of stochastic optimization. Finally, in topic modeling, researchers have developed several approaches to parallel MCMC (Newman et al., 2009; Smola and Narayanamurthy, 2010; Ahmed et al., 2012).

*The organization of this paper.* In Section 2, we review variational inference for graphical models and then derive stochastic variational inference. In Section 3, we review probabilistic topic models and Bayesian nonparametric models and then derive the stochastic variational inference algorithms in these settings. In Section 4, we study stochastic variational inference on several large text data sets.

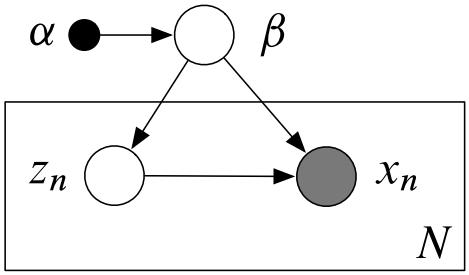



Figure 2: A graphical model with observations $x_{1:N}$, local hidden variables $z_{1:N}$ and global hidden variables $\beta$. The distribution of each observation $x_n$ only depends on its corresponding local variable $z_n$ and the global variables $\beta$. (Though not pictured, each hidden variable $z_n$, observation $x_n$, and global variable $\beta$ may be a collection of multiple random variables.)

## 2. Stochastic Variational Inference

We derive *stochastic variational inference*, a stochastic optimization algorithm for mean-field variational inference. Our algorithm approximates the posterior distribution of a probabilistic model with hidden variables, and can handle massive data sets of observations.

We divide this section into four parts.

1. We define the class of models to which our algorithm applies. We define *local* and *global* hidden variables, and requirements on the conditional distributions within the model.

2. We review *mean-field variational inference*, an approximate inference strategy that seeks a tractable distribution over the hidden variables which is close to the posterior distribution. We derive the traditional variational inference algorithm for our class of models, which is a coordinate ascent algorithm.

3. We review the *natural gradient* and derive the natural gradient of the variational objective function. The natural gradient closely relates to coordinate ascent variational inference.

4. We review stochastic optimization, a technique that uses noisy estimates of a gradient to optimize an objective function, and apply it to variational inference. Specifically, we use stochastic optimization with noisy estimates of the natural gradient of the variational objective. These estimates arise from repeatedly subsampling the data set. We show how the resulting algorithm, *stochastic variational inference*, easily builds on traditional variational inference algorithms but can handle much larger data sets.

### 2.1 Models with Local and Global Hidden Variables

Our class of models involves observations, global hidden variables, local hidden variables, and fixed parameters. The $N$ observations are $x = x_{1:N}$; the vector of global hidden variables is $\beta$; the $N$ local hidden variables are $z = z_{1:N}$, each of which is a collection of $J$ variables $z_n = z_{n,1:J}$; the vector of fixed parameters is $\alpha$. (Note we can easily allow $\alpha$ to partly govern any of the random variables,

such as fixed parts of the conditional distribution of observations. To keep notation simple, we assume that they only govern the global hidden variables.)

The joint distribution factorizes into a global term and a product of local terms,

$$p(x,z,\beta\,|\,\alpha) = p(\beta\,|\,\alpha)\prod_{n=1}^{N} p(x_n,z_n\,|\,\beta). \tag{1}$$

Figure 2 illustrates the graphical model. Our goal is to approximate the posterior distribution of the hidden variables given the observations, $p(\beta,z\,|\,x)$.

The distinction between local and global hidden variables is determined by the conditional dependencies. In particular, the $n$th observation $x_n$ and the $n$th local variable $z_n$ are conditionally independent, given global variables $\beta$, of all other observations and local hidden variables,

$$p(x_n,z_n\,|\,x_{-n},z_{-n},\beta,\alpha) = p(x_n,z_n\,|\,\beta,\alpha).$$

The notation $x_{-n}$ and $z_{-n}$ refers to the set of variables except the $n$th.

This kind of model frequently arises in Bayesian statistics. The global variables $\beta$ are parameters endowed with a prior $p(\beta)$ and each local variable $z_n$ contains the hidden structure that governs the $n$th observation. For example, consider a Bayesian mixture of Gaussians. The global variables are the mixture proportions and the means and variances of the mixture components; the local variable $z_n$ is the hidden cluster label for the $n$th observation $x_n$.

We have described the independence assumptions of the hidden variables. We make further assumptions about the *complete conditionals* in the model. A complete conditional is the conditional distribution of a hidden variable given the other hidden variables and the observations. We assume that these distributions are in the exponential family,

$$p(\beta\,|\,x,z,\alpha) = h(\beta)\exp\{\eta_g(x,z,\alpha)^\top t(\beta) - a_g(\eta_g(x,z,\alpha))\}, \tag{2}$$

$$p(z_{nj}\,|\,x_n,z_{n,-j},\beta) = h(z_{nj})\exp\{\eta_\ell(x_n,z_{n,-j},\beta)^\top t(z_{nj}) - a_\ell(\eta_\ell(x_n,z_{n,-j},\beta))\}. \tag{3}$$

The scalar functions $h(\cdot)$ and $a(\cdot)$ are respectively the *base measure* and *log-normalizer*; the vector functions $\eta(\cdot)$ and $t(\cdot)$ are respectively the *natural parameter* and *sufficient statistics*.[1] These are conditional distributions, so the natural parameter is a function of the variables that are being conditioned on. (The subscripts on the natural parameter $\eta$ indicate complete conditionals for local or global variables.) For the local variables $z_{nj}$, the complete conditional distribution is determined by the global variables $\beta$ and the other local variables in the $n$th context, that is, the $n$th data point $x_n$ and the local variables $z_{n,-j}$. This follows from the factorization in Equation 1.

These assumptions on the complete conditionals imply a conjugacy relationship between the global variables $\beta$ and the local contexts $(z_n,x_n)$, and this relationship implies a specific form of the complete conditional for $\beta$. Specifically, the distribution of the local context given the global variables must be in an exponential family,

$$p(x_n,z_n|\beta) = h(x_n,z_n)\exp\{\beta^\top t(x_n,z_n) - a_\ell(\beta)\}. \tag{4}$$

1. We use overloaded notation for the functions $h(\cdot)$ and $t(\cdot)$ so that they depend on the names of their arguments; for example, $h(z_{nj})$ can be thought of as a shorthand for the more formal (but more cluttered) notation $h_{z_{nj}}(z_{nj})$. This is analogous to the standard convention of overloading the probability function $p(\cdot)$.

The prior distribution $p(\beta)$ must also be in an exponential family,

$$p(\beta) = h(\beta)\exp\{\alpha^\top t(\beta) - a_g(\alpha)\}. \tag{5}$$

The sufficient statistics are $t(\beta) = (\beta, -a_\ell(\beta))$ and thus the hyperparameter $\alpha$ has two components $\alpha = (\alpha_1, \alpha_2)$. The first component $\alpha_1$ is a vector of the same dimension as $\beta$; the second component $\alpha_2$ is a scalar.

Equations 4 and 5 imply that the complete conditional for the global variable in Equation 2 is in the same exponential family as the prior with natural parameter

$$\eta_g(x, z, \alpha) = (\alpha_1 + \textstyle\sum_{n=1}^N t(z_n, x_n), \alpha_2 + N). \tag{6}$$

This form will be important when we derive stochastic variational inference in Section 2.4. See Bernardo and Smith (1994) for a general discussion of conjugacy and the exponential family.

This family of distributions—those with local and global variables, and where the complete conditionals are in the exponential family—contains many useful statistical models from the machine learning and statistics literature. Examples include Bayesian mixture models (Ghahramani and Beal, 2000; Attias, 2000), latent Dirichlet allocation (Blei et al., 2003), hidden Markov models (and many variants) (Rabiner, 1989; Fine et al., 1998; Fox et al., 2011b; Paisley and Carin, 2009), Kalman filters (and many variants) (Kalman, 1960; Fox et al., 2011a), factorial models (Ghahramani and Jordan, 1997), hierarchical linear regression models (Gelman and Hill, 2007), hierarchical probit classification models (McCullagh and Nelder, 1989; Girolami and Rogers, 2006), probabilistic factor analysis/matrix factorization models (Spearman, 1904; Tipping and Bishop, 1999; Collins et al., 2002; Wang, 2006; Salakhutdinov and Mnih, 2008; Paisley and Carin, 2009; Hoffman et al., 2010b), certain Bayesian nonparametric mixture models (Antoniak, 1974; Escobar and West, 1995; Teh et al., 2006a), and others.[2]

Analyzing data with one of these models amounts to computing the posterior distribution of the hidden variables given the observations,

$$p(z, \beta \,|\, x) = \frac{p(x, z, \beta)}{\int p(x, z, \beta) dz d\beta}. \tag{7}$$

We then use this posterior to explore the hidden structure of our data or to make predictions about future data. For many models however, such as the examples listed above, the denominator in Equation 7 is intractable to compute. Thus we resort to approximate posterior inference, a problem that has been a focus of modern Bayesian statistics. We now turn to mean-field variational inference, the approximation inference technique which roots our strategy for scalable inference.

### 2.2 Mean-Field Variational Inference

Variational inference casts the inference problem as an optimization. We introduce a family of distributions over the hidden variables that is indexed by a set of free parameters, and then optimize those parameters to find the member of the family that is closest to the posterior of interest. (Closeness is measured with Kullback-Leibler divergence.) We use the resulting distribution, called the *variational distribution*, to approximate the posterior.

2. We note that our assumptions can be relaxed to the case where the full conditional $p(\beta|x,z)$ is not tractable, but each partial conditional $p(\beta_k|x,z,\beta_{-k})$ associated with the global variable $\beta_k$ is in a tractable exponential family. The topic models of the next section do not require this complexity, so we chose to keep the derivation a little simpler.

In this section we review mean-field variational inference, the form of variational inference that uses a family where each hidden variable is independent. We describe the variational objective function, discuss the mean-field variational family, and derive the traditional coordinate ascent algorithm for fitting the variational parameters. This algorithm is a stepping stone to stochastic variational inference.

*The evidence lower bound.* Variational inference minimizes the Kullback-Leibler (KL) divergence from the variational distribution to the posterior distribution. It maximizes the *evidence lower bound* (ELBO), a lower bound on the logarithm of the marginal probability of the observations $\log p(x)$. The ELBO is equal to the negative KL divergence up to an additive constant.

We derive the ELBO by introducing a distribution over the hidden variables $q(z,\beta)$ and using Jensen's inequality. (Jensen's inequality and the concavity of the logarithm function imply that $\log \mathbb{E}[f(y)] \geq \mathbb{E}[\log f(y)]$ for any random variable $y$.) This gives the following bound on the log marginal,

$$\begin{aligned}
\log p(x) &= \log \int p(x,z,\beta)\,dz\,d\beta \\
&= \log \int p(x,z,\beta)\frac{q(z,\beta)}{q(z,\beta)}\,dz\,d\beta \\
&= \log\left(\mathbb{E}_q\left[\frac{p(x,z,\beta)}{q(z,\beta)}\right]\right) \\
&\geq \mathbb{E}_q[\log p(x,z,\beta)] - \mathbb{E}_q[\log q(z,\beta)] \qquad (8)\\
&\triangleq \mathcal{L}(q).
\end{aligned}$$

The ELBO contains two terms. The first term is the expected log joint, $\mathbb{E}_q[\log p(x,z,\beta)]$. The second term is the entropy of the variational distribution, $-\mathbb{E}_q[\log q(z,\beta)]$. Both of these terms depend on $q(z,\beta)$, the variational distribution of the hidden variables.

We restrict $q(z,\beta)$ to be in a family that is tractable, one for which the expectations in the ELBO can be efficiently computed. We then try to find the member of the family that maximizes the ELBO. Finally, we use the optimized distribution as a proxy for the posterior.

Solving this maximization problem is equivalent to finding the member of the family that is closest in KL divergence to the posterior (Jordan et al., 1999; Wainwright and Jordan, 2008),

$$\begin{aligned}
\mathrm{KL}(q(z,\beta)||p(z,\beta|x)) &= \mathbb{E}_q\left[\log q(z,\beta)\right] - \mathbb{E}_q\left[\log p(z,\beta\,|\,x)\right] \\
&= \mathbb{E}_q\left[\log q(z,\beta)\right] - \mathbb{E}_q\left[\log p(x,z,\beta)\right] + \log p(x) \\
&= -\mathcal{L}(q) + \text{const.}
\end{aligned}$$

$\log p(x)$ is replaced by a constant because it does not depend on $q$.

*The mean-field variational family.* The simplest variational family of distributions is the *mean-field family*. In this family, each hidden variable is independent and governed by its own parameter,

$$q(z,\beta) = q(\beta\,|\,\lambda)\prod_{n=1}^{N}\prod_{j=1}^{J} q(z_{nj}\,|\,\phi_{nj}). \qquad (9)$$

The global parameters $\lambda$ govern the global variables; the local parameters $\phi_n$ govern the local variables in the $n$th context. The ELBO is a function of these parameters.

Equation 9 gives the factorization of the variational family, but does not specify its form. We set $q(\beta|\lambda)$ and $q(z_{nj}|\phi_{nj})$ to be in the same exponential family as the complete conditional distributions $p(\beta|x,z)$ and $p(z_{nj}|x_n, z_{n,-j}, \beta)$, from Equations 2 and 3. The variational parameters $\lambda$ and $\phi_{nj}$ are the natural parameters to those families,

$$q(\beta \mid \lambda) = h(\beta)\exp\{\lambda^\top t(\beta) - a_g(\lambda)\}, \tag{10}$$

$$q(z_{nj} \mid \phi_{nj}) = h(z_{nj})\exp\{\phi_{nj}^\top t(z_{nj}) - a_\ell(\phi_{nj})\}. \tag{11}$$

These forms of the variational distributions lead to an easy coordinate ascent algorithm. Further, the optimal mean-field distribution, without regard to its particular functional form, has factors in these families (Bishop, 2006).

Note that assuming that these exponential families are the same as their corresponding conditionals means that $t(\cdot)$ and $h(\cdot)$ in Equation 10 are the same functions as $t(\cdot)$ and $h(\cdot)$ in Equation 2. Likewise, $t(\cdot)$ and $h(\cdot)$ in Equation 11 are the same as in Equation 3. We will sometimes suppress the explicit dependence on $\phi$ and $\lambda$, substituting $q(z_{nj})$ for $q(z_{nj}|\phi_{nj})$ and $q(\beta)$ for $q(\beta|\lambda)$.

The mean-field family has several computational advantages. For one, the entropy term decomposes,

$$-\mathbb{E}_q[\log q(z,\beta)] = -\mathbb{E}_\lambda[\log q(\beta)] - \sum_{n=1}^{N}\sum_{j=1}^{J}\mathbb{E}_{\phi_{nj}}[\log q(z_{nj})],$$

where $\mathbb{E}_{\phi_{nj}}[\cdot]$ denotes an expectation with respect to $q(z_{nj} \mid \phi_{nj})$ and $\mathbb{E}_\lambda[\cdot]$ denotes an expectation with respect to $q(\beta \mid \lambda)$. Its other computational advantages will emerge as we derive the gradients of the variational objective and the coordinate ascent algorithm.

*The gradient of the ELBO and coordinate ascent inference.* We have defined the objective function in Equation 8 and the variational family in Equations 9, 10 and 11. Our goal is to optimize the objective with respect to the variational parameters.

In traditional mean-field variational inference, we optimize Equation 8 with coordinate ascent. We iteratively optimize each variational parameter, holding the other parameters fixed. With the assumptions that we have made about the model and variational distribution—that each conditional is in an exponential family and that the corresponding variational distribution is in the same exponential family—we can optimize each coordinate in closed form.

We first derive the coordinate update for the parameter $\lambda$ to the variational distribution of the global variables $q(\beta \mid \lambda)$. As a function of $\lambda$, we can rewrite the objective as

$$\mathcal{L}(\lambda) = \mathbb{E}_q[\log p(\beta \mid x,z)] - \mathbb{E}_q[\log q(\beta)] + \text{const.} \tag{12}$$

The first two terms are expectations that involve $\beta$; the third term is constant with respect to $\lambda$. The constant absorbs quantities that depend only on the other hidden variables. Those quantities do not depend on $q(\beta \mid \lambda)$ because all variables are independent in the mean-field family.

Equation 12 reproduces the full ELBO in Equation 8. The second term of Equation 12 is the entropy of the global variational distribution. The first term derives from the expected log joint likelihood, where we use the chain rule to separate terms that depend on the variable $\beta$ from terms that do not,

$$\mathbb{E}_q[\log p(x,z,\beta)] = \mathbb{E}_q[\log p(x,z)] + \mathbb{E}_q[\log p(\beta \mid x,z)].$$

The constant absorbs $\mathbb{E}_q[\log p(x,z)]$, leaving the expected log conditional $\mathbb{E}_q[\log p(\beta \mid x,z)]$.

Finally, we substitute the form of $q(\beta \mid \lambda)$ in Equation 10 to obtain the final expression for the ELBO as a function of $\lambda$,

$$\mathcal{L}(\lambda) = \mathbb{E}_q[\eta_g(x,z,\alpha)]^\top \nabla_\lambda a_g(\lambda) - \lambda^\top \nabla_\lambda a_g(\lambda) + a_g(\lambda) + \text{const.} \tag{13}$$

In the first and second terms on the right side, we used the exponential family identity that the expectation of the sufficient statistics is the gradient of the log normalizer, $\mathbb{E}_q[t(\beta)] = \nabla_\lambda a_g(\lambda)$. The constant has further absorbed the expected log normalizer of the conditional distribution $-\mathbb{E}_q[a_g(\eta_g(x,z,\alpha))]$, which does not depend on $q(\beta)$.

Equation 13 simplifies the ELBO as a function of the global variational parameter. To derive the coordinate ascent update, we take the gradient,

$$\nabla_\lambda \mathcal{L} = \nabla^2_\lambda a_g(\lambda)(\mathbb{E}_q[\eta_g(x,z,\alpha)] - \lambda). \tag{14}$$

We can set this gradient to zero by setting

$$\lambda = \mathbb{E}_q[\eta_g(x,z,\alpha)]. \tag{15}$$

This sets the global variational parameter equal to the expected natural parameter of its complete conditional distribution. Implementing this update, holding all other variational parameters fixed, optimizes the ELBO over $\lambda$. Notice that the mean-field assumption plays an important role. The update is the expected conditional parameter $\mathbb{E}_q[\eta_g(x,z,\alpha)]$, which is an expectation of a function of the other random variables and observations. Thanks to the mean-field assumption, this expectation is only a function of the local variational parameters and does not depend on $\lambda$.

We now turn to the local parameters $\phi_{nj}$. The gradient is nearly identical to the global case,

$$\nabla_{\phi_{nj}} \mathcal{L} = \nabla^2_{\phi_{nj}} a_\ell(\phi_{nj})(\mathbb{E}_q[\eta_\ell(x_n, z_{n,-j}, \beta)] - \phi_{nj}).$$

It equals zero when

$$\phi_{nj} = \mathbb{E}_q[\eta_\ell(x_n, z_{n,-j}, \beta)]. \tag{16}$$

Mirroring the global update, this expectation does not depend on $\phi_{nj}$. However, while the global update in Equation 15 depends on all the local variational parameters—and note there is a set of local parameters for each of the $N$ observations—the local update in Equation 16 only depends on the global parameters and the other parameters associated with the $n$th context. The computational difference between local and global updates will be important in the scalable algorithm of Section 2.4.

The updates in Equations 15 and 16 form the algorithm for coordinate ascent variational inference, iterating between updating each local parameter and the global parameters. The full algorithm is in Figure 3, which is guaranteed to find a local optimum of the ELBO. Computing the expectations at each step is easy for directed graphical models with tractable complete conditionals, and in Section 3 we show that these updates are tractable for many topic models. Figure 3 is the "classical" variational inference algorithm, used in many settings.

As an aside, these updates reveal a connection between mean-field variational inference and Gibbs sampling (Gelfand and Smith, 1990). In Gibbs sampling, we iteratively sample from each complete conditional. In variational inference, we take variational expectations of the natural parameters of the same distributions. The updates also show a connection to the expectation-maximization

1: Initialize $\lambda^{(0)}$ randomly.
2: **repeat**
3: **for** each local variational parameter $\phi_{nj}$ **do**
4: Update $\phi_{nj}$, $\phi_{nj}^{(t)} = \mathbb{E}_{q^{(t-1)}}[\eta_{\ell,j}(x_n, z_{n,-j}, \beta)]$.
5: **end for**
6: Update the global variational parameters, $\lambda^{(t)} = \mathbb{E}_{q^{(t)}}[\eta_g(z_{1:N}, x_{1:N})]$.
7: **until** the ELBO converges

Figure 3: Coordinate ascent mean-field variational inference.

(EM) algorithm (Dempster et al., 1977)—Equation 16 corresponds to the E step, and Equation 15 corresponds to the M step (Neal and Hinton, 1999).

We mentioned that the local steps (Steps 3 and 4 in Figure 3) only require computation with the global parameters and the $n$th local context. Thus, the data can be distributed across many machines and the local variational updates can be implemented in parallel. These results can then be aggregated in Step 6 to find the new global variational parameters.

However, the local steps also reveal an inefficiency in the algorithm. The algorithm begins by initializing the global parameters $\lambda$ randomly—the initial value of $\lambda$ does not reflect any regularity in the data. But before completing even one iteration, the algorithm must analyze every data point using these initial (random) values. This is wasteful, especially if we expect that we can learn something about the global variational parameters from only a subset of the data.

We solve this problem with stochastic optimization. This leads to stochastic variational inference, an efficient algorithm that continually improves its estimate of the global parameters as it analyzes more observations. Though the derivation requires some details, we have now described all of the computational components of the algorithm. (See Figure 4.) At each iteration, we sample a data point from the data set and compute its optimal local variational parameters; we form *intermediate global parameters* using classical coordinate ascent updates where the sampled data point is repeated $N$ times; finally, we set the new global parameters to a weighted average of the old estimate and the intermediate parameters.

The algorithm is efficient because it need not analyze the whole data set before improving the global variational parameters, and the per-iteration steps only require computation about a single local context. Furthermore, it only uses calculations from classical coordinate inference. Any existing implementation of variational inference can be easily configured to this scalable alternative.

We now show how stochastic inference arises by applying stochastic optimization to the natural gradients of the variational objective. We first discuss natural gradients and their relationship to the coordinate updates in mean-field variational inference.

### 2.3 The Natural Gradient of the ELBO

The natural gradient of a function accounts for the information geometry of its parameter space, using a Riemannian metric to adjust the direction of the traditional gradient. Amari (1998) discusses natural gradients for maximum-likelihood estimation, which give faster convergence than standard

gradients. In this section we describe Riemannian metrics for probability distributions and the natural gradient of the ELBO.

*Gradients and probability distributions.* The classical gradient method for maximization tries to find a maximum of a function $f(\lambda)$ by taking steps of size $\rho$ in the direction of the gradient,

$$\lambda^{(t+1)} = \lambda^{(t)} + \rho \nabla_\lambda f(\lambda^{(t)}).$$

The gradient (when it exists) points in the direction of steepest ascent. That is, the gradient $\nabla_\lambda f(\lambda)$ points in the same direction as the solution to

$$\arg\max_{d\lambda} f(\lambda + d\lambda) \quad \text{subject to } ||d\lambda||^2 < \varepsilon^2 \tag{17}$$

for sufficiently small $\varepsilon$. Equation 17 implies that if we could only move a tiny distance $\varepsilon$ away from $\lambda$ then we should move in the direction of the gradient. Initially this seems reasonable, but there is a complication. The gradient direction implicitly depends on the Euclidean distance metric associated with the space in which $\lambda$ lives. However, the Euclidean metric might not capture a meaningful notion of distance between settings of $\lambda$.

The problem with Euclidean distance is especially clear in our setting, where we are trying to optimize an objective with respect to a parameterized probability distribution $q(\beta \mid \lambda)$. When optimizing over a probability distribution, the Euclidean distance between two parameter vectors $\lambda$ and $\lambda'$ is often a poor measure of the dissimilarity of the distributions $q(\beta \mid \lambda)$ and $q(\beta \mid \lambda')$. For example, suppose $q(\beta)$ is a univariate normal and $\lambda$ is the mean $\mu$ and scale $\sigma$. The distributions $\mathcal{N}(0, 10000)$ and $\mathcal{N}(10, 10000)$ are almost indistinguishable, and the Euclidean distance between their parameter vectors is 10. In contrast, the distributions $\mathcal{N}(0, 0.01)$ and $\mathcal{N}(0.1, 0.01)$ barely overlap, but this is not reflected in the Euclidean distance between their parameter vectors, which is only 0.1. The *natural gradient* corrects for this issue by redefining the basic definition of the gradient (Amari, 1998).

*Natural gradients and probability distributions.* A natural measure of dissimilarity between probability distributions is the symmetrized KL divergence

$$D_{KL}^{\text{sym}}(\lambda, \lambda') = \mathbb{E}_\lambda \left[ \log \tfrac{q(\beta \mid \lambda)}{q(\beta \mid \lambda')} \right] + \mathbb{E}_{\lambda'} \left[ \log \tfrac{q(\beta \mid \lambda')}{q(\beta \mid \lambda)} \right]. \tag{18}$$

Symmetrized KL depends on the distributions themselves, rather than on how they are parameterized; it is invariant to parameter transformations.

With distances defined using symmetrized KL, we find the direction of steepest ascent in the same way as for gradient methods,

$$\arg\max_{d\lambda} f(\lambda + d\lambda) \quad \text{subject to } D_{KL}^{\text{sym}}(\lambda, \lambda + d\lambda) < \varepsilon. \tag{19}$$

As $\varepsilon \to 0$, the solution to this problem points in the same direction as the *natural gradient*. While the Euclidean gradient points in the direction of steepest ascent in Euclidean space, the natural gradient points in the direction of steepest ascent in the Riemannian space, that is, the space where local distance is defined by KL divergence rather than the $L^2$ norm.

We manage the more complicated constraint in Equation 19 with a Riemannian metric $G(\lambda)$ (Do Carmo, 1992). This metric defines linear transformations of $\lambda$ under which the squared Euclidean distance between $\lambda$ and a nearby vector $\lambda + d\lambda$ is the KL between $q(\beta|\lambda)$ and $q(\beta|\lambda + d\lambda)$,

$$d\lambda^T G(\lambda) d\lambda = D_{KL}^{\text{sym}}(\lambda, \lambda + d\lambda), \tag{20}$$

and note that the transformation can be a function of $\lambda$. Amari (1998) showed that we can compute the natural gradient by premultiplying the gradient by the inverse of the Riemannian metric $G(\lambda)^{-1}$,

$$\hat{\nabla}_\lambda f(\lambda) \triangleq G(\lambda)^{-1} \nabla_\lambda f(\lambda),$$

where $G$ is the Fisher information matrix of $q(\lambda)$ (Amari, 1982; Kullback and Leibler, 1951),

$$G(\lambda) = \mathbb{E}_\lambda \left[ (\nabla_\lambda \log q(\beta \,|\, \lambda))(\nabla_\lambda \log q(\beta \,|\, \lambda))^\top \right]. \tag{21}$$

We can show that Equation 21 satisfies Equation 20 by approximating $\log q(\beta \,|\, \lambda + d\lambda)$ using the first-order Taylor approximations about $\lambda$

$$\begin{aligned} \log q(\beta|\lambda + d\lambda) &= O(d\lambda^2) + \log q(\beta|\lambda) + d\lambda^\top \nabla_\lambda \log q(\beta|\lambda), \\ q(\beta|\lambda + d\lambda) &= O(d\lambda^2) + q(\beta|\lambda) + q(\beta|\lambda) d\lambda^\top \nabla_\lambda \log q(\beta|\lambda), \end{aligned}$$

and plugging the result into Equation 18:

$$\begin{aligned} D_{KL}^{\text{sym}}(\lambda, \lambda + d\lambda) &= \int_\beta (q(\beta|\lambda + d\lambda) - q(\beta|\lambda))(\log q(\beta|\lambda + d\lambda) - \log q(\beta|\lambda)) d\beta \\ &= O(d\lambda^3) + \int_\beta q(\beta|\lambda)(d\lambda^\top \nabla_\lambda \log q(\beta|\lambda))^2 d\beta \\ &= O(d\lambda^3) + \mathbb{E}_q[(d\lambda^\top \nabla_\lambda \log q(\beta|\lambda))^2] = O(d\lambda^3) + d\lambda^\top G(\lambda) d\lambda. \end{aligned}$$

For small enough $d\lambda$ we can ignore the $O(d\lambda^3)$ term.

When $q(\beta \,|\, \lambda)$ is in the exponential family (Equation 10) the metric is the second derivative of the log normalizer,

$$\begin{aligned} G(\lambda) &= \mathbb{E}_\lambda \left[ (\nabla_\lambda \log p(\beta \,|\, \lambda))(\nabla_\lambda \log p(\beta \,|\, \lambda))^\top \right] \\ &= \mathbb{E}_\lambda \left[ (t(\beta) - \mathbb{E}_\lambda[t(\beta)])(t(\beta) - \mathbb{E}_\lambda[t(\beta)])^\top \right] \\ &= \nabla_\lambda^2 a_g(\lambda). \end{aligned}$$

This follows from the exponential family identity that the Hessian of the log normalizer function $a$ with respect to the natural parameter $\lambda$ is the covariance matrix of the sufficient statistic vector $t(\beta)$.

*Natural gradients and mean field variational inference.* We now return to variational inference and compute the natural gradient of the ELBO with respect to the variational parameters. Researchers have used the natural gradient in variational inference for nonlinear state space models (Honkela et al., 2008) and Bayesian mixtures (Sato, 2001).[3]

Consider the global variational parameter $\lambda$. The gradient of the ELBO with respect to $\lambda$ is in Equation 14. Since $\lambda$ is a natural parameter to an exponential family distribution, the Fisher metric defined by $q(\beta)$ is $\nabla_\lambda^2 a_g(\lambda)$. Note that the Fisher metric is the first term in Equation 14. We premultiply the gradient by the inverse Fisher information to find the natural gradient. This reveals that the natural gradient has the following simple form,

$$\hat{\nabla}_\lambda \mathcal{L} = \mathbb{E}_\phi[\eta_g(x, z, \alpha)] - \lambda. \tag{22}$$

3. Our work here—using the natural gradient in a stochastic optimization algorithm—is closest to that of Sato (2001), though we develop the algorithm via a different path and Sato does not address models for which the joint conditional $p(z_n|\beta, x_n)$ is not tractable.

An analogous computation goes through for the local variational parameters,

$$\hat{\nabla}_{\phi_{nj}} \mathcal{L} = \mathbb{E}_{\lambda,\phi_{n,-j}}[\eta_\ell(x_n, z_{n,-j}, \beta)] - \phi_{nj}.$$

The natural gradients are closely related to the coordinate ascent updates of Equation 15 or Equation 16. Consider a full set of variational parameters $\lambda$ and $\phi$. We can compute the natural gradient by computing the coordinate updates in parallel and subtracting the current setting of the parameters. The classical coordinate ascent algorithm can thus be interpreted as a projected natural gradient algorithm (Sato, 2001). Updating a parameter by taking a natural gradient step of length one is equivalent to performing a coordinate update.

We motivated natural gradients by mathematical reasoning around the geometry of the parameter space. More importantly, however, natural gradients are easier to compute than classical gradients. They are easier to compute because premultiplying by the Fisher information matrix—which we must do to compute the classical gradient in Equation 14 but which disappears from the natural gradient in Equation 22—is prohibitively expensive for variational parameters with many components. In the next section we will see that efficiently computing the natural gradient lets us develop scalable variational inference algorithms.

### 2.4 Stochastic Variational Inference

The coordinate ascent algorithm in Figure 3 is inefficient for large data sets because we must optimize the local variational parameters for each data point before re-estimating the global variational parameters. Stochastic variational inference uses stochastic optimization to fit the global variational parameters. We repeatedly subsample the data to form noisy estimates of the natural gradient of the ELBO, and we follow these estimates with a decreasing step-size.

We have reviewed mean-field variational inference in models with exponential family conditionals and showed that the natural gradient of the variational objective function is easy to compute. We now discuss stochastic optimization, which uses a series of noisy estimates of the gradient, and use it with noisy natural gradients to derive stochastic variational inference.

*Stochastic optimization.* Stochastic optimization algorithms follow noisy estimates of the gradient with a decreasing step size. Noisy estimates of a gradient are often cheaper to compute than the true gradient, and following such estimates can allow algorithms to escape shallow local optima of complex objective functions. In statistical estimation problems, including variational inference of the global parameters, the gradient can be written as a sum of terms (one for each data point) and we can compute a fast noisy approximation by subsampling the data. With certain conditions on the step-size schedule, these algorithms provably converge to an optimum (Robbins and Monro, 1951). Spall (2003) gives an overview of stochastic optimization; Bottou (2003) gives an overview of its role in machine learning.

Consider an objective function $f(\lambda)$ and a random function $B(\lambda)$ that has expectation equal to the gradient so that $\mathbb{E}_q[B(\lambda)] = \nabla_\lambda f(\lambda)$. The stochastic gradient algorithm, which is a type of stochastic optimization, optimizes $f(\lambda)$ by iteratively following realizations of $B(\lambda)$. At iteration $t$, the update for $\lambda$ is

$$\lambda^{(t)} = \lambda^{(t-1)} + \rho_t b_t(\lambda^{(t-1)}),$$

where $b_t$ is an independent draw from the noisy gradient $B$. If the sequence of step sizes $\rho_t$ satisfies

$$\sum \rho_t = \infty; \quad \sum \rho_t^2 < \infty \tag{23}$$

then $\lambda^{(t)}$ will converge to the optimal $\lambda^*$ (if $f$ is convex) or a local optimum of $f$ (if not convex).[4] The same results apply if we premultiply the noisy gradients $b_t$ by a sequence of positive-definite matrices $G_t^{-1}$ (whose eigenvalues are bounded) (Bottou, 1998). The resulting algorithm is

$$\lambda^{(t)} = \lambda^{(t-1)} + \rho_t G_t^{-1} b_t(\lambda^{(t-1)}).$$

As our notation suggests, we will use the Fisher metric for $G_t$, replacing stochastic Euclidean gradients with stochastic natural gradients.

*Stochastic variational inference.* We use stochastic optimization with noisy natural gradients to optimize the variational objective function. The resulting algorithm is in Figure 4. At each iteration we have a current setting of the global variational parameters. We repeat the following steps:

1. Sample a data point from the set; optimize its local variational parameters.
2. Form intermediate global variational parameters, as though we were running classical coordinate ascent and the sampled data point were repeated $N$ times to form the collection.
3. Update the global variational parameters to be a weighted average of the intermediate parameters and their current setting.

We show that this algorithm is stochastic natural gradient ascent on the global variational parameters.

Our goal is to find a setting of the global variational parameters $\lambda$ that maximizes the ELBO. Writing $\mathcal{L}$ as a function of the global and local variational parameters, Let the function $\phi(\lambda)$ return a local optimum of the local variational parameters so that

$$\nabla_\phi \mathcal{L}(\lambda, \phi(\lambda)) = 0.$$

Define the *locally maximized ELBO* $\mathcal{L}(\lambda)$ to be the ELBO when $\lambda$ is held fixed and the local variational parameters $\phi$ are set to a local optimum $\phi(\lambda)$,

$$\mathcal{L}(\lambda) \triangleq \mathcal{L}(\lambda, \phi(\lambda)).$$

We can compute the (natural) gradient of $\mathcal{L}(\lambda)$ by first finding the corresponding optimal local parameters $\phi(\lambda)$ and then computing the (natural) gradient of $\mathcal{L}(\lambda, \phi(\lambda))$, holding $\phi(\lambda)$ fixed. The reason is that the gradient of $\mathcal{L}(\lambda)$ is the same as the gradient of the two-parameter ELBO $\mathcal{L}(\lambda, \phi(\lambda))$,

$$\begin{aligned} \nabla_\lambda \mathcal{L}(\lambda) &= \nabla_\lambda \mathcal{L}(\lambda, \phi(\lambda)) + (\nabla_\lambda \phi(\lambda))^\top \nabla_\phi \mathcal{L}(\lambda, \phi(\lambda)) \\ &= \nabla_\lambda \mathcal{L}(\lambda, \phi(\lambda)), \end{aligned}$$

where $\nabla_\lambda \phi(\lambda)$ is the Jacobian of $\phi(\lambda)$ and we use the fact that the gradient of $\mathcal{L}(\lambda, \phi)$ with respect to $\phi$ is zero at $\phi(\lambda)$.

Stochastic variational inference optimizes the maximized ELBO $\mathcal{L}(\lambda)$ by subsampling the data to form noisy estimates of the natural gradient. First, we decompose $\mathcal{L}(\lambda)$ into a global term and a sum of local terms,

$$\mathcal{L}(\lambda) = \mathbb{E}_q[\log p(\beta)] - \mathbb{E}_q[\log q(\beta)] + \sum_{n=1}^{N} \max_{\phi_n} (\mathbb{E}_q[\log p(x_n, z_n \mid \beta)] - \mathbb{E}_q[\log q(z_n)]). \tag{24}$$

4. To find a local optimum, $f$ must be three-times differentiable and meet a few mild technical requirements (Bottou, 1998). The variational objective satisfies these criteria.

Now consider a variable that chooses an index of the data uniformly at random, $I \sim \text{Unif}(1,\ldots,N)$. Define $\mathcal{L}_I(\lambda)$ to be the following random function of the variational parameters,

$$\mathcal{L}_I(\lambda) \triangleq \mathbb{E}_q[\log p(\beta)] - \mathbb{E}_q[\log q(\beta)] + N \max_{\phi_I} \left(\mathbb{E}_q[\log p(x_I, z_I \mid \beta) - \mathbb{E}_q[\log q(z_I)]\right). \tag{25}$$

The expectation of $\mathcal{L}_I$ is equal to the objective in Equation 24. Therefore, the natural gradient of $\mathcal{L}_I$ with respect to each global variational parameter $\lambda$ is a noisy but unbiased estimate of the natural gradient of the variational objective. This process—sampling a data point and then computing the natural gradient of $\mathcal{L}_I$—will provide cheaply computed noisy gradients for stochastic optimization.

We now compute the noisy gradient. Suppose we have sampled the $i$th data point. Notice that Equation 25 is equivalent to the full objective of Equation 24 where the $i$th data point is observed $N$ times. Thus the natural gradient of Equation 25—which is a noisy natural gradient of the ELBO—can be found using Equation 22,

$$\hat{\nabla}\mathcal{L}_i = \mathbb{E}_q\left[\eta_g\left(x_i^{(N)}, z_i^{(N)}, \alpha\right)\right] - \lambda,$$

where $\left\{x_i^{(N)}, z_i^{(N)}\right\}$ are a data set formed by $N$ replicates of observation $x_n$ and hidden variables $z_n$.

We compute this expression in more detail. Recall the complete conditional $\eta_g(x,z,\alpha)$ from Equation 6. From this equation, we can compute the conditional natural parameter for the global parameter given $N$ replicates of $x_n$,

$$\eta_g\left(x_i^{(N)}, z_i^{(N)}, \alpha\right) = \alpha + N \cdot (t(x_n, z_n), 1).$$

Using this in the natural gradient of Equation 22 gives a noisy natural gradient,

$$\hat{\nabla}_\lambda \mathcal{L}_i = \alpha + N \cdot \left(\mathbb{E}_{\phi_i(\lambda)}[t(x_i, z_i)], 1\right) - \lambda,$$

where $\phi_i(\lambda)$ gives the elements of $\phi(\lambda)$ associated with the $i$th local context. While the full natural gradient would use the local variational parameters for the whole data set, the noisy natural gradient only considers the local parameters for one randomly sampled data point. These noisy gradients are cheaper to compute.

Finally, we use the noisy natural gradients in a Robbins-Monro algorithm to optimize the ELBO. We sample a data point $x_i$ at each iteration. Define the intermediate global parameter $\hat{\lambda}_t$ to be the estimate of $\lambda$ that we would obtain if the sampled data point was replicated $N$ times,

$$\hat{\lambda}_t \triangleq \alpha + N\mathbb{E}_{\phi_i(\lambda)}[(t(x_i, z_i), 1)].$$

This comprises the first two terms of the noisy natural gradient. At each iteration we use the noisy gradient (with step size $\rho_t$) to update the global variational parameter. The update is

$$\begin{aligned} \lambda^{(t)} &= \lambda^{(t-1)} + \rho_t\left(\hat{\lambda}_t - \lambda^{(t-1)}\right) \\ &= (1-\rho_t)\lambda^{(t-1)} + \rho_t\hat{\lambda}_t. \end{aligned}$$

This is a weighted average of the previous estimate of $\lambda$ and the estimate of $\lambda$ that we would obtain if the sampled data point was replicated $N$ times.

1: Initialize $\lambda^{(0)}$ randomly.
2: Set the step-size schedule $\rho_t$ appropriately.
3: **repeat**
4: Sample a data point $x_i$ uniformly from the data set.
5: Compute its local variational parameter,

$$\phi = \mathbb{E}_{\lambda^{(t-1)}}[\eta_g(x_i^{(N)}, z_i^{(N)})].$$

6: Compute intermediate global parameters as though $x_i$ is replicated $N$ times,

$$\hat{\lambda} = \mathbb{E}_{\phi}[\eta_g(x_i^{(N)}, z_i^{(N)})].$$

7: Update the current estimate of the global variational parameters,

$$\lambda^{(t)} = (1-\rho_t)\lambda^{(t-1)} + \rho_t\hat{\lambda}.$$

8: **until** forever

Figure 4: Stochastic variational inference.

Figure 4 presents the full algorithm. At each iteration, the algorithm has an estimate of the global variational parameter $\lambda^{(t-1)}$. It samples a single data point from the data and cheaply computes the intermediate global parameter $\hat{\lambda}_t$, that is, the next value of $\lambda$ if the data set contained $N$ replicates of the sampled point. It then sets the new estimate of the global parameter to be a weighted average of the previous estimate and the intermediate parameter.

We set the step-size at iteration $t$ as follows,

$$\rho_t = (t+\tau)^{-\kappa}. \tag{26}$$

This satisfies the conditions in Equation 23. The *forgetting rate* $\kappa \in (0.5, 1]$ controls how quickly old information is forgotten; the *delay* $\tau \geq 0$ down-weights early iterations. In Section 4 we fix the delay to be one and explore a variety of forgetting rates. Note that this is just one way to parameterize the learning rate. As long as the step size conditions in Equation 23 are satisfied, this iterative algorithm converges to a local optimum of the ELBO.

### 2.5 Extensions

We now describe two extensions of the basic stochastic inference algorithm in Figure 4: the use of multiple samples ("minibatches") to improve the algorithm's stability, and empirical Bayes methods for hyperparameter estimation.

*Minibatches.* So far, we have considered stochastic variational inference algorithms where only one observation $x_t$ is sampled at a time. Many stochastic optimization algorithms benefit from "minibatches," that is, several examples at a time (Bottou and Bousquet, 2008; Liang et al., 2009; Mairal et al., 2010). In stochastic variational inference, we can sample a set of $S$ examples at each iteration $x_{t,1:S}$ (with or without replacement), compute the local variational parameters $\phi_s(\lambda^{(t-1)})$ for

each data point, compute the intermediate global parameters $\hat{\lambda}_s$ for each data point $x_{ts}$, and finally average the $\hat{\lambda}_s$ variables in the update

$$\lambda^{(t)} = (1-\rho_t)\lambda^{(t-1)} + \frac{\rho_t}{S}\sum_s \hat{\lambda}_s.$$

The stochastic natural gradients associated with each point $x_s$ have expected value equal to the gradient. Therefore, the average of these stochastic natural gradients has the same expectation and the algorithm remains valid.

There are two reasons to use minibatches. The first reason is to amortize any computational expenses associated with updating the global parameters across more data points; for example, if the expected sufficient statistics of $\beta$ are expensive to compute, using minibatches allows us to incur that expense less frequently. The second reason is that it may help the algorithm to find better local optima. Stochastic variational inference is guaranteed to converge to a local optimum but taking large steps on the basis of very few data points may lead to a poor one. As we will see in Section 4, using more of the data per update can help the algorithm.

*Empirical Bayes estimation of hyperparameters.* In some cases we may want to both estimate the posterior of the hidden random variables $\beta$ and $z$ and obtain a point estimate of the values of the hyperparameters $\alpha$. One approach to fitting $\alpha$ is to try to maximize the marginal likelihood of the data $p(x \mid \alpha)$, which is also known as empirical Bayes (Maritz and Lwin, 1989) estimation. Since we cannot compute $p(x \mid \alpha)$ exactly, an approximate approach is to maximize the fitted variational lower bound $\mathcal{L}$ over $\alpha$. In the non-stochastic setting, $\alpha$ can be optimized by interleaving the coordinate ascent updates in Figure 3 with an update for $\alpha$ that increases the ELBO. This is called variational expectation-maximization.

In the stochastic setting, we update $\alpha$ simultaneously with $\lambda$. We can take a step in the direction of the gradient of the noisy ELBO $\mathcal{L}_t$ (Equation 25) with respect to $\alpha$, scaled by the step-size $\rho_t$,

$$\alpha^{(t)} = \alpha^{(t-1)} + \rho_t \nabla_\alpha \mathcal{L}_t(\lambda^{(t-1)}, \phi, \alpha^{(t-1)}).$$

Here $\lambda^{(t-1)}$ are the global parameters from the previous iteration and $\phi$ are the optimized local parameters for the currently sampled data point. We can also replace the standard Euclidean gradient with a natural gradient or Newton step.

## 3. Stochastic Variational Inference in Topic Models

We derived stochastic variational inference, a scalable inference algorithm that can be applied to a large class of hierarchical Bayesian models. In this section we show how to use the general algorithm of Section 2 to derive stochastic variational inference for two probabilistic topic models: latent Dirichlet allocation (LDA) (Blei et al., 2003) and its Bayesian nonparametric counterpart, the hierarchical Dirichlet process (HDP) topic model (Teh et al., 2006a).

Topic models are probabilistic models of document collections that use latent variables to encode recurring patterns of word use (Blei, 2012). Topic modeling algorithms are inference algorithms; they uncover a set of patterns that pervade a collection and represent each document according to how it exhibits them. These patterns tend to be thematically coherent, which is why the models are called "topic models." Topic models are used for both descriptive tasks, such as to build thematic navigators of large collections of documents, and for predictive tasks, such as to aid document classification. Topic models have been extended and applied in many domains.

Topic models assume that the words of each document arise from a mixture of multinomials. Across a collection, the documents share the same mixture components (called *topics*). Each document, however, is associated with its own mixture proportions (called *topic proportions*). In this way, topic models represent documents heterogeneously—the documents share the same set of topics, but each exhibits them to a different degree. For example, a document about sports and health will be associated with the sports and health topics; a document about sports and business will be associated with the sports and business topics. They both share the sports topic, but each combines sports with a different topic. More generally, this is called *mixed membership* (Erosheva, 2003).

The central computational problem in topic modeling is posterior inference: Given a collection of documents, what are the topics that it exhibits and how does each document exhibit them? In practical applications of topic models, scale is important—these models promise an unsupervised approach to organizing large collections of text (and, with simple adaptations, images, sound, and other data). Thus they are a good testbed for stochastic variational inference.

More broadly, this section illustrates how to use the results from Section 2 to develop algorithms for specific models. We will derive the algorithms in several steps: (1) we specify the model assumptions; (2) we derive the complete conditional distributions of the latent variables; (3) we form the mean-field variational family; (4) we derive the corresponding stochastic inference algorithm. In Section 4, we will report our empirical study of stochastic variational inference with these models.

### 3.1 Notation

We follow the notation of Blei et al. (2003).

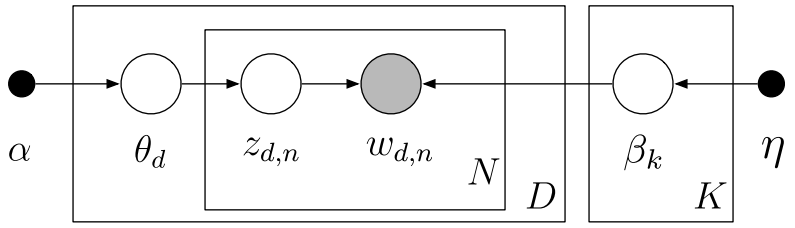
- Observations are *words*, organized into documents. The $n$th word in the $d$th document is $w_{dn}$. Each word is an element in a fixed vocabulary of $V$ terms.
- A *topic* $\beta_k$ is a distribution over the vocabulary. Each topic is a point on the $V-1$-simplex, a positive vector of length $V$ that sums to one. We denote the $w$th entry in the $k$th topic as $\beta_{kw}$. In LDA there are $K$ topics; in the HDP topic model there are an infinite number of topics.
- Each document in the collection is associated with a vector of *topic proportions* $\theta_d$, which is a distribution over topics. In LDA $\theta_d$ is a point on the $K-1$-simplex. In the HDP topic model, $\theta_d$ is a point on the infinite simplex. (We give details about this below in Section 3.3.) We denote the $k$th entry of the topic proportion vector $\theta_d$ as $\theta_{dk}$.
- Each word in each document is assumed to have been drawn from a single topic. The *topic assignment* $z_{dn}$ indexes the topic from which $w_{dn}$ is drawn.

The only observed variables are the words of the documents. The topics, topic proportions, and topic assignments are latent variables.

### 3.2 Latent Dirichlet Allocation

LDA is the simplest topic model. It assumes that each document exhibits $K$ topics with different proportions. The generative process is

1. Draw topics $\beta_k \sim \text{Dirichlet}(\eta,\dots,\eta)$ for $k \in \{1,\dots,K\}$.
2. For each document $d \in \{1,\dots,D\}$:




| Var | Type | Conditional | Param | Relevant Expectations |
|---|---|---|---|---|
| $z_{dn}$ | Multinomial | $\log \theta_{dk} + \log \beta_{k,w_{dn}}$ | $\phi_{dn}$ | $\mathbb{E}[Z^k_{dn}] = \phi^k_{dn}$ |
| $\theta_d$ | Dirichlet | $\alpha + \sum_{n=1}^{N} z_{dn}$ | $\gamma_d$ | $\mathbb{E}[\log \theta_{dk}] = \Psi(\gamma_{dk}) - \sum_{j=1}^{K} \Psi(\gamma_{dj})$ |
| $\beta_k$ | Dirichlet | $\eta + \sum_{d=1}^{D} \sum_{n=1}^{N} z^k_{dn} w_{dn}$ | $\lambda_k$ | $\mathbb{E}[\log \beta_{kv}] = \Psi(\lambda_{kv}) - \sum_{y=1}^{V} \Psi(\lambda_{ky})$ |

Figure 5: (Top) The graphical model representation of Latent Dirichlet allocation. Note that in practice each document $d$ may not have the same number of words $N$. (Bottom) In LDA: hidden variables, complete conditionals, variational parameters, and expected sufficient statistics.

(a) Draw topic proportions $\theta \sim \text{Dirichlet}(\alpha, \ldots, \alpha)$.

(b) For each word $w \in \{1, \ldots, N\}$:

i. Draw topic assignment $z_{dn} \sim \text{Multinomial}(\theta_d)$.

ii. Draw word $w_{dn} \sim \text{Multinomial}(\beta_{z_{dn}})$.

Figure 5 illustrates LDA as a graphical model.

In LDA, each document exhibits the same shared topics but with different proportions. LDA assumes Dirichlet priors for $\beta_k$ and $\theta_d$. Dirichlet distributions over the $D$-simplex take $D+1$ parameters, but for simplicity we assume exchangeable Dirichlet priors; that is, we require that all of these parameters are set to the same value. (The prior on $\beta_k$ has parameter $\eta$; the prior on $\theta_d$ has parameter $\alpha$.). We note that Blei et al. (2003) and Wallach et al. (2009) found improved empirical performance with non-exchangeable priors.

LDA models an observed collection of documents $w = w_{1:D}$, where each $w_d$ is a collection of words $w_{d,1:N}$. Analyzing the documents amounts to posterior inference of $p(\beta, \theta, z \mid w)$. Conditioned on the documents, the posterior distribution captures the topics that describe them ($\beta = \beta_{1:K}$), the degree to which each document exhibits those topics ($\theta = \theta_{1:D}$), and which topics each word was assigned to ($z = z_{1:D,1:N}$). We can use the posterior to explore large collections of documents. Figure 1 illustrates posterior topics found with stochastic variational inference.

The posterior is intractable to compute (Blei et al., 2003). Approximating the posterior in LDA is a central computational problem for topic modeling. Researchers have developed many methods, including Markov chain Monte Carlo methods (Griffiths and Steyvers, 2004), expectation propagation (Minka and Lafferty, 2002), and variational inference (Blei et al., 2003; Teh et al., 2006b; Asuncion et al., 2009). Here we use the results of Section 2 to develop stochastic inference for LDA. This scales the original variational algorithm for LDA to massive collections of documents.[5]

Figure 7 illustrates the performance of 100-topic LDA on three large collections—*Nature* contains 350K documents, *New York Times* contains 1.8M documents, and *Wikipedia* contains 3.8M documents. (Section 4 describes the complete study, including the details of the performance measure and corpora.) We compare two inference algorithms for LDA: stochastic inference on the full collection and batch inference on a subset of 100,000 documents. (This is the size of collection that batch inference can handle.) We see that stochastic variational inference converges faster and to a better model. It is both more efficient and lets us handle the full data set.

*Indicator vectors and Dirichlet distributions.* Before deriving the algorithm, we discuss two mathematical details. These will be useful both here and in the next section.

First, we represent categorical variables like the topic assignments $z_{dn}$ and observed words $w_{dn}$ with *indicator vectors*. An indicator vector is a binary vector with a single one. For example, the topic assignment $z_{dn}$ can take on one of $K$ values (one for each topic). Thus, it is represented as a $K$-vector with a one in the component corresponding to the value of the variable: if $z_{dn}^k = 1$ then the $n$th word in document $d$ is assigned to the $k$th topic. Likewise, $w_{dn}^v = 1$ implies that the $n$th word in document $d$ is $v$. In a slight abuse of notation, we will sometimes use $w_{dn}$ and $z_{dn}$ as indices—for example, if $z_{dn}^k = 1$, then $\beta_{z_{dn}}$ refers to the $k$th topic $\beta_k$.

Second, we review the Dirichlet distribution. As we described above, a $K$-dimensional Dirichlet is a distribution on the $K-1$-simplex, that is, positive vectors over $K$ elements that sum to one. It is

5. The algorithm we present was originally developed in Hoffman et al. (2010a), which is a special case of the stochastic variational inference algorithm we developed in Section 2.

parameterized by a positive $K$-vector $\gamma$,

$$\text{Dirichlet}(\theta;\gamma) = \frac{\Gamma\left(\sum_{i=1}^{K}\gamma_i\right)}{\prod_{i=1}^{K}\Gamma(\gamma_i)}\prod_{i=1}^{K}\theta^{\gamma_i-1},$$

where $\Gamma(\cdot)$ is the Gamma function, which is a real-valued generalization of the factorial function. The expectation of the Dirichlet is its normalized parameter,

$$\mathbb{E}[\theta_k \,|\, \gamma] = \frac{\gamma_k}{\sum_{i=1}^{K}\gamma_i}.$$

The expectation of its log uses $\Psi(\cdot)$, which is the first derivative of the log Gamma function,

$$\mathbb{E}[\log\theta_k \,|\, \gamma] = \Psi(\gamma_k) - \Psi\left(\textstyle\sum_{i=1}^{K}\gamma_i\right). \tag{27}$$

This can be derived by putting the Dirichlet in exponential family form, noticing that $\log\theta$ is the vector of sufficient statistics, and computing its expectation by taking the gradient of the log-normalizer with respect to the natural parameter vector $\gamma$.

*Complete conditionals and variational distributions.* We specify the global and local variables of LDA to place it in the stochastic variational inference setting of Section 2. In topic modeling, the local context is a document $d$. The local observations are its observed words $w_{d,1:N}$. The local hidden variables are the topic proportions $\theta_d$ and the topic assignments $z_{d,1:N}$. The global hidden variables are the topics $\beta_{1:K}$.

Recall from Section 2 that the complete conditional is the conditional distribution of a variable given all of the other variables, hidden and observed. In mean-field variational inference, the variational distributions of each variable are in the same family as the complete conditional.

We begin with the topic assignment $z_{dn}$. The complete conditional of the topic assignment is a multinomial,

$$p(z_{dn} = k \,|\, \theta_d, \beta_{1:K}, w_{dn}) \propto \exp\{\log\theta_{dk} + \log\beta_{k,w_{dn}}\}. \tag{28}$$

Thus its variational distribution is a multinomial $q(z_{dn}) = \text{Multinomial}(\phi_{dn})$, where the variational parameter $\phi_{dn}$ is a point on the $K-1$-simplex. Per the mean-field approximation, each observed word is endowed with a different variational distribution for its topic assignment, allowing different words to be associated with different topics.

The complete conditional of the topic proportions is a Dirichlet,

$$p(\theta_d \,|\, z_d) = \text{Dirichlet}\left(\alpha + \textstyle\sum_{n=1}^{N} z_{dn}\right). \tag{29}$$

Since $z_{dn}$ is an indicator vector, the $k$th element of the parameter to this Dirichlet is the sum of the hyperparameter $\alpha$ and the number of words assigned to topic $k$ in document $d$. Note that, although we have assumed an exchangeable Dirichlet prior, when we condition on $z$ the conditional $p(\theta_d|z_d)$ is a non-exchangeable Dirichlet.

With this conditional, the variational distribution of the topic proportions is also Dirichlet $q(\theta_d) = \text{Dirichlet}(\gamma_d)$, where $\gamma_d$ is a $K$-vector Dirichlet parameter. There is a different variational Dirichlet parameter for each document, allowing different documents to be associated with different topics in different proportions.

These are local hidden variables. The complete conditionals only depend on other variables in the local context (i.e., the document) and the global variables; they do not depend on variables from other documents.

Finally, the complete conditional for the topic $\beta_k$ is also a Dirichlet,

$$p(\beta_k \mid z, w) = \text{Dirichlet}\left(\eta + \sum_{d=1}^{D}\sum_{n=1}^{N} z_{dn}^k w_{dn}\right). \tag{30}$$

The $v$th element of the parameter to the Dirichlet conditional for topic $k$ is the sum of the hyperparameter $\eta$ and the number of times that the term $v$ was assigned to topic $k$. This is a global variable—its complete conditional depends on the words and topic assignments of the entire collection.

The variational distribution for each topic is a $V$-dimensional Dirichlet,

$$q(\beta_k) = \text{Dirichlet}(\lambda_k).$$

As we will see in the next section, the traditional variational inference algorithm for LDA is inefficient with large collections of documents. The root of this inefficiency is the update for the topic parameter $\lambda_k$, which (from Equation 30) requires summing over variational parameters for every word in the collection.

*Batch variational inference.*

With the complete conditionals in hand, we now derive the coordinate ascent variational inference algorithm, that is, the batch inference algorithm of Figure 3. We form each coordinate update by taking the expectation of the natural parameter of the complete conditional. This is the stepping stone to stochastic variational inference.

The variational parameters are the global per-topic Dirichlet parameters $\lambda_{1:K}$, local per-document Dirichlet parameters $\gamma_{1:D}$, and local per-word multinomial parameters $\phi_{1:D,1:N}$. Coordinate ascent variational inference iterates between updating all of the local variational parameters (Equation 16) and updating the global variational parameters (Equation 15).

We update each document $d$'s local variational in a local coordinate ascent routine, iterating between updating each word's topic assignment and the per-document topic proportions,

$$\phi_{dn}^k \propto \exp\{\Psi(\gamma_{dk}) + \Psi(\lambda_{k,w_{dn}}) - \Psi(\textstyle\sum_v \lambda_{kv})\} \quad \text{for } n \in \{1,\ldots,N\}, \tag{31}$$

$$\gamma_d = \alpha + \textstyle\sum_{n=1}^{N} \phi_{dn}. \tag{32}$$

These updates derive from taking the expectations of the natural parameters of the complete conditionals in Equation 28 and Equation 29. (We then map back to the usual parameterization of the multinomial.) For the update on the topic assignment, we have used the Dirichlet expectations in Equation 27. For the update on the topic proportions, we have used that the expectation of an indicator is its probability, $\mathbb{E}_q[z_{dn}^k] = \phi_{dn}^k$.

After finding variational parameters for each document, we update the variational Dirichlet for each topic,

$$\lambda_k = \eta + \sum_{d=1}^{D}\sum_{n=1}^{N} \phi_{dn}^k w_{dn}. \tag{33}$$

This update depends on the variational parameters $\phi$ from every document.

Batch inference is inefficient for large collections of documents. Before updating the topics $\lambda_{1:K}$, we compute the local variational parameters for every document. This is particularly wasteful in the beginning of the algorithm when, before completing the first iteration, we must analyze every document with randomly initialized topics.

*Stochastic variational inference*

Initialize $\lambda^{(0)}$ randomly.
Set the step-size schedule $\rho_t$ appropriately.
**repeat**
  Sample a document $w_d$ uniformly from the data set.
  Initialize $\gamma_{dk} = 1$, for $k \in \{1,\ldots,K\}$.
  **repeat**
    For $n \in \{1,\ldots,N\}$ set

$$\phi_{dn}^k \propto \exp\{\mathbb{E}[\log\theta_{dk}] + \mathbb{E}[\log\beta_{k,w_{dn}}]\}, k \in \{1,\ldots,K\}.$$

    Set $\gamma_d = \alpha + \sum_n \phi_{dn}$.
  **until** local parameters $\phi_{dn}$ and $\gamma_d$ converge.
  For $k \in \{1,\ldots,K\}$ set intermediate topics

$$\hat{\lambda}_k = \eta + D\sum_{n=1}^{N} \phi_{dn}^k w_{dn}.$$

  Set $\lambda^{(t)} = (1-\rho_t)\lambda^{(t-1)} + \rho_t\hat{\lambda}$.
**until** forever

Figure 6: Stochastic variational inference for LDA. The relevant expectations for each update are found in Figure 5.

Stochastic variational inference provides a scalable method for approximate posterior inference in LDA. The global variational parameters are the topic Dirichlet parameters $\lambda_k$; the local variational parameters are the per-document topic proportion Dirichlet parameters $\gamma_d$ and the per-word topic assignment multinomial parameters $\phi_{dn}$.

We follow the general algorithm of Figure 4. Let $\lambda^{(t)}$ be the topics at iteration $t$. At each iteration we sample a document $d$ from the collection. In the local phase, we compute optimal variational parameters by iterating between updating the per-document parameters $\gamma_d$ (Equation 32) and $\phi_{d,1:N}$ (Equation 31). This is the same subroutine as in batch inference, though here we only analyze one randomly chosen document.

In the global phase we use these fitted local variational parameters to form intermediate topics,

$$\hat{\lambda}_k = \eta + D\textstyle\sum_{n=1}^{N} \phi_{dn}^k w_{dn}.$$

This comes from applying Equation 33 to a hypothetical corpus containing $D$ replicates of document $d$. We then set the topics at the next iteration to be a weighted combination of the intermediate topics and current topics,

$$\lambda_k^{(t+1)} = (1-\rho_t)\lambda_k^{(t)} + \rho_t\hat{\lambda}_k.$$

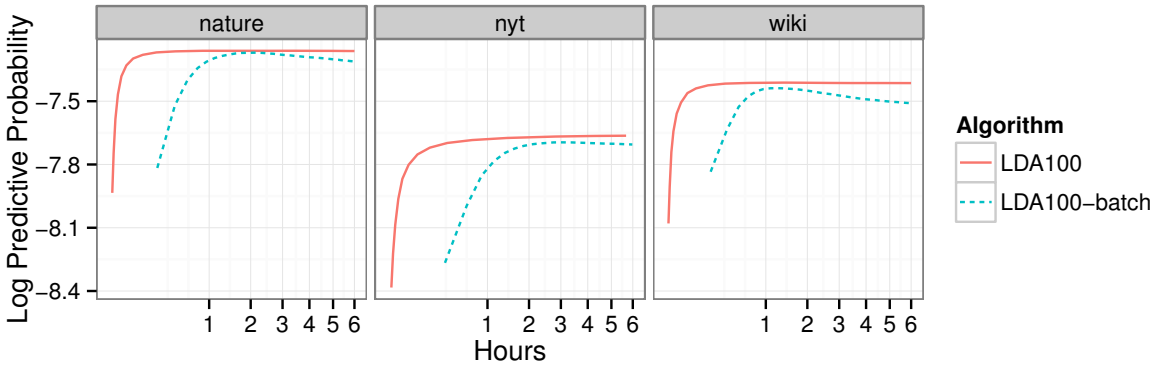



Figure 7: The per-word predictive log likelihood for a 100-topic LDA model on three large corpora. (Time is on the square root scale.) Stochastic variational inference on the full data converges faster and to a better place than batch variational inference on a reasonably sized subset. Section 4 gives the details of our empirical study.

Figure 6 gives the algorithm for stochastic variational inference for LDA.[6]

### 3.3 Bayesian Nonparametric Topic Models with the HDP

Stochastic inference for LDA lets us analyze large collections of documents. One limitation of LDA, however, is that the number of topics is fixed in advance. Typically, researchers find the "best" number of topics with cross-validation (Blei et al., 2003). However, for very large data this approach is not practical. We can address this issue with a Bayesian nonparametric topic model, a model where the documents themselves determine the number of topics.

We derive stochastic variational inference for the Bayesian nonparametric variant of LDA, the hierarchical Dirichlet process (HDP) topic model. Like LDA, the HDP topic model is a mixed-membership model of text collections. However, the HDP assumes an "infinite" number of topics. Given a collection of documents, the posterior distribution of the hidden structure determines how many topics are needed to describe them. Further, the HDP is flexible in that it allows future data to exhibit new and previously unseen topics.

More broadly, stochastic variational inference for the HDP topic model demonstrates the possibilities of stochastic inference in the context of Bayesian nonparametric statistics. Bayesian nonparametrics gives us a collection of flexible models—mixture models, mixed-membership models, factor models, and models with more complex structure—which grow and expand with data (Hjort et al., 2010). Flexible and expanding models are particularly important when analyzing large data sets, where it is prohibitive to search for a specific latent structure (such as a number of topics or a tree structure of components) with cross-validation. Here we demonstrate how to use stochastic

6. This algorithm, as well as the algorithm for the HDP, specifies that we initialize the topics $\lambda_k$ randomly. There are many ways to initialize the topics. We use an exponential distribution,

$$\lambda_{kv} - \eta \sim \text{Exponential}(D * 100/(KV)).$$

This gives a setting of $\lambda$ similar to the one we would get by applying Equation 33 after randomly assigning words to topics in a corpus of size $D$ with 100 words per document.

inference in the context of a simple Bayesian nonparametric topic model. In other work, we built on this algorithm to give scalable inference methods for Bayesian nonparametric models of topic correlation (Paisley et al., 2012b) and tree structures of topics (Paisley et al., 2012c).

This section is organized as follows. We first give some background on the Dirichlet process and its definition via Sethuraman's stick breaking construction, which is a distribution on the infinite simplex. We then show how to use this construction to form the HDP topic model and how to use stochastic variational inference to approximate the posterior.[7]

*The stick-breaking construction of the Dirichlet process.* Bayesian nonparametric (BNP) methods use distributions of distributions, placing flexible priors on the shape of the data-generating density function. BNP models draw a distribution from that prior and then independently draw data from that random distribution. Data analysis proceeds by evaluating the posterior distribution of the (random) distribution from which the data were drawn. Because of the flexible prior, that posterior can potentially have mass on a wide variety of distribution shapes. For a reviews of BNP methods, see the edited volume of Hjort et al. (2010) and the tutorial of Gershman and Blei (2012).

The most common BNP prior is the *Dirichlet process* (DP). The Dirichlet process is parameterized by a *base distribution* $G_0$ (which may be either continuous or discrete) and a non-negative scaling factor $\alpha$. These are used to form a distribution over discrete distributions, that is, over distributions that place their mass on a countably infinite set of atoms. The locations of the atoms are independently drawn from the base distribution $G_0$ and the closeness of the probabilities to $G_0$ is determined by the scaling factor $\alpha$. When $\alpha$ is small, more mass is placed on fewer atoms, and the draw will likely look very different from $G_0$; when $\alpha$ is large, the mass is spread around many atoms, and the draw will more closely resemble the base distribution.

There are several representations of the Dirichlet process. For example, it is a normalized gamma process (Ferguson, 1973), and its marginalization gives the Chinese restaurant process (Pitman, 2002). We will focus on its definition via Sethuraman's stick breaking construction (Sethuraman, 1994). The stick-breaking construction explicitly defines the distribution of the probabilities that make up a random discrete distribution. It is the gateway to variational inference in Bayesian nonparametric models (Blei and Jordan, 2006).

Let $G \sim \mathrm{DP}(\alpha, G_0)$ be drawn from a Dirichlet process prior. It is a discrete distribution with mass on an infinite set of atoms. Let $\beta_k$ be the atoms in this distribution and $\sigma_k$ be their corresponding probabilities. We can write $G$ as

$$G = \sum_{k=1}^{\infty} \sigma_k \delta_{\beta_k}.$$

The atoms are drawn independently from $G_0$. The stick-breaking construction specifies the distribution of their probabilities.

The stick-breaking construction uses an infinite collection of beta-distributed random variables. Recall that the beta is a distribution on $(0,1)$ and define the following collection,

$$v_i \sim \mathrm{Beta}(1,\alpha) \quad i \in \{1,2,3,\ldots\}.$$

These variables combine to form a point on the infinite simplex. Imagine a stick of unit length. Break off the proportion of the stick given by $v_1$, call it $\sigma_1$, and set it aside. From the remainder (of length $1-\sigma_1$) break off the proportion given by $v_2$, call it $\sigma_2$, and set it aside. The remainder of

7. This algorithm first appeared in Wang et al. (2011). Here we place it in the more general context of Section 2 and relate it to stochastic inference for LDA.

the stick is now $1-\sigma_2-\sigma_1=(1-v_1)(1-v_2)$. Repeat this process for the infinite set of $v_i$. The resulting stick lengths $\sigma_i$ will sum to one.

More formally, we define the function $\sigma_i$ to take the collection of realized $v_i$ variables and to return the stick length of the $i$th component,

$$\sigma_i(v) = v_i \prod_{j=1}^{i-1}(1-v_j),$$

and note that $\sum_{i=1}^{\infty}\sigma_i(v)=1$. We call $v_i$ the $i$th *breaking proportion*.

Combining these steps, we form the distribution $G$ according to the following process,

$$\begin{aligned}
\beta_i &\sim G_0 \quad i\in\{1,2,3,\ldots\},\\
v_i &\sim \text{Beta}(1,\alpha) \quad i\in\{1,2,3,\ldots\},\\
G &= \textstyle\sum_{i=1}^{\infty}\sigma_i(v)\delta_{\beta_i}.
\end{aligned}$$

In the random distribution $G$ the $i$th atom $\beta_i$ is an independent draw from $G_0$ and it has probability given by the $i$th stick length $\sigma_i(v)$. Sethuraman (1994) showed that the distribution of $G$ is $\text{DP}(\alpha,G_0)$.

The most important property of $G$ is the "clustering" property. Even though $G$ places mass on a countably infinite set of atoms, $N$ draws from $G$ will tend to exhibit only a small number of them. (How many depends on the scalar $\alpha$, as we described above.) Formally, this is most easily seen via other perspectives on the DP (Ferguson, 1973; Blackwell and MacQueen, 1973; Pitman, 2002), though it can be seen intuitively with the stick-breaking construction. The intuition is that as $\alpha$ gets smaller more of the stick is absorbed in the first break locations because the breaking proportions are drawn from $\text{Beta}(1,\alpha)$. Thus, those atoms associated with the first breaks of the stick will have larger mass in the distribution $G$, and that in turn encourages draws from the distribution to realize fewer individual atoms. In general, the first break locations tend to be larger than the later break locations. This property is called *size biasedness*.

*The HDP topic model.* We now construct a Bayesian nonparametric topic model that has an "infinite" number of topics. The hierarchical Dirichlet process topic model couples a set of document-level DPs via a single top-level DP (Teh et al., 2006a). The base distribution $H$ of the top-level DP is a symmetric Dirichlet over the vocabulary simplex—its atoms are topics. We draw once from this DP, $G_0\sim\text{DP}(\omega,H)$. In the second level, we use $G_0$ as a base measure to a document-level DP, $G_d\sim\text{DP}(\alpha,G_0)$. We draw the words of each document $d$ from topics from $G_d$. The consequence of this two-level construction is that all documents share the same collection of topics but exhibit them with different proportions.

We construct the HDP topic model using a stick-breaking construction at each level—one at the document level and one at the corpus level.[8] The generative process of the HDP topic model is as follows.

1. Draw an infinite number of topics, $\beta_k\sim\text{Dirichlet}(\eta)$ for $k\in\{1,2,3,\ldots\}$.
2. Draw corpus breaking proportions, $v_k\sim\text{Beta}(1,\omega)$ for $k\in\{1,2,3,\ldots\}$.
3. For each document $d$:

8. See the original HDP paper of Teh et al. (2006a) for other constructions of the HDP—the random measure construction, the construction by the Chinese restaurant franchise, and an alternative stick-breaking construction. This construction was mentioned by Fox et al. (2008). We used it for the HDP in Wang et al. (2011).

(a) Draw document-level topic indices, $c_{di} \sim \text{Multinomial}(\sigma(v))$ for $i \in \{1,2,3,\ldots\}$.

(b) Draw document breaking proportions, $\pi_{di} \sim \text{Beta}(1,\alpha)$ for $i \in \{1,2,3,\ldots\}$.

(c) For each word $n$:

i. Draw topic assignment $z_{dn} \sim \text{Multinomial}(\sigma(\pi_d))$.

ii. Draw word $w_n \sim \text{Multinomial}(\beta_{c_{d,z_{dn}}})$.

Figure 8 illustrates this process as a graphical model.

In this construction, topics $\beta_k$ are drawn as in LDA (Step 1). Corpus-level breaking proportions $v$ (Step 2) define a probability distribution on these topics, which indicates their relative prevalence in the corpus. At the document level, breaking proportions $\pi_d$ create a set of probabilities (Step 3b) and topic indices $c_d$, drawn from $\sigma(v)$, attach each document-level stick length to a topic (Step 3a). This creates a document-level distribution over topics, and words are then drawn as for LDA (Step 3c).

The posterior distribution of the HDP topic model gives a mixed-membership decomposition of a corpus where the number of topics is unknown in advance and unbounded. However, it is not possible to compute the posterior. Approximate posterior inference for BNP models in general is an active field of research (Escobar and West, 1995; Neal, 2000; Blei and Jordan, 2006; Teh et al., 2007).

The advantage of our construction over others is that it meets the conditions of Section 2. All the complete conditionals are in exponential families in closed form, and it neatly separates global variables from local variables. The global variables are topics and corpus-level breaking proportions; the local variables are document-level topic indices and breaking proportions. Following the same procedure as for LDA, we now derive stochastic variational inference for the HDP topic model.

*Complete conditionals and variational distributions.* We form the complete conditional distributions of all variables in the HDP topic model. We begin with the latent indicator variables,

$$\begin{aligned} p(z^i_{dn} = 1 | \pi_d, \beta_{1:K}, w_{dn}, c_d) &\propto \exp\{\log \sigma_i(\pi_d) + \textstyle\sum_{k=1}^{\infty} c^k_{di} \log \beta_{k,w_{dn}}\}, \\ p(c^k_{di} = 1 | v, \beta_{1:K}, w_d, z_d) &\propto \exp\{\log \sigma_k(v) + \textstyle\sum_{n=1}^{N} z^i_{dn} \log \beta_{k,w_{dn}}\}. \end{aligned}$$

Note the interaction between the two levels of latent indicators. In LDA the $i$th component of the topic proportions points to the $i$th topic. Here we must account for the topic index $c_{di}$, which is a random variable that points to one of the topics.

This interaction between indicators is also seen in the conditionals for the topics,

$$p(\beta_k | z, c, w) = \text{Dirichlet}\left(\eta + \textstyle\sum_{d=1}^{D} \sum_{i=1}^{\infty} c^k_{di} \sum_{n=1}^{N} z^i_{dn} w_{dn}\right).$$

The innermost sum collects the sufficient statistics for words in the $d$th document that are allocated to the $i$th local component index. However, these statistics are only kept when the $i$th topic index $c_{di}$ points to the $k$th global topic.

The full conditionals for the breaking proportions follow those of a standard stick-breaking construction (Blei and Jordan, 2006),

$$\begin{aligned} p(v_k | c) &= \text{Beta}\left(1 + \textstyle\sum_{d=1}^{D} \sum_{i=1}^{\infty} c^k_{di}, \omega + \sum_{d=1}^{D} \sum_{i=1}^{\infty} \sum_{j>k} c^j_{di}\right), \\ p(\pi_{di} | z_d) &= \text{Beta}\left(1 + \textstyle\sum_{n=1}^{N} z^i_{dn}, \alpha + \sum_{n=1}^{N} \sum_{j>i} z^j_{dn}\right). \end{aligned}$$

The complete conditionals for all the latent variables are all in the same family as their corresponding distributions in the generative process. Accordingly, we will define the variational distributions to be in the same family. However, the main difference between BNP models and parametric models is that BNP models contain an infinite number of hidden variables. These cannot be completely represented in the variational distribution as this would require optimizing an infinite number of variational parameters. We solve this problem by truncating the variational distribution (Blei and Jordan, 2006). At the corpus level, we truncate at $K$, fitting posteriors to $K$ breaking points, $K$ topics, and allowing the topic pointer variables to take on one of $K$ values. At the document level we truncate at $T$, fitting $T$ breaking proportions, $T$ topic pointers, and letting the topic assignment variable take on one of $T$ values. Thus the variational family is,

$$q(\beta, v, z, \pi) = \left(\prod_{k=1}^{K} q(\beta_k \mid \lambda_k) q(v_k \mid a_k)\right) \left(\prod_{d=1}^{D} \prod_{i=1}^{T} q(c_{di} \mid \zeta_{di}) q(\pi_{di} \mid \gamma_{di}) \prod_{n=1}^{N} q(z_{dn} \mid \phi_{dn})\right)$$

We emphasize that this is not a finite model. With truncation levels set high enough, the variational posterior will use as many topics as the posterior needs, but will not necessarily use all $K$ topics to explain the observations. (If $K$ is set too small then the truncated variational distribution will use all of the topics, but this problem can be easily diagnosed and corrected.) Further, a particular advantage of this two-level stick-breaking distribution is that the document truncation $T$ can be much smaller than $K$. Though there may be hundreds of topics in a large corpus, we expect each document will only exhibit a small subset of them.

*Stochastic variational inference for HDP topic models.* From the complete conditionals, batch variational inference proceeds by updating each variational parameter using the expectation of its conditional distribution's natural parameter. In stochastic inference, we sample a data point, update its local parameters as for batch inference, and then update the global variables.

To update the global topic parameters, we again form intermediate topics with the sampled document's optimized local parameters,

$$\hat{\lambda}_k = \eta + D \sum_{i=1}^{T} \mathbb{E}_q[c_{di}^k] \sum_{n=1}^{N} \mathbb{E}_q[z_{dn}^i] w_{dn}.$$

We then update the global variational parameters by taking a step in the direction of the stochastic natural gradient

$$\lambda^{(t+1)} = (1 - \rho_t)\lambda^{(t)} + \rho_t \hat{\lambda}_k.$$

This mirrors the update for LDA.

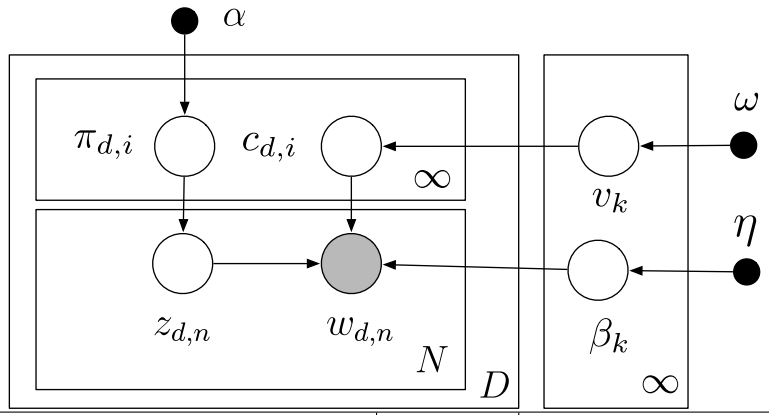
The other global variables in the HDP are the corpus-level breaking proportions $v_k$, each of which is associated with a set of beta parameters $a_k = (a_k^{(1)}, a_k^{(2)})$ for its variational distribution. Using the same randomly selected document and optimized variational parameters as above, first construct the two-dimensional vector

$$\hat{a}_k = \left(1 + D \sum_{i=1}^{T} \mathbb{E}_q[c_{di}^k], \omega + D \sum_{i=1}^{T} \sum_{j=k+1}^{K} \mathbb{E}_q[c_{di}^j]\right).$$

Then, update the parameters

$$a_k^{(t+1)} = (1 - \rho_t) a_k^{(t)} + \rho_t \hat{a}_k.$$

Note that we use the truncations $K$ and $T$. Figure 8 summarizes the complete conditionals, variational parameters, and relevant expectations for the full algorithm. Figure 9 gives the stochastic variational inference algorithm for the HDP topic model.




| Var | Type | Conditional | Param | Relevant expectation |
|---|---|---|---|---|
| $z_{dn}$ | Multinomial | $\log \sigma_i(\pi_d) + \sum_{k=1}^{\infty} c_{di}^k \log \beta_{k,w_{dn}}$ | $\phi_{dn}$ | $\mathbb{E}[Z_{dn}^i] = \phi_{dn}^i$ |
| $\pi_{di}$ | Beta | $(1 + \sum_{n=1}^{N} z_{dn}^i,\ \alpha + \sum_{n=1}^{N} \sum_{j=i+1}^{\infty} z_{dn}^j)$ | $(\gamma_{di}^{(1)}, \gamma_{di}^{(2)})$ | (Expectations are similar to those for $v_k$.) |
| $c_{di}$ | Multinomial | $\log \sigma_k(V) + \sum_{n=1}^{N} z_{dn}^i \log \beta_{k,w_{dn}}$ | $\zeta_{di}$ | $\mathbb{E}[c_{di}^k] = \zeta_{di}^k$ |
| $v_k$ | Beta | $(1 + \sum_d \sum_i c_{di}^k,\ \omega + \sum_d \sum_i \sum_{\ell=k+1}^{\infty} c_{di}^{\ell})$ | $(a_k^{(1)}, a_k^{(2)})$ | $\mathbb{E}[\log V_k] = \Psi(a_k) - \Psi(a_k + b_k)$<br>$\mathbb{E}[\log(1 - V_k)] = \Psi(b_k) - \Psi(a_k + b_k)$<br>$\mathbb{E}[\log \sigma_k(V)] = \mathbb{E}[\log V_k] + \sum_{\ell=1}^{k-1} \mathbb{E}[\log(1 - V_\ell)]$ |
| $\beta_k$ | Dirichlet | $\eta + \sum_{d=1}^{D} \sum_{i=1}^{\infty} c_{di}^k \sum_{n=1}^{N} z_{dn}^i w_{dn}$ | $\lambda_k$ | $\mathbb{E}[\log \beta_{kv}] = \Psi(\lambda_{kv}) - \Psi(\sum_{v'} \lambda_{kv'})$ |

Figure 8: A graphical model for the HDP topic model, and a summary of its variational inference algorithm.

Initialize $\lambda^{(0)}$ randomly. Set $a^{(0)} = 1$ and $b^{(0)} = \omega$.
Set the step-size schedule $\rho_t$ appropriately.
**repeat**
  Sample a document $w_d$ uniformly from the data set.
  For $i \in \{1,\ldots,T\}$ initialize

$$\zeta_{di}^k \propto \exp\{\textstyle\sum_{n=1}^N \mathbb{E}[\log \beta_{k,w_{dn}}]\}, k \in \{1,\ldots,K\}.$$

  For $n \in \{1,\ldots,N\}$ initialize

$$\phi_{dn}^i \propto \exp\left\{\textstyle\sum_{k=1}^K \zeta_{di}^k \mathbb{E}[\log \beta_{k,w_{dn}}]\right\}, i \in \{1,\ldots,T\}.$$

  **repeat**
    For $i \in \{1,\ldots,T\}$ set

$$\begin{aligned}
\gamma_{di}^{(1)} &= 1 + \textstyle\sum_{n=1}^N \phi_{dn}^i, \\
\gamma_{di}^{(2)} &= \alpha + \textstyle\sum_{n=1}^N \sum_{j=i+1}^T \phi_{dn}^j \\
\zeta_{di}^k &\propto \exp\left\{\mathbb{E}[\log \sigma_k(V)] + \textstyle\sum_{n=1}^N \phi_{dn}^i \mathbb{E}[\log \beta_{k,w_{dn}}]\right\}, k \in \{1,\ldots,K\}.
\end{aligned}$$

    For $n \in \{1,\ldots,N\}$ set

$$\phi_{dn}^i \propto \exp\left\{\mathbb{E}[\log \sigma_i(\pi_d)] + \textstyle\sum_{k=1}^K \zeta_{di}^k \mathbb{E}[\log \beta_{k,w_{dn}}]\right\}, i \in \{1,\ldots,T\}.$$

  **until** local parameters converge.
  For $k \in \{1,\ldots,K\}$ set intermediate topics

$$\begin{aligned}
\hat{\lambda}_{kv} &= \eta + D \textstyle\sum_{i=1}^T \zeta_{di}^k \sum_{n=1}^N \phi_{dn}^i w_{dn}, \\
\hat{a}_k &= 1 + D \textstyle\sum_{i=1}^T \zeta_{di}^k, \\
\hat{b}_k &= \omega + D \textstyle\sum_{i=1}^T \sum_{\ell=k+1}^K \zeta_{di}^\ell.
\end{aligned}$$

  Set

$$\begin{aligned}
\lambda^{(t)} &= (1-\rho_t)\lambda^{(t-1)} + \rho_t \hat{\lambda}, \\
a^{(t)} &= (1-\rho_t) a^{(t-1)} + \rho_t \hat{a}, \\
b^{(t)} &= (1-\rho_t) b^{(t-1)} + \rho_t \hat{b}.
\end{aligned}$$

**until** forever

Figure 9: Stochastic variational inference for the HDP topic model. The corpus-level truncation is $K$; the document-level truncation as $T$. Relevant expectations are found in Figure 8.

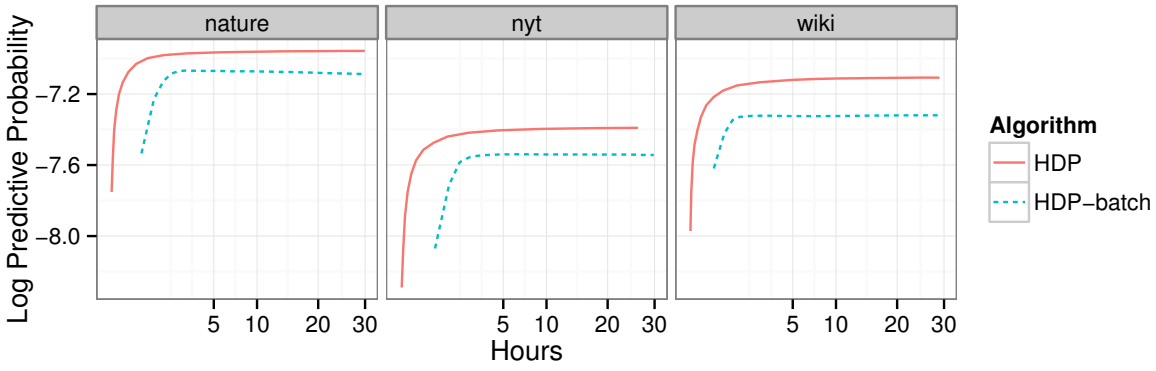



Figure 10: The per-word predictive log likelihood for an HDP model on three large corpora. (Time is on the square root scale.) As for LDA, stochastic variational inference on the full data converges faster and to a better place than batch variational inference on a reasonably sized subset. Section 4 gives the details of our empirical study.

*Stochastic inference versus batch inference for the HDP.* Figure 10 illustrates the performance of the HDP topic model on the same three large collections as in Figure 7. As with LDA, stochastic variational inference for the HDP converges faster and to a better model.

## 4. Empirical Study

In this section we study the empirical performance and effectiveness of stochastic variational inference for latent Dirichlet allocation (LDA) and the hierarchical Dirichlet process (HDP) topic model. With these algorithms, we can apply and compare these models with very large collections of documents. We also investigate how the forgetting rate $\kappa$ and mini-batch size $S$ influence the algorithms. Finally, we compare stochastic variational inference to the traditional batch variational inference algorithm.[9]

*Data.* We evaluated our algorithms on three collections of documents. For each collection, we computed a vocabulary by removing stop words, rare words, and very frequent words. The data are as follows.

- *Nature*: This collection contains 350,000 documents from the journal *Nature* (spanning the years 1869–2008). After processing, it contains 58M observed words from a vocabulary of 4,200 terms.

- *New York Times*: This collection contains 1.8M documents from the *New York Times* (spanning the years 1987–2007). After processing, this data contains 461M observed words from a vocabulary of 8,000 terms.

9. We implemented all algorithms in Python using the NumPy and SciPy packages, making the implementations as similar as possible. Links to these implementations are available on the web at `http://www.cs.princeton.edu/~blei/topicmodeling.html`.

- *Wikipedia*: This collections contains 3.8M documents from Wikipedia. After processing, it contains 482M observed words from a vocabulary of 7,700 terms.

For each collection, we set aside a test set of $10,000$ documents for evaluating model fitness; these test sets were not given to the algorithms for training.

*Evaluating model fitness.* We evaluate how well a model fits the data with the predictive distribution (Geisser, 1975). We are given a corpus and estimate its topics. We then are given part of a test document, which we use to estimate that document's topic proportions. Combining those topic proportions with the topics, we form a predictive distribution over the vocabulary. Under this predictive distribution, a better model will assign higher probability to the held-out words.

In more detail, we divide each test document's words $w$ into a set of observed words $w_{\text{obs}}$ and held-out words $w_{\text{ho}}$, keeping the sets of unique words in $w_{\text{obs}}$ and $w_{\text{ho}}$ disjoint. We approximate the posterior distribution of topics $\beta$ implied by the training data $\mathcal{D}$, and then use that approximate posterior to estimate the predictive distribution $p(w_{\text{new}} \mid w_{\text{obs}}, \mathcal{D})$ of a new held-out word $w_{\text{new}}$ from the test document. Finally, we evaluate the log probability of the words in $w_{\text{ho}}$ under this distribution.

This metric was used in Teh et al. (2007) and Asuncion et al. (2009). Unlike previous methods, like held-out perplexity (Blei et al., 2003), evaluating the predictive distribution avoids comparing bounds or forming approximations of the evaluation metric. It rewards a good predictive distribution, however it is computed.

Operationally, we use the training data to compute variational Dirichlet parameters for the topics. We then use these parameters with the observed test words $w_{\text{obs}}$ to compute the variational distribution of the topic proportions. Taking the inner product of the expected topics and the expected topic proportions gives the predictive distribution.

To see this is a valid approximation, note the following for a $K$-topic LDA model,

$$\begin{aligned} p(w_{\text{new}} \mid \mathcal{D}, w_{\text{obs}}) &= \int\int \left(\textstyle\sum_{k=1}^{K} \theta_k \beta_{k,w_{\text{new}}}\right) p(\theta \mid w_{\text{obs}}, \beta) p(\beta \mid \mathcal{D}) d\theta d\beta \\ &\approx \int\int \left(\textstyle\sum_{k=1}^{K} \theta_k \beta_{k w_{\text{new}}}\right) q(\theta) q(\beta) d\theta d\beta \\ &= \textstyle\sum_{k=1}^{K} \mathbb{E}_q[\theta_k] \mathbb{E}_q[\beta_{k,w_{\text{new}}}], \end{aligned}$$

where $q(\beta)$ depends on the training data $\mathcal{D}$ and $q(\theta)$ depends on $q(\beta)$ and $w_{\text{obs}}$. The metric independently evaluates each held out word under this distribution. In the HDP, the reasoning is identical. The differences are that the topic proportions are computed via the two-level variational stick-breaking distribution and $K$ is the truncation level of the approximate posterior.

*Setting the learning rate.* Stochastic variational inference introduces several parameters in setting the learning rate schedule (see Equation 26). The forgetting rate $\kappa \in (0.5, 1]$ controls how quickly old information is forgotten; the delay $\tau \geq 0$ down-weights early iterations; and the mini-batch size $S$ is how many documents are subsampled and analyzed in each iteration. Although stochastic variational inference algorithm converges to a stationary point for any valid $\kappa$, $\tau$, and $S$, the quality of this stationary point and the speed of convergence may depend on how these parameters are set.

We set $\tau = 1$ and explored the following forgetting rates and minibatch sizes:[10]

- Forgetting rate $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$

10. We also explored various values of the delay $\tau$, but found that the algorithms were not sensitive. To make this presentaton simpler, we fixed $\tau = 1$ in our report of the empirical study.

| | *Nature* | *New York Times* | *Wikipedia* |
|---|---|---|---|
| LDA 25 | -7.24 | -7.73 | -7.44 |
| LDA 50 | -7.23 | -7.68 | -7.43 |
| LDA 100 | -7.26 | -7.66 | -7.41 |
| LDA 200 | -7.50 | -7.78 | -7.64 |
| LDA 300 | -7.86 | -7.98 | -7.74 |
| HDP | **-6.97** | **-7.38** | **-7.07** |

Figure 11: Stochastic inference lets us compare performance on several large data sets. We fixed the forgetting rate $\kappa = 0.9$ and the batch size to 500 documents. We find that LDA is sensitive to the number of topics; the HDP gives consistently better predictive performance. Traditional variational inference (on subsets of each corpus) did not perform as well as stochastic inference.

- Minibatch size $S \in \{10, 50, 100, 500, 1000\}$

We periodically paused each run to compute predictive likelihoods from the test data.

*Results on LDA and HDP topic models.* We studied LDA and the HDP. In LDA, we varied the number of topics $K$ to be 25, 50, 100, 200 and 300; we set the Dirichlet hyperparameters $\alpha = 1/K$. In the HDP, we set both concentration parameters $\gamma$ and $\alpha$ equal to 1; we set the top-level truncation $K = 300$ and the second level truncation $T = 20$. (Here $T \ll K$ because we do not expect documents to exhibit very many unique topics.) In both models, we set the topic Dirichlet parameter $\eta = 0.01$. Figure 1 shows example topics from the HDP (on *New York Times* and *Nature*).

Figure 11 gives the average predictive log likelihood for both models. We report the value for a forgetting rate $\kappa = 0.9$ and a batch size of 500. Stochastic inference lets us perform a large-scale comparison of these models. The HDP gives consistently better performance. For larger numbers of topics, LDA overfits the data. As the modeling assumptions promise, the HDP stays robust to overfitting.[11] That the HDP outperforms LDA regardless of how many topics LDA uses may be due in part to the additional modeling flexibility given by the corpus breaking proportions $v$; these variables give the HDP the ability to say that certain topics are a priori more likely to appear than others, whereas the exchangeable Dirichlet prior used in LDA assumes that all topics are equally common.

We now turn to the sensitivity of stochastic inference to its learning parameters. First, we consider the HDP (the algorithm presented in Figure 9). We fixed the batch size to 500 and explored the forgetting rate.[12] Figure 12 shows the results on all three corpora. All three fits were sensitive to the forgetting rate; we see that a higher value (i.e., close to one) leads to convergence to a better local optimum.

Fixing the forgetting rate to 0.9, we explored various mini-batch sizes. Figure 13 shows the results on all three corpora. Batch sizes that are too small (e.g., ten documents) can affect perfor-

11. Though not illustrated, we note that using the traditional measure of fit, held-out perplexity, does *not* reveal this overfitting (though the HDP still outperforms LDA with that metric as well). We feel that the predictive distribution is a better metric for model fitness.

12. We fit distributions using the entire grid of parameters described above. However, to simplify presenting results we will hold one of the parameters fixed and vary the other.

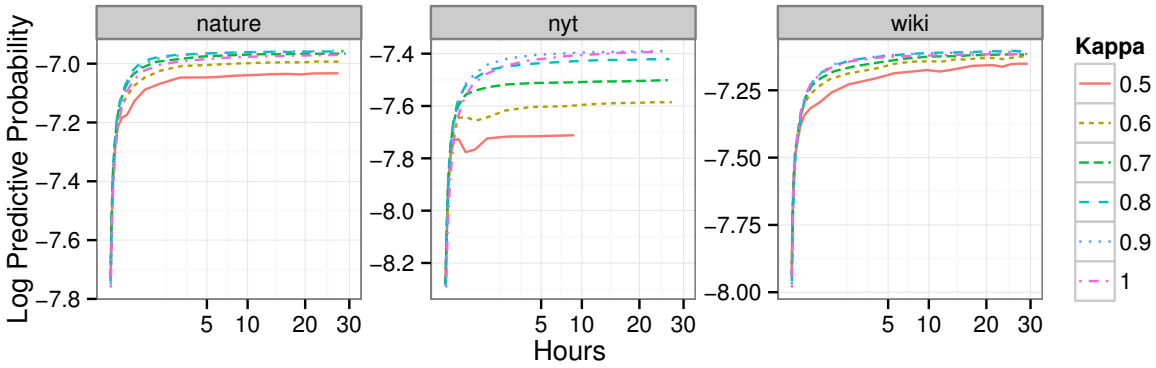



Figure 12: HDP inference: Holding the batch size fixed at 500, we varied the forgetting rate $\kappa$. Slower forgetting rates are preferred.

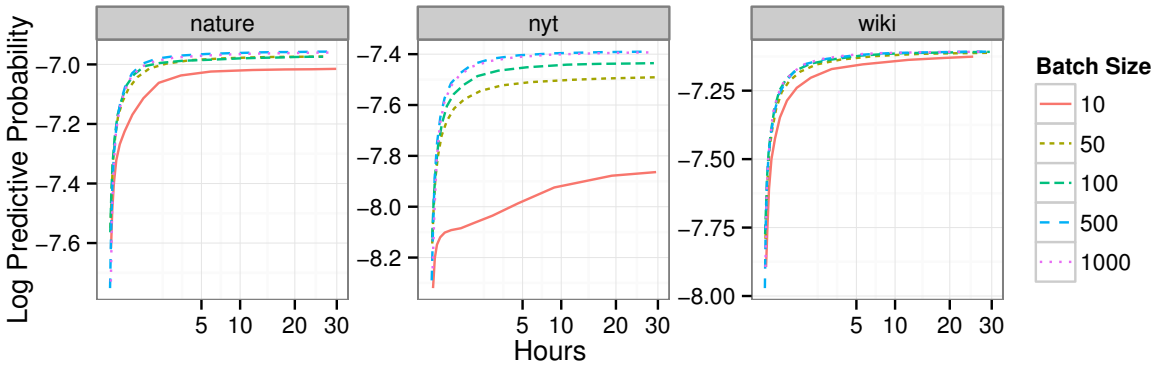



Figure 13: HDP inference: Holding the forgetting rate $\kappa$ fixed at 0.9, we varied the batch size. Batch sizes may be set too small (e.g., ten documents) but the difference in performance is small once set high enough.

mance; larger batch sizes are preferred. That said, there was not a big difference between batch sizes of 500 and 1,000. The *New York Times* corpus was most sensitive to batch size; the *Wikipedia* corpus was least sensitive.

Figure 14 and Figure 15 illustrate LDA's sensitivity to the forgetting rate and batch size, respectively. Again, we find that large learning rates and batch sizes perform well.

## 5. Discussion

We developed stochastic variational inference, a scalable variational inference algorithm that lets us analyze massive data sets with complex probabilistic models. The main idea is to use stochastic optimization to optimize the variational objective, following noisy estimates of the natural gradient where the noise arises by repeatedly subsampling the data. We illustrated this approach with two probabilistic topic models, latent Dirichlet allocation and the hierarchical Dirichlet process topic

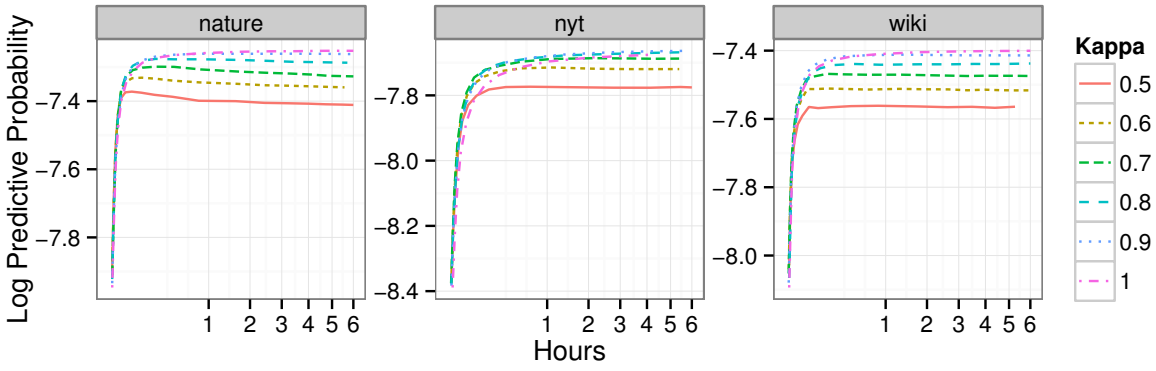



Figure 14: 100-topic LDA inference: Holding the batch size fixed at 500, we varied the forgetting rate $\kappa$. Slower forgetting rates are preferred.

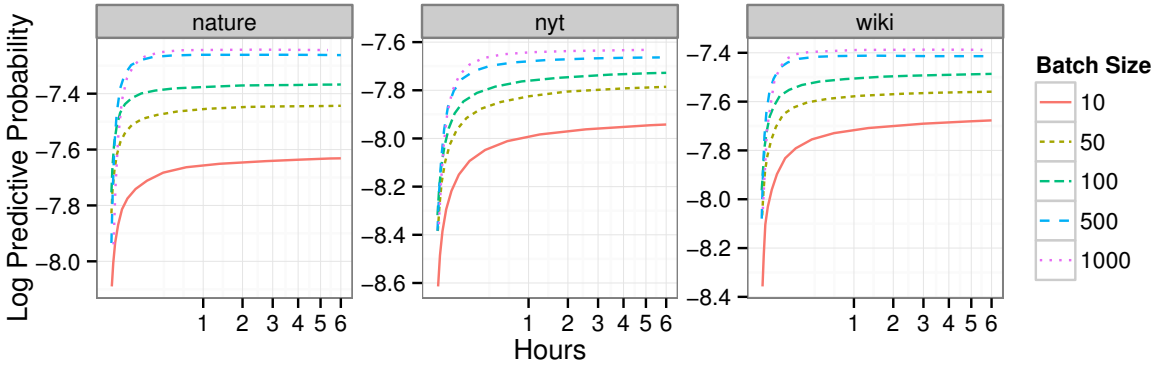



Figure 15: 100-topic LDA inference: Holding the learning rate $\kappa$ fixed at 0.9, we varied the batch size. Bigger batch sizes are preferred.

model. With stochastic variational inference, we can easily apply topic modeling to collections of millions of documents. More importantly, this algorithm generalizes to many settings.

Since developing this algorithm, we have improved on stochastic inference in a number of ways. In Gopalan et al. (2012), we applied it to the mixed-membership stochastic blockmodel for uncovering overlapping communities in large social networks. This required sampling non-uniformly from the data and adjusting the noisy gradient accordingly. In Mimno et al. (2012), we developed a variant of stochastic inference that combines MCMC for the local updates with stochastic optimization for the global updates. In topic modeling this allows for efficient and sparse updates. Finally, in Ranganath et al. (2013), we developed adaptive learning rates for stochastic inference. These outperform preset learning-rate schedules and require less hand-tuning by the user.

Stochastic variational inference opens the door to several promising research directions.

We developed our algorithm with conjugate exponential family models. This class of models is expressive, but nonconjugate models—models where a richer prior is used at the expense of

mathematical convenience—have expanded the suite of probabilistic tools at our disposal. For example, nonconjugate models can capture correlations between topics (Blei and Lafferty, 2007) or topics changing over time (Blei and Lafferty, 2006; Wang et al., 2008), and the general algorithm presented here cannot be used in these settings. (In other work, Paisley et al., 2012b developed a stochastic variational inference algorithm for a specific nonconjugate Bayesian nonparametric model.) Recent research has developed general methods for non-conjugate models (Knowles and Minka, 2011; Gershman et al., 2012; Paisley et al., 2012a; Wang and Blei, 2013). Can these be scaled up with stochastic optimization?

We developed our algorithm with mean-field variational inference and closed form coordinate updates. Another promising direction is to use stochastic optimization to scale up recent advances in variational inference, moving beyond closed form updates and fully factorized approximate posteriors. As one example, collapsed variational inference (Teh et al., 2006b, 2007) marginalizes out some of the hidden variables, trading simple closed-form updates for a lower-dimensional posterior. As another example, structured variational distributions relax the mean-field approximation, letting us better approximate complex posteriors such as those arising in time-series models (Ghahramani and Jordan, 1997; Blei and Lafferty, 2006).

Finally, our algorithm lets us potentially connect innovations in stochastic optimization to better methods for approximate posterior inference. Wahabzada and Kersting (2011) and Gopalan et al. (2012) sample from data non-uniformly to better focus on more informative data points. We might also consider data whose distribution changes over time, such as when we want to model an infinite stream of data but to "forget" data from the far past in a current estimate of the model. Or we can study and try to improve our estimates of the gradient. Are there ways to reduce its variance, but maintain its unbiasedness?

## Acknowledgments

David M. Blei is supported by NSF CAREER IIS-0745520, NSF BIGDATA IIS-1247664, NSF NEURO IIS-1009542, ONR N00014-11-1-0651, and the Alfred P. Sloan Foundation. The authors are grateful to John Duchi, Sean Gerrish, Lauren Hannah, Neil Lawrence, Jon McAuliffe, and Rajesh Ranganath for comments and discussions.

## Appendix A.

In Section 2, we assumed that we can calculate $p(\beta|x,z)$, the conditional distribution of the global hidden variables $\beta$ given the local hidden variables $z$ and observed variables $x$. In this appendix, we show how to do stochastic variational inference under the weaker assumption that we can break the global parameter vector $\beta$ into a set of $K$ subvectors $\beta_{1:K}$ such that each conditional distribution $p(\beta_k|x,z,\beta_{-k})$ is in a tractable exponential family:

$$p(\beta_k|x,z,\beta_{-k}) = h(\beta_k)\exp\{\eta_g(x,z,\beta_{-k},\alpha)^\top t(\beta_k) - a_g(\eta_g(x,z,\beta_{-k},\alpha))\}.$$

We will assign each $\beta_k$ an independent variational distribution so that

$$q(z,\beta) = (\textstyle\prod_{n,j} q(z_{n,j}))\prod_k q(\beta_k).$$

We choose each $q(\beta_k)$ to be in the same exponential family as the complete conditional $p(\beta_k|x,z,\beta_{-k})$,

$$q(\beta_k) = h(\beta_k)\exp\{\lambda_k^\top t(\beta_k) - a_g(\lambda_k)\}.$$

We overload $h(\cdot)$, $t(\cdot)$, and $a(\cdot)$ so that, for example, $q(\beta_k) = p(\beta_k|x,z,\beta_{-k})$ when $\lambda_k = \eta_g(x,z,\beta_{-k},\alpha)$.

The natural parameter $\eta_g(x,z,\beta_{-k},\alpha)$ decomposes into two terms,

$$\eta_g(x,z,\beta_{-k},\alpha) = \eta_g(\beta_{-k},\alpha) + \sum_n \eta_g(x_n,z_n,\beta_{-k},\alpha).$$

The first depends only on the global parameters $\beta_{-k}$ and the hyperparameters $\alpha$; the second is a sum of $N$ terms that depend on $\beta_{-k}$, $\alpha$, and a single local context $(x_n,z_n)$.

Proceeding as in Section 2, we will derive the natural gradient of the ELBO implied by this model and choice of variational distribution. Focusing on a particular $\beta_k$, we can write the ELBO as

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_q[\log p(\beta_k|x,z,\beta_{-k})] - \mathbb{E}_q[\log q(\beta_k)] + \text{const.} \\ &= (\mathbb{E}_q[\eta_g(x,z,\beta_{-k},\alpha)] - \lambda_k)^\top \nabla_{\lambda_k} a_g(\lambda_k) + a_g(\lambda_k) + \text{const.}\end{aligned}$$

The gradient of $\mathcal{L}$ with respect to $\lambda_k$ is then

$$\nabla_{\lambda_k}\mathcal{L} = \nabla^2_{\lambda_k} a_g(\lambda_k)(\mathbb{E}_q[\eta_g(x,z,\beta_{-k},\alpha)] - \lambda_k),$$

and the natural gradient of $\mathcal{L}$ with respect to $\lambda_k$ is

$$\begin{aligned}\hat{\nabla}_{\lambda_k}\mathcal{L} &= \mathbb{E}_q[\eta_g(x,z,\beta_{-k},\alpha)] - \lambda_k \\ &= -\lambda_k + \mathbb{E}_q[\eta_g(\beta_{-k},\alpha)] + \sum_n \mathbb{E}_q[\eta_g(x_n,z_n,\beta_{-k},\alpha)].\end{aligned}$$

Randomly sampling a local context $(x_i,z_i)$ yields a noisy (but unbiased) estimate of the natural gradient,

$$\hat{\nabla}_{\lambda_k}\mathcal{L}_i = -\lambda_k + \mathbb{E}_q[\eta_g(\beta_{-k},\alpha)] + N\mathbb{E}_q[\eta_g(x_i,z_i,\beta_{-k},\alpha)] \equiv -\lambda_k + \hat{\lambda}_k.$$

We can use this noisy natural gradient exactly as in Section 2. For each update $t$, we sample a context $(x_t,z_t)$, optimize the local variational parameters $\phi_t$ by repeatedly applying equation Equation 16, and take a step of size $\rho_t = (t+\tau)^{-\kappa}$ in the direction of the noisy natural gradient:

$$\lambda_k^{(t)} = (1-\rho_t)\lambda_k^{(t-1)} + \rho_t\hat{\lambda}_k \tag{34}$$

Note that the update in Equation 34 depends only on $\lambda^{(t-1)}$; we compute all elements of $\lambda^{(t)}$ simultaneously, whereas in a batch coordinate ascent algorithm $\lambda_k^{(t)}$ could depend on $\lambda_{1:k-1}^{(t)}$.

## References

A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. Smola. Scalable inference in latent variable models. In *Web Search and Data Mining*, New York, NY, USA, 2012.

S. Amari. Differential geometry of curved exponential families-curvatures and information loss. *The Annals of Statistics*, 10(2):357–385, 1982.

S. Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

C. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174, 1974.

A. Asuncion, M. Welling, P. Smyth, and Y. Teh. On smoothing and inference for topic models. In *Uncertainty in Artificial Intelligence*, 2009.

H. Attias. Inferring parameters and structure of latent variable models by variational bayes. In *Uncertainty in Artificial Intelligence*, 1999.

H. Attias. A variational Bayesian framework for graphical models. In *Neural Information Processing Systems*, 2000.

J. Bernardo and A. Smith. *Bayesian Theory*. John Wiley & Sons Ltd., Chichester, 1994.

C. Bishop. *Pattern Recognition and Machine Learning*. Springer New York., 2006.

C. Bishop, D. Spiegelhalter, and J. Winn. VIBES: A variational inference engine for Bayesian networks. In *Neural Information Processing Systems*. Cambridge, MA, 2003.

D. Blackwell and J. MacQueen. Ferguson distributions via Pólya urn schemes. *The Annals of Statistics*, 1(2):353–355, 1973.

D. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

D. Blei and M. Jordan. Variational inference for Dirichlet process mixtures. *Journal of Bayesian Analysis*, 1(1):121–144, 2006.

D. Blei and J. Lafferty. Dynamic topic models. In *International Conference on Machine Learning*, pages 113–120, 2006.

D. Blei and J. Lafferty. A correlated topic model of Science. *Annals of Applied Statistics*, 1(1): 17–35, 2007.

D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

L. Bottou. On-line learning and stochastic approximations. In *On-line Learning in Neural Networks*, pages 9–42. Cambridge University Press, 1998.

L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, pages 146–168. Springer, 2003.

L. Bottou and O. Bousquet. Learning using large datasets. In *Mining Massive Datasets for Security*. IOS Press, 2008.

Olivier Cappé and Eric Moulines. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613, 2009.

M. Collins, S. Dasgupta, and R. Schapire. A generalization of principal component analysis to the exponential family. In *Neural Information Processing Systems*, 2002.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

M. Do Carmo. *Riemannian Geometry*. Birkhäuser, 1992.

A. Doucet, N. De Freitas, and N. Gordon. *An introduction to sequential Monte Carlo methods*. Springer, 2001.

E. Erosheva. Bayesian estimation of the grade of membership model. *Bayesian Statistics*, 7:501–510, 2003.

M. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

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

S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32:41–62, 1998.

E. Fox, E. Sudderth, M. Jordan, and A. Willsky. An HDP-HMM for systems with state persistence. In *International Conference on Machine Learning*, 2008.

E. Fox, E. Sudderth, M. Jordan, and A. Willsky. Bayesian nonparametric inference of switching dynamic linear models. *IEEE Transactions on Signal Processing*, 59(4):1569–1585, 2011a.

E. Fox, E. Sudderth, M. Jordan, and A. Willsky. A sticky HDP-HMM with application to speaker diarization. *Annals of Applied Statistics*, 5(2A):1020–1056, 2011b.

S. Geisser. The predictive sample reuse method with applications. *Journal of the American Statistical Association*, 70:320–328, 1975.

A. Gelfand and A. Smith. Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409, 1990.

A. Gelman and J. Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press, 2007.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

S. Gershman and D. Blei. A tutorial on Bayesian nonparametric models. *Journal of Mathematical Psychology*, 56:1–12, 2012.

S. Gershman, M. Hoffman, and D. Blei. Nonparametric variational inference. In *International Conference on Machine Learning*, 2012.

Z. Ghahramani and M. Beal. Variational inference for Bayesian mixtures of factor analysers. In *Neural Information Processing Systems*, 2000.

Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In *Neural Information Processing Systems*, pages 507–513, 2001.

Z. Ghahramani and M. Jordan. Factorial hidden Markov models. *Machine Learning*, 31(1), 1997.

M. Girolami and S. Rogers. Variational Bayesian multinomial probit regression with Gaussian process priors. *Neural Computation*, 18(8), 2006.

P. Gopalan, D. Mimno, S. Gerrish, M. Freedman, and D. Blei. Scalable inference of overlapping communities. In *Neural Information Processing Systems*, 2012.

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

W. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

G. Hinton and D. Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Computational Learning Theory*, pages 5–13. ACM, 1993.

N. Hjort, C. Holmes, P. Muller, and S. Walker, editors. *Bayesian Nonparametrics*. Cambridge University Press, 2010.

M. Hoffman, D. Blei, and F. Bach. On-line learning for latent Dirichlet allocation. In *Neural Information Processing Systems*, 2010a.

M. Hoffman, D. Blei, and P. Cook. Bayesian nonparametric matrix factorization for recorded music. In *International Conference on Machine Learning*, 2010b.

A. Honkela, M. Tornio, T. Raiko, and J. Karhunen. Natural conjugate gradient in variational inference. In *Neural Information Processing Systems*, 2008.

T. Jaakkola. *Variational Methods for Inference and Estimation in Graphical Models*. PhD thesis, Massachusetts Institute of Technology, 1997.

M. Jordan, editor. *Learning in Graphical Models*. MIT Press, Cambridge, MA, 1999.

M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

R. Kalman. A new approach to linear filtering and prediction problems a new approach to linear filtering and prediction problems,". *Transaction of the AMSE: Journal of Basic Engineering*, 82: 35–45, 1960.

D. Knowles and T. Minka. Non-conjugate variational message passing for multinomial and binary regression. In *Neural Information Processing Systems*, 2011.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

S. Kullback and R.A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

P. Liang, M. Jordan, and D. Klein. Learning semantic correspondences with less supervision. In *Association of Computational Linguisitics*, 2009.

J. Mairal, J. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

J. Maritz and T. Lwin. *Empirical Bayes Methods*. Monographs on Statistics and Applied Probability. Chapman & Hall, London, 1989.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. London: Chapman and Hall, 1989.

N. Metropolis, A. Rosenbluth, M. Rosenbluth, M. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

D. Mimno, M. Hoffman, and D. Blei. Sparse stochastic inference for latent Dirichlet allocation. In *International Conference on Machine Learning*, 2012.

T. Minka and J. Lafferty. Expectation-propagation for the generative aspect model. In *Uncertainty in Artificial Intelligence (UAI)*, 2002.

K. Murphy. *Machine Learning: A Probabilistic Approach*. MIT Press, 2012.

R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. MIT Press, 1999.

D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10:1801–1828, 2009.

J. Paisley and L. Carin. Nonparametric factor analysis with beta process priors. In *International Conference on Machine Learning*, 2009.

J. Paisley, D. Blei, and M. Jordan. Variational Bayesian inference with stochastic search. In *International Conference on Machine Learning*, 2012a.

J. Paisley, C. Wang, and D. Blei. The discrete infinite logistic normal distribution. *Bayesian Analysis*, 7(2):235–272, 2012b.

J. Paisley, C. Wang, D. Blei, and M. Jordan. Nested hierarchical Dirichlet processes. *arXiv preprint arXiv:1210.6738*, 2012c.

G. Parisi. *Statistical Field Theory*. Perseus Books, 1988.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988. ISBN 1558604790.

C. Peterson and J. Anderson. A mean field theory learning algorithm for neural networks. *Complex Systems*, 1(5):995–1019, 1987.

J. Pitman. *Combinatorial Stochastic Processes*. Lecture Notes for St. Flour Summer School. Springer-Verlag, New York, NY, 2002.

J. Platt, E. Kıcıman, and D. Maltz. Fast variational inference for large-scale internet diagnosis. *Neural Information Processing Systems*, 2008.

L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, 1989.

R. Ranganath, C. Wang, D. Blei, and E. Xing. An adaptive learning rate for stochastic variational inference. In *International Conference on Machine Learning*, 2013.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag, New York, NY, 2004.

R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *International Conference on Machine learning*, pages 880–887, 2008.

M. Sato. Online model selection based on the variational Bayes. *Neural Computation*, 13(7):1649–1681, 2001.

L. Saul and M. Jordan. Exploiting tractable substructures in intractable networks. *Neural Information Processing Systems*, 1996.

L. Saul, T. Jaakkola, and M. Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996.

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

A. Smola and S. Narayanamurthy. An architecture for parallel topic models. In *Very Large Databases*, 2010.

J. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. John Wiley and Sons, 2003.

C. Spearman. "General intelligence," objectively determined and measured. *The American Journal of Psychology*, pages 201–292, 1904.

Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006a.

Y. Teh, D. Newman, and M. Welling. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In *Neural Information Processing Systems*, 2006b.

Y. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for HDP. In *Neural Information Processing Systems*, 2007.

M. Tipping and C. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

M. Wahabzada and K. Kersting. Larger residuals, less work: Active document scheduling for latent dirichlet allocation. In *European Conference on Machine Learning*, 2011.

M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

H. Wallach, D. Mimno, and A. McCallum. Rethinking LDA: Why priors matter. In *Neural Information Processing Systems*. 2009.

C. Wang. Variational Bayesian approach to canonical correlation analysis. *IEEE Transactions on Neural Networks*, 2006.

C. Wang and D. Blei. Variational inference in nonconjugate models. *Journal of Machine Learning Research*, 2013.

C. Wang, D. Blei, and D. Heckerman. Continuous time dynamic topic models. In *Uncertainty in Artificial Intelligence*, 2008.

C. Wang, J. Paisley, and D. Blei. Online variational inference for the hierarchical Dirichlet process. In *Artificial Intelligence and Statistics*, 2011.

S. Waterhouse, D. MacKay, and T. Robinson. Bayesian methods for mixtures of experts. *Neural Information Processing Systems*, pages 351–357, 1996.

M. Welling and Y. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *International Conference on Machine Learning*, 2011.

W. Wiegerinck. Variational approximations between mean field theory and the junction tree algorithm. In *Uncertainty in Artificial Intelligence*, 2000.

E. Xing, M. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. In *Uncertainty in Artificial Intelligence*, 2003.

Journal of Machine Learning Research 14 (2013) 1349-1386 Submitted 2/12; Revised 11/12; Published 5/13

# Multicategory Large-Margin Unified Machines

**Chong Zhang** CHONGZ@LIVE.UNC.EDU
*Department of Statistics and Operations Research*
*The University of North Carolina at Chapel Hill*
*Chapel Hill, NC 27599, USA*

**Yufeng Liu** YFLIU@EMAIL.UNC.EDU
*Department of Statistics and Operations Research*
*Carolina Center for Genome Sciences*
*Department of Biostatistics*
*The University of North Carolina at Chapel Hill*
*Chapel Hill, NC 27599, USA*

**Editor:** Saharon Rosset

## Abstract

Hard and soft classifiers are two important groups of techniques for classification problems. Logistic regression and Support Vector Machines are typical examples of soft and hard classifiers respectively. The essential difference between these two groups is whether one needs to estimate the class conditional probability for the classification task or not. In particular, soft classifiers predict the label based on the obtained class conditional probabilities, while hard classifiers bypass the estimation of probabilities and focus on the decision boundary. In practice, for the goal of accurate classification, it is unclear which one to use in a given situation. To tackle this problem, the Large-margin Unified Machine (LUM) was recently proposed as a unified family to embrace both groups. The LUM family enables one to study the behavior change from soft to hard binary classifiers. For multicategory cases, however, the concept of soft and hard classification becomes less clear. In that case, class probability estimation becomes more involved as it requires estimation of a probability vector. In this paper, we propose a new Multicategory LUM (MLUM) framework to investigate the behavior of soft versus hard classification under multicategory settings. Our theoretical and numerical results help to shed some light on the nature of multicategory classification and its transition behavior from soft to hard classifiers. The numerical results suggest that the proposed tuned MLUM yields very competitive performance.

**Keywords:** hard classification, large-margin, soft classification, support vector machine

## 1. Introduction

Classification problems are commonly seen in practice. When one faces a classification task, there are many possible techniques to choose from. To list a few, logistic regression and Fisher linear discriminant analysis (LDA) are classical classification methods. The Support Vector Machine (SVM, Boser et al., 1992; Cortes and Vapnik, 1995; Wahba, 1999) and Boosting (Freund and Schapire, 1997) are more recent machine learning based large-margin classification tools. Despite some known properties of these methods, a practitioner often needs to face one natural question: which method should one choose to solve the classification problem in hand?

©2013 Chong Zhang and Yufeng Liu.

Wahba (2002) discussed the concept of soft versus hard classification. Soft classifiers estimate the class conditional probabilities and make the decision rule based on the obtained probabilities. Typical examples include logistic regression and LDA. Hard classifiers, on the other hand, focus on estimating the decision boundary without probability estimation. One typical example of hard classifiers is the SVM, which is a well known hard classifier without strong distributional assumptions. Another example of hard classifiers is $\psi-$learning (Shen et al., 2003). When class probability estimation is necessary, one can perform multiple weighted learning for probability estimation of hard classifiers (Wang et al., 2008). For a given problem, the choice between hard and soft classifiers can be difficult. Recently, Liu et al. (2011) proposed a family of large-margin classifiers, namely, the Large-margin Unified Machine (LUM). The LUM family is a rich group of classifiers in the sense that it connects hard and soft classifiers in one spectrum. It provides a natural platform for comparisons between soft and hard classifiers. More importantly, it enables us to observe the performance transition from soft to hard classification.

The existing development on the LUM is limited to the binary case. For multicategory problems, further development is necessary. In particular, probability estimation becomes more challenging as one needs to estimate a probability vector. Furthermore, multicategory consistency is much more involved, especially for hard classifiers. For instance, there are a lot of developments on multicategory SVMs in the literature (Vapnik, 1998; Weston and Watkins, 1999; Crammer et al., 2001; Lee et al., 2004; Wang and Shen, 2007; Liu and Yuan, 2011). Most of them are not consistent when there is no dominating class, that is, the maximum class probability is less than 0.5 (Tewari and Bartlett, 2007; Liu, 2007). Recently, Liu and Yuan (2011) proposed a group of consistent multicategory piecewise linear hinge loss functions, namely a family of reinforced hinge loss functions, which covers the loss by Lee et al. (2004) as a special case. For probability estimation, there are several existing multicategory soft classifiers, such as Adaboost in Boosting (Freund and Schapire, 1997; Zou et al., 2008; Zhu et al., 2009), logistic regression (Lin et al., 2000), proximal SVMs (Tang and Zhang, 2006), and multicategory composite least squares classifiers (Park et al., 2010).

We propose a new group of Multicategory Large-margin Unified Machines (MLUMs) in this paper. Similar to the binary case, the MLUM is a broad family that embraces many of the aforementioned classifiers as special cases. It helps to shed some light on the choice between multicategory soft and hard classifiers, and provide some insights on the behavior change from soft to hard classification methods. Our theoretical studies show that the MLUM is always Fisher consistent, and is able to provide class conditional probability estimation. Moreover, we extend the excess risk concept discussed in Bartlett et al. (2006) to the multicategory case and study its convergence rate. We also propose an efficient tuning procedure for the MLUM family. Our numerical results show that the behaviors of different classifiers vary from setting to setting. In particular, we have the following observations.

- Soft classifiers tend to give more accurate classification results by estimating the conditional class probability when the true probability functions are relatively smooth.

- Hard classifiers bypass the probability estimation and may work better when estimation of the underlying probability functions is challenging, such as the step function.

- When the data are noisy with outliers, soft classifiers tend to be very sensitive and unstable. A MLUM member, in-between hard and soft classifiers, tends to work the best. This was not observed in the binary case (Liu et al., 2011).

Although our observations may not hold for all classification problems, it can help us to understand the classification behaviors better. Furthermore, our numerical results suggest that the performance of the proposed tuned MLUM is very competitive.

The rest of this paper is organized as follows. In Section 2, we give some motivation and introduce the MLUM family. Section 3 explores some statistical properties of the MLUM family. Section 4 addresses the computational aspect of the MLUM. In Section 5, we demonstrate the numerical performance of MLUM via several simulated examples. Section 6 discusses some benchmark examples and one gene expression data set. Some discussion is provided in Section 7. The technical proofs are collected in the appendix.

## 2. Methodology

In this section, we first introduce the background of binary classification, then discuss different ways of generalization to multicategory problems. The notion of soft and hard classification is first reviewed in the binary classification context. Then we propose a MLUM framework which helps us to understand soft versus hard classification in the multicategory setting.

### 2.1 Background on Binary Classification

With a training data set given, one main goal of classification is to build a classifier for us to predict the class label $y$ using the input vector $\mathbf{x}$. Here we assume that the training data are $i.i.d.$ samples from an unknown underlying distribution $D(\mathbf{x}, y)$. In binary classification with $y \in \{\pm 1\}$, we want to estimate a function $f(\mathbf{x}) : \mathbf{R}^d \to \mathbf{R}$ and use $\text{sign}\big(f(\mathbf{x})\big)$ as the classification rule. Because of the sign rule and the class labels $\{\pm 1\}$, the quantity $yf(\mathbf{x})$ indicates whether the classification of a point $(\mathbf{x}, y)$, $\text{sign}(f)$, is correct or not. In particular, we have correct classification if and only if $yf(\mathbf{x}) > 0$. This quantity $yf(\mathbf{x})$ is known as the functional margin in the large-margin classification literature.

Using the functional margin, the theoretical $0-1$ loss can be directly written as $L\big(yf(\mathbf{x})\big) = I\big(yf(\mathbf{x}) \le 0\big)$. Our goal is to find a classification function $f$ such that the expected loss of $f$, denoted by

$$R\big(f(\cdot)\big) = E_{\mathbf{X},Y} L\big(Yf(\mathbf{X})\big), \tag{1}$$

is as small as possible. The infimum of $R\big(f(\cdot)\big)$, denoted by $R^*$, is called the Bayes error. In practice, given a training sample $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ independently drawn from $D$, we want to find a function $f$ in a functional space $H$ that minimizes the empirical loss

$$\min_{f \in H} \frac{1}{n} \sum_{i=1}^{n} L\big(y_i f(\mathbf{x}_i)\big), \tag{2}$$

which can be considered as an empirical approximation to the expected loss (1).

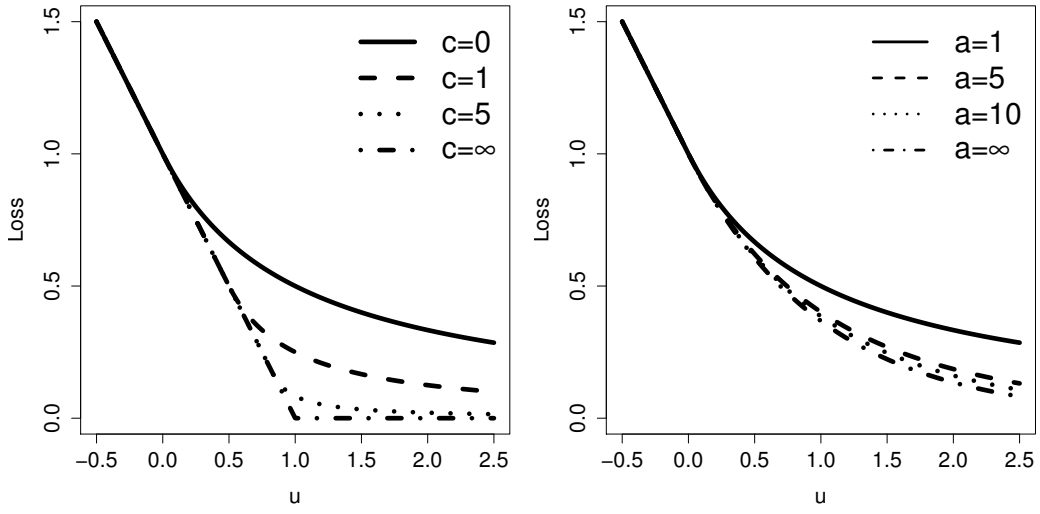
Due to the non-convexity and discontinuity of the $0-1$ loss function $L$, the minimization of the empirical loss (2) is typically NP-hard and difficult to implement in practice. Many surrogate loss functions have been proposed to alleviate this problem. Furthermore, regularization is often used as well to avoid overfitting. In particular, one can replace the $0-1$ loss $L$ with a surrogate loss $V$, and solve the following optimization problem

$$\min_{f \in H} \frac{1}{n} \sum_{i=1}^{n} V\big(y_i f(\mathbf{x}_i)\big) + \lambda J(f), \tag{3}$$

where $J(f)$ is a regularization function of $f$ that helps to prevent overfitting, and $\lambda$ is the tuning parameter that balances the loss function term and the regularization term. Different loss functions correspond to different classification methods. To list a few, AdaBoost employs the exponential loss $V = \exp(-yf)$ (Friedman et al., 2000), SVM (Boser et al., 1992) uses the hinge loss $V = [1-yf]_+$, and logistic regression (Lin et al., 2000; Zhu and Hastie, 2005) uses the deviance loss $V = \log\big(1+\exp(-yf)\big)$. Different methods can be roughly grouped into two categories, namely, soft and hard classifiers. In practice, it is unclear which one to use for a particular problem. To answer this question, Liu et al. (2011) proposed to use the LUM loss function $\ell(\cdot)$ for $V$ in (3), where

$$\ell(u) = \begin{cases} 1-u & \text{if } u < \frac{c}{1+c}, \\ \frac{1}{1+c}\big(\frac{a}{(1+c)u-c+a}\big)^a & \text{if } u \geq \frac{c}{1+c}, \end{cases}$$

with $c \geq 0$ and $a > 0$ being parameters of the LUM family. See Figure 1 for the shape of $\ell(u)$ with a few values of $a$ and $c$ (Liu et al., 2011). Note that the LUM family includes the SVM hinge loss with $c \to \infty$, and the Distance Weighted Discrimination (DWD, Marron et al., 2007) with $c = 1$ and $a = 1$, as special cases. The parameter $c$ is an index of soft versus hard classifiers. In particular, $c = 0$ corresponds to a typical soft classifier, and $c \to \infty$ corresponds to the SVM, a typical hard classifier. Consequently, the LUM connects soft and hard classifiers as a family, and enables one to thoroughly investigate the transition behavior in this spectrum.




(a) a=1 (b) c=0

Figure 1: Plots of several LUM loss functions. On the left panel, we have $a = 1$ and $c = 0, 1, 5, \infty$, and on the right panel we have $c = 0$ and $a = 1, 5, 10, \infty$.

So far, our focus has been on binary methods. In the next section, we briefly introduce some existing methods for multicategory classification problems.

### 2.2 Existing Multicategory Classification Methods

To solve a multicategory problem, a natural and direct way is to implement multiple binary classifiers. For example, one can implement the one-versus-one or one-versus-rest methods. Consider a $k-$category classification problem with the label $y \in \{1,2,\ldots,k\}$. The one-versus-one approach applies a given binary classifier to a problem of $j_1$ versus $j_2$ for all possible $j_1, j_2 \in \{1,2,\ldots,k\}$. Overall, $\frac{k(k-1)}{2}$ binary classifiers are performed, followed by a majority vote step. In particular, one counts the number of votes for each class obtained from the binary classifiers and classifies the point into the class with the maximum number of votes. When there is a tie for the maximum votes among the classes, one can combine the binary probability information to assign labels, if the binary classification probability estimation is available (Wu et al., 2004). If a hard classifier is used for the one-versus-one approach, then the label is randomly chosen among the classes with equal maximum votes. This can be suboptimal. Furthermore, when $k$ is large, the number of binary classifiers needed can be large as well. In addition, some class sizes can be very small. Another similar technique is the one-versus-rest approach. In particular, it relabels the data in the class $j$ as the positive class and the rest as the negative class, for $j \in \{1,2,\ldots,k\}$, and performs a sequence of $k$ binary classification problems. The one-versus-rest approach may have inconsistency for some classifiers such as SVMs (Liu and Yuan, 2011). Hence, it is desirable to have a simultaneous multicategory classifier that considers $k$ classes altogether.

The idea of simultaneous multicategory classifiers is as follows. Consider a $k-$category classification problem. Given an input vector $\mathbf{x}$, we would like to predict its corresponding label $y \in \{1,2,\ldots,k\}$. Instead of using a one-dimensional classification function $f(\mathbf{x})$ as in the binary case, now we employ a $k-$dimensional function vector $\mathbf{f}(\mathbf{x}) = \big(f_1(\mathbf{x}),\ldots,f_k(\mathbf{x})\big)$, and predict the class label $y$ of $\mathbf{x}$ using $\mathrm{argmax}_{j=1,\ldots,k} f_j(\mathbf{x})$. Similar as in (3), we are interested in solving the following optimization problem

$$\min_{f\in H} \frac{1}{n}\sum_{i=1}^{n} V\big(\mathbf{f}(\mathbf{x}_i), y_i\big) + \lambda J(\mathbf{f}), \tag{4}$$

with $V$ being a loss function for a multicategory problem, and $J(\mathbf{f})$ being a regularization term defined for the multicategory problems. To reduce the dimension of the problem and to obtain good theoretical properties, a sum-to-zero constraint, $\sum_{j=1}^{k} f_j(\mathbf{x}) = 0$, is commonly used. As a result, this formulation is equivalent to the binary problem with $k = 2$. With the argmax prediction rule, a sensible loss function $V$ should encourage $f_y$ to be the maximum among $\{f_j;\ j = 1,\ldots,k\}$.

For soft classification in multiclass problems, Zhu and Hastie (2005) used the generalized logistic loss $V = -f_y(\mathbf{x}) + \log(e^{f_1(\mathbf{x})} + \cdots + e^{f_k(\mathbf{x})})$. Tang and Zhang (2006) employed the squared loss $V = (\mathbf{z}-\mathbf{f})^T(\mathbf{z}-\mathbf{f})$, where $\mathbf{z} = -\frac{1}{k-1}(1,1,\ldots,1)^T + \frac{k}{k-1}\mathbf{e}_y$, and $\mathbf{e}_j$ is the vector with 1 at the $j^{th}$ element and 0 elsewhere. Zhu et al. (2009) extended the Adaboost to a multicategory learning method with the exponential loss $V = \exp(-\frac{1}{k}\mathbf{z}^T\mathbf{f})$. In the literature of hard classifiers, there are several ways to extend the binary hinge loss of the SVM to the simultaneous multicategory case. Here we list several commonly used versions with the sum-to-zero constraint:

Loss 1 (Naive hinge loss) $[1 - f_y(\mathbf{x})]_+$;

Loss 2 (Vapnik, 1998) $\sum_{j\neq y}[1 - \big(f_y(\mathbf{x}) - f_j(\mathbf{x})\big)]_+$;

Loss 3 (Crammer et al., 2001; Liu et al., 2005) $\sum_{j\neq y}[1 - \min_j \big(f_y(\mathbf{x}) - f_j(\mathbf{x})\big)]_+$;

Loss 4 (Lee et al., 2004) $\sum_{j\neq y}[1+f_j(\mathbf{x})]_+$.

Losses 1, 2 and 3 are known to be inconsistent (Lee et al., 2004; Liu, 2007; Tewari and Bartlett, 2007). In contrast, Loss 4 is Fisher consistent. Recently, Liu and Yuan (2011) proposed the Reinforced Multicategory Support Vector Machine (RMSVM), which employs a convex combination of the naive hinge loss and Loss 4 by Lee et al. (2004) as follows,

$$V(\mathbf{f}(\mathbf{x}),y)=\gamma[(k-1)-f_y(\mathbf{x})]_+ +(1-\gamma)\sum_{j\neq y}[1+f_j(\mathbf{x})]_+, \tag{5}$$

subject to $\sum_{j=1}^k f_j(\mathbf{x})=0$. Interestingly, the reinforced hinge loss function is Fisher consistent when $0\leq\gamma\leq 1/2$, and includes Loss 4 in Lee et al. (2004) as a special case with $\gamma=0$. Liu and Yuan (2011) showed that the loss function (4) with $\gamma=1/2$ yields the best overall classification performance, among $\gamma\in[0,0.5]$. Inspired by the RMSVM loss formulation, we propose to extend the LUM family to the MLUM in an analogous way, as discussed in the next section.

Next we examine the sum-to-zero constraint $\sum_{j=1}^k f_j(\mathbf{x})=0$ for different multicategory losses. In linear learning, we assume that $f_j(\mathbf{x})=\mathbf{x}^T\beta_j+b_j;\ j=1,\ldots,k$, and the $L_2$ penalty is $J(\mathbf{f})=\frac{1}{2}\sum_{j=1}^k\|\beta_j\|^2$. In kernel learning, we have $f_j=g_{j,\mathcal{H}}+b_j$ and $g_{j,\mathcal{H}};\ j=1,\ldots,k$ belong to some Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$, while $b_j$'s are constants. The $L_2$ penalty for kernel learning is $J(\mathbf{f})=\frac{1}{2}\sum_{j=1}^k\|g_{j,\mathcal{H}}\|^2_{\mathcal{H}}$, where $\|\cdot\|_{\mathcal{H}}$ is the norm in $\mathcal{H}$ introduced by its corresponding kernel. See, for example, Aronszajn (1950) and Wahba (1999) for more details about RKHS. Notice that in both cases, the intercepts $b_j;\ j=1,\ldots,k$ are not penalized. Thus, if we add a constant to all $b_j;\ f=1,\ldots,k$, the prediction on any new instance does not change. Hence, the sum-to-zero constraint helps to obtain unique solutions. The next proposition shows that, if the loss function depends on $\mathbf{f}$ only through its element-wise differences $f_i-f_j;\ i\neq j$ as in Losses 2 and 3, then without the sum-to-zero constraint, the solutions $\{\hat{\beta}_j\}$ in linear learning or $\{\hat{g}_{j,\mathcal{H}}\}$ in kernel learning automatically sum to zero, under the $L_2$ penalty. Note that this phenomenon for the SVM was previously noted by Wu and Liu (2007).

**Proposition 1:** Suppose the loss function $V(\mathbf{f},y)$ depends on $\mathbf{f}$ only through $f_i-f_j;\ i\neq j$, then the solution to (4), using the $L_2$ penalty without the sum-to-zero constraint, satisfies that $\sum_{j=1}^k\hat{\beta}_j=\mathbf{0}$ for linear learning, and $\sum_{j=1}^k\hat{g}_{j,\mathcal{H}}=0$ for RKHS learning.

Notice that the condition in Proposition 1 is satisfied by MSVM Losses 2 and 3 mentioned above. For other loss functions such as Losses 1 and 4, the result does not hold. However, for those loss functions, the sum-to-zero constraint is more essential for theoretical properties such as Fisher consistency. This constraint was also used in many other simultaneous multicategory classification papers, for example, Tang and Zhang (2006), Wang and Shen (2007), and Zhu et al. (2009).

### 2.3 MLUM Family

Soft and hard classifiers have been both studied in the literature of simultaneous multicategory classification. Which one to use in practice remains to be a challenging question. The LUM family is a broad family which embraces both soft and hard classifiers in binary cases, yet no such a convenient platform is available in the multicategory framework. In this paper, we propose a new family of MLUMs to study multicategory problems. In particular, we make use of the idea of reinforced multicategory hinge loss by Liu and Yuan (2011), and propose the following MLUM

loss family,

$$V(\mathbf{f},y) = \gamma\ell\big(f_y(\mathbf{x})\big) + (1-\gamma)\sum_{j\neq y}\ell\big(-f_j(\mathbf{x})\big), \tag{6}$$

under the constraint $\sum_{j=1}^{k} f_j(\mathbf{x}) = 0$, where $\ell(u) : \mathbf{R} \to \mathbf{R}$ is the LUM loss function, and $\gamma \in [0,1]$. When $k = 2$, the MLUM reduces to the binary LUM loss.

The main motivation to use the MLUM loss function (6) is based on the argmax rule for multicategory classification. For a given data point $(\mathbf{x},y)$, in order to obtain a correct classification decision rule, we need to have the corresponding $f_y(\mathbf{x})$ to be the maximum among $\mathbf{f}(\mathbf{x})$. To that end, the first term in (6) encourages $f_y(\mathbf{x})$ to be large, and the second term encourages $f_j(\mathbf{x}), j \neq y$, to be small. Consequently, both terms in (6) try to make $f_y$ big.

To further comprehend the MLUM family, we rewrite (6) using the multiple comparison vector representation proposed by Liu and Shen (2006). In particular, we define the comparison vector $g\big(\mathbf{f}(\mathbf{x}),y\big) = \big(f_y(\mathbf{x}) - f_1(\mathbf{x}),\ldots,f_y(\mathbf{x}) - f_{y-1}(\mathbf{x}),f_y(\mathbf{x}) - f_{y+1}(\mathbf{x}),\ldots,f_y(\mathbf{x}) - f_k(\mathbf{x})\big)$. Then by the argmax classification rule, a data point $(\mathbf{x},y)$ is misclassified if and only if $\min g\big(\mathbf{f}(\mathbf{x}),y\big) \leq 0$. Let $u = g\big(\mathbf{f}(\mathbf{x}),y\big)$, then the $0-1$ loss can be written as $I\{\min_j u_j \leq 0\}$, and the MLUM loss can be expressed as

$$V(\mathbf{f},y) = \gamma\ell(\sum_{j=1}^{k-1} u_j/k) + (1-\gamma)\sum_{j=1}^{k-1}\ell(-\sum_{i=1}^{k-1} u_i/k + u_j). \tag{7}$$

Figure 2 shows the plot of (7) with $\gamma = 0, 0.5, 1$ and $c = 0, a = 1$ (soft classifier), and the $0-1$ loss with $k = 3$, for comparison. We can see that as $\gamma$ changes, the shape of the MLUM loss functions varies a lot, although they are all convex upper envelopes of the $0-1$ loss. Moreover, as $\gamma$ increases, the value of the loss function increases when $u_1$ and $u_2$ are both negative, and decreases when just one of them is negative.

In the next section, we explore some statistical properties of the MLUM family, which can help us understand the MLUM better with respect to the parameters involved in (6).

## 3. Statistical Properties

In this section, we study statistical properties of the MLUM family. We first study its consistency in Section 3.1, and then derive the formula for class probability estimation in Section 3.2. In Section 3.3, we extend the notion of the excess $V$-risk, as defined in Bartlett et al. (2006) and Zhang (2004a), from the binary case to the multicategory one. We show that the convergence rate of the excess $V$-risk depends on the size of the functional space, as well as the convergence rate of the estimated classification function to its theoretical minimizer within the functional space.

### 3.1 Fisher Consistency

As different loss functions yield different methods, it is essential to study the properties of these loss functions. One important concept is the Fisher consistency (Zhang, 2004b; Bartlett et al., 2006), defined as follows. For a binary classification problem with the corresponding classification function $f$, a loss function $V(\cdot)$ is Fisher consistent if and only if $\operatorname{sign}\big(f^*(\mathbf{x})\big) = \operatorname{sign}\big(p(\mathbf{x}) - \frac{1}{2}\big)$, where $f^*(\mathbf{x}) = \operatorname{arginf}_f E[V\big(Yf(\mathbf{X})\big)|\mathbf{X} = \mathbf{x}]$ and $p(\mathbf{x}) = p(Y = 1|\mathbf{X} = \mathbf{x})$. Based on the definition, Fisher

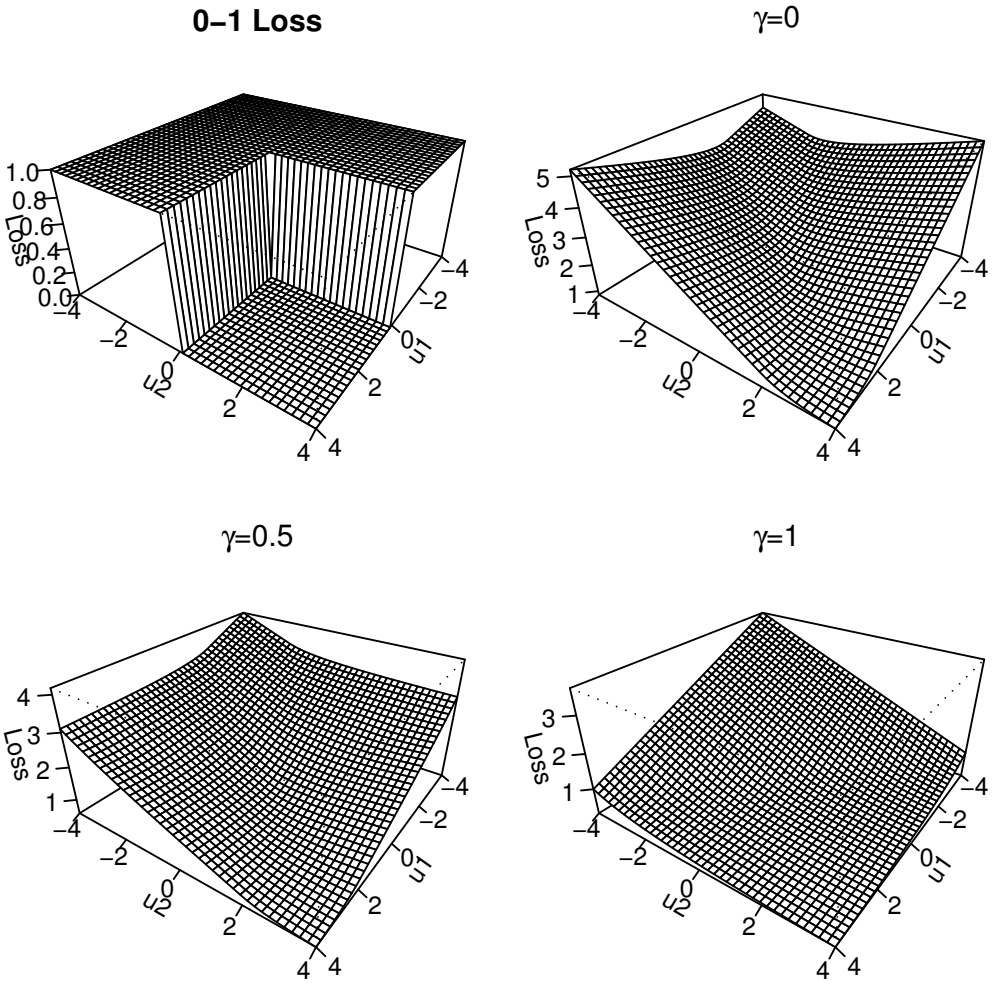



Figure 2: Plots of the $0-1$ loss function (top left panel) and MLUM loss functions with $\gamma = 0$ (top right), $\gamma = 0.5$ (bottom left), $\gamma = 1$ (bottom right), for $c = 0$, $a = 1$ and $k = 3$.

consistency essentially ensures the corresponding decision boundary induced by $f^*$ is identical to the Bayes boundary $\{\mathbf{x} : p(\mathbf{x}) = 1/2\}$.

In the multicategory classification literature, to tackle the Fisher consistency problem, we need the following definitions.

**Definition 1 (Expected $V-$loss):** Define the expected $V-$loss as

$$E_{\mathbf{X},Y} V(\mathbf{f}, y) = E_{\mathbf{X}}\big[\sum_{j=1}^{k} V\big(\mathbf{f}(\mathbf{X}), j\big) P_j(\mathbf{X}) | \mathbf{X} = \mathbf{x}\big],$$

where $\mathbf{P}(\mathbf{x}) = \big(P_1(\mathbf{x}), \ldots, P_k(\mathbf{x})\big)$ is the class conditional probability.

**Definition 2 (Conditional $V-$loss):** For any $\mathbf{x}$, define the conditional $V-$loss as

$$S(\mathbf{f},\mathbf{x}) = \sum_{j=1}^{k} V\big(\mathbf{f}(\mathbf{x}), j\big) P_j(\mathbf{x}). \tag{8}$$

Because of the argmax rule, Fisher consistency means that for any given $\mathbf{P}(\mathbf{x})$, the minimizer $\mathbf{f}^*(\mathbf{x}) = \big(f_1^*(\mathbf{x}),\ldots,f_k^*(\mathbf{x})\big)$ of $S(\mathbf{f},\mathbf{x})$ is such that $\operatorname{argmax}_j P_j(\mathbf{x}) = \operatorname{argmax}_j f_j^*(\mathbf{x})$. Furthermore, if $\operatorname{argmax}_j P_j(\mathbf{x})$ is unique, then so is $\operatorname{argmax}_j f_j^*(\mathbf{x})$. Next we show that the MLUM loss function is always Fisher consistent with a finite $c$ and for any $\gamma \in [0,1]$, $a>0$. To that end, we need to show that, if $P_1$ is the unique maximum among $\{P_1,\ldots,P_k\}$, then $f_1^* > \max\{f_2^*,\ldots,f_k^*\}$. The next lemma assures that in the MLUM family, $f_1^*$ is the maximum among $\{f_1^*,\ldots,f_k^*\}$ (not necessarily unique), even with $c=\infty$.

**Lemma 1** *In the MLUM family with $c\in[0,\infty]$, $a>0$ and $\gamma\in[0,1]$, suppose for $i,j\in\{1,\ldots,k\}$, we have $P_i > P_j$, then $f_i^* \geq f_j^*$.*

Clearly the above lemma is not sufficient for Fisher consistency, because the uniqueness of $f_1^*$ is not guaranteed. Liu and Yuan (2011) showed that if we replace $\ell(\cdot)$ in (6) with the hinge loss with some minor modifications as in (5), the Fisher consistency will fail when $\gamma > 1/2$. The deficiency of the hinge loss is due to its non-differentiability at the point 1, which then assures $f_j^* \leq 1;\ j=1,\ldots,k$. Because the LUM loss $\ell$ is always differentiable with finite $c$, Fisher consistency of the MLUM is guaranteed, as in the following theorem.

**Theorem 2** *The MLUM loss function (6) with any $a>0$, $\gamma\in[0,1]$ and $c\in[0,\infty)$, is Fisher consistent.*

As a remark, we note that the proof of the preceding theorem can shed some light on why the RMSVM is not Fisher consistent when $\gamma > 1/2$. Since the hinge loss is not differentiable at 1, the maximal possible value of $f_j^*;\ j=1,\ldots,k$ in the RMSVM is 1. When $P_1 > P_2 \geq P_3 \cdots \geq P_k$, we may have $f_1^* = f_2^* = 1$. Thus the loss can be inconsistent. See Remark 3 in the appendix for more discussions.

### 3.2 Probability Estimation

Class conditional probability estimation is very important in many applications. It is common to use the relationship between $\mathbf{f}^*$ and the class probability $\mathbf{P}$ for estimation of the latter using $\hat{\mathbf{f}}^n$, where $\hat{\mathbf{f}}^n$ is the empirical solution to (4) with $V$ being the proposed MLUM loss. In this section we convert the minimizer $\mathbf{f}^*$ into the class conditional probability estimation. When an estimated $\hat{\mathbf{f}}^n$ is obtained, one can use the formula to obtain the estimated probability $\hat{\mathbf{P}}$. The following theorem gives the probability estimation formula for the MLUM family with any finite $c$.

**Theorem 3** *Let $\hat{E}(j) = [\gamma\ell'(\hat{f}_j) + (1-\gamma)\ell'(-\hat{f}_j)]$ and $\hat{F}(j) = (1-\gamma)\ell'(-\hat{f}_j);\ j=1,\ldots,k$, where $\hat{f}_j$ is the $j^{th}$ element of $\hat{\mathbf{f}}^n$. Then the probability estimation of $\hat{P}_j$ for the MLUM can be expressed as*

$$\hat{P}_j = \frac{1}{\hat{E}(j)} \left\{ \hat{F}(j) + \frac{1}{\sum_{i=1}^{k} \frac{1}{\hat{E}(i)}} \Big(1 - \sum_{i=1}^{k} \frac{\hat{F}(i)}{\hat{E}(i)}\Big) \right\};\ j=1,\ldots,k. \tag{9}$$

Note that the class probability estimation requires that $\hat{P}_j \in [0,1]$ for $j = 1,\ldots,k$ and $\sum_{j=1}^{k} \hat{P}_j = 1$. One can check that the requirement of the estimated probabilities summing to one is satisfied in (9). For $\gamma = 1$, $F(j) = 0$, and one can directly verify the estimated $\hat{P}_i$ using (9) is proper. However, with $\gamma \neq 1$, $\hat{P}_j$ in (9) may be outside of $[0,1]$. To ensure a proper estimation in the sense of each $0 \leq \hat{P}_j \leq 1$ and $\sum_{j=1}^{k} \hat{P}_j = 1$, we apply a scaled probability estimates using the following formula

$$\hat{P}_j^{scaled} = \frac{\hat{P}_j - \min_{i=1,\ldots,k} \hat{P}_i}{\sum_{m=1}^{k} (\hat{P}_m - \min_{i=1,\ldots,k} \hat{P}_i)}.$$

A similar strategy was previously used in Park et al. (2010). Note that besides this one, other scaling strategies can be used as well.

In the binary case, the LUM provides class conditional probability estimation with any finite $c$. In particular, with the classification function $f$, $p(\mathbf{x})$ does not have a one-to-one relationship with $f^*(\mathbf{x})$ when $p(\mathbf{x}) = 1/2$ and $c > 0$. In that case, all values of $f^*(\mathbf{x}) \in [-\frac{c}{1+c}, \frac{c}{1+c}]$ correspond to $p(\mathbf{x}) = 1/2$. Figure 3 displays the relationship between $p(\mathbf{x})$ and $f^*(\mathbf{x})$ with $a = 1$ and $c \in \{0, 1, \infty\}$. As shown in Figure 3, when $c > 0$, the flat region of $p(\mathbf{x})$ makes the estimation of class conditional probability more difficult (Liu et al., 2011). However, the LUM is still able to provide probability estimation for any finite $c$, although as $c$ increases, the probability information becomes less complete. When $c \to \infty$, the LUM reduces to the standard SVM, which cannot provide any detailed information about probability, due to its minimizer $f^*(\mathbf{x}) = \text{sign}\big(p(\mathbf{x}) - 1/2\big)$.

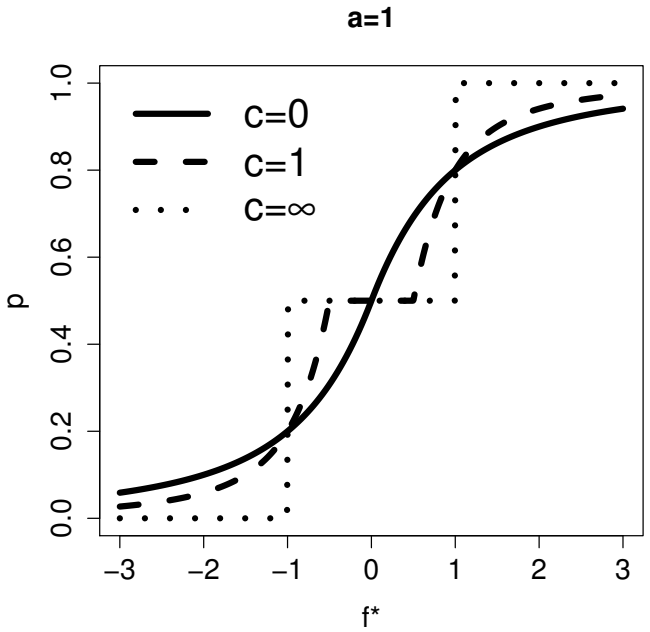



Figure 3: Plot of the correspondence between $f^*(\mathbf{x})$ and $p(\mathbf{x})$ with $a = 1$, $c \in \{0, 1, \infty\}$.

A similar pattern exists in the MLUM case. In multicategory problems, the conditional probability becomes a vector, and this makes the transition behavior of probability estimation from soft to hard classification more complex. In particular, we need to generalize the flat region in Figure 3

to the multicategory setting. From (9), we can see that for any given $\hat{\mathbf{f}}^n = (\hat{f}_1, \ldots, \hat{f}_k)$, the estimated probability depends entirely on $\hat{E}(j)$ and $\hat{F}(j)$; $j = 1, \ldots, k$. Note that the LUM loss has the derivative

$$\ell'(u) = \begin{cases} -1 & \text{if } u < \frac{c}{1+c}, \\ -\left(\frac{a}{(1+c)u - c + a}\right)^{(a+1)} & \text{if } u \geq \frac{c}{1+c}. \end{cases}$$

When $\hat{f}_i \in [-\frac{c}{1+c}, \frac{c}{1+c}], \hat{f}_j \in [-\frac{c}{1+c}, \frac{c}{1+c}]$, one can see that $\hat{E}(i) = \hat{E}(j), \hat{F}(i) = \hat{F}(j)$, and consequently we have $\hat{P}_i = \hat{P}_j$. This implies that for any obtained $\hat{\mathbf{f}}^n$, if the classification signal is weak such that both $\hat{f}_i$ and $\hat{f}_j$ fall into $[-\frac{c}{1+c}, \frac{c}{1+c}]$, then we are not able to tell the difference between the two classes in terms of conditional probabilities. Moreover, if $\hat{f}_j \in [-\frac{c}{1+c}, \frac{c}{1+c}]$ for all $j \in \{1, \ldots, k\}$, then $\hat{P}_j = 1/k$ for all $j$ using (9). In that case, the estimated probability cannot help us to identify the max probability class.

To further illustrate the relationship between $\hat{\mathbf{f}}^n$ and $\hat{\mathbf{P}}$, we use Figure 4 to display the relationship between $(\hat{f}_1, \hat{f}_2)$ and $\hat{P}_1$ for $k = 3$, $\gamma = 1$, $a = 1$ and $c = 0, 2, 5$ and $\infty$. When $c > 0$, the flat region of $\hat{\mathbf{P}}$ makes the estimation of the conditional probability more difficult. As $c$ increases, $[-\frac{c}{1+c}, \frac{c}{1+c}]$ becomes wider, and the function becomes closer to a step function. Eventually, when $c \to \infty$, the method reduces to the RMSVM whose classification function $\mathbf{f}$ can only produce classification boundary without containing any further probability information. Therefore, similar to the binary case, the MLUM can provide class probability estimation for any finite $c$, although the estimation deteriorates as $c$ increases.

### 3.3 Asymptotic Properties

For binary problems, Zhang (2004a) and Bartlett et al. (2006) investigated the effect of employing the surrogate loss in place of the $0-1$ loss, in terms of classification performance. They considered the excess risk and the excess $V-$risk, defined as follows. The excess risk is the difference between the expected $0-1$ loss for $f$ and its theoretical infimum. It can be written as $R\big(f(\cdot)\big) - R^*$, where $R\big(f(\cdot)\big)$ and $R^*$ are defined in Section 2.1. Similarly, the excess $V-$risk can be written as $Q\big(f(\cdot)\big) - Q^*$, where $Q\big(f(\cdot)\big) = E_{\mathbf{X},Y} V\big(f(\mathbf{X}), Y\big)$ is the expected loss using $V$ as the loss function, and $Q^* = \min Q\big(f(\cdot)\big)$.

Typically we are interested in the convergence of the excess risk. Steinwart and Scovel (2007) studied the convergence rate of the Gaussian kernel binary SVM problem. For problems with general loss functions in the binary case, Zhang (2004a) and Bartlett et al. (2006) showed that if the excess $V$-risk converges to 0, so does the excess risk, under some mild assumptions. Zhang (2004b) further studied the relationship between the two excess risks in multicategory problems. In particular, the convergence rates of the two excess risks are well studied in Wang and Shen (2007), in the setting of $L_1$ penalized multicategory SVM with linear learning. In this section, we employ the generalization of the excess $V$-risk from binary to multicategory cases as in Zhang (2004b), and explore the explicit form for MLUM. Moreover, we study the relationship between the convergence rate of the classification function $\hat{\mathbf{f}}^n$ and that of the excess $V$-risk, as well as the size of the functional space.

Recall that we can rewrite $Q\big(\mathbf{f}(\cdot)\big)$ as $E_{\mathbf{X}} S(\mathbf{f}, \mathbf{x})$. Consider the MLUM case with $S(\mathbf{f}, \mathbf{x}) = \sum_{j=1}^k \big[\gamma \ell\big(f_j(\mathbf{x})\big) + (1-\gamma) \sum_{i \neq j} \ell\big(-f_i(\mathbf{x})\big)\big] P_j(\mathbf{x})$. It can be verified that $S(\mathbf{f}, \mathbf{x}) = \sum_{j=1}^k P_j(\mathbf{x}) \, [\gamma \ell\big(f_j(\mathbf{x})\big) - (1-\gamma)\ell\big(-f_j(\mathbf{x})\big)] + (1-\gamma) \sum_{j=1}^k \ell\big(-f_j(\mathbf{x})\big)$. For brevity, let $Q(\mathbf{P}, \mathbf{f}) = \sum_{j=1}^k P_j \, [\gamma \ell(f_j) - (1 -$

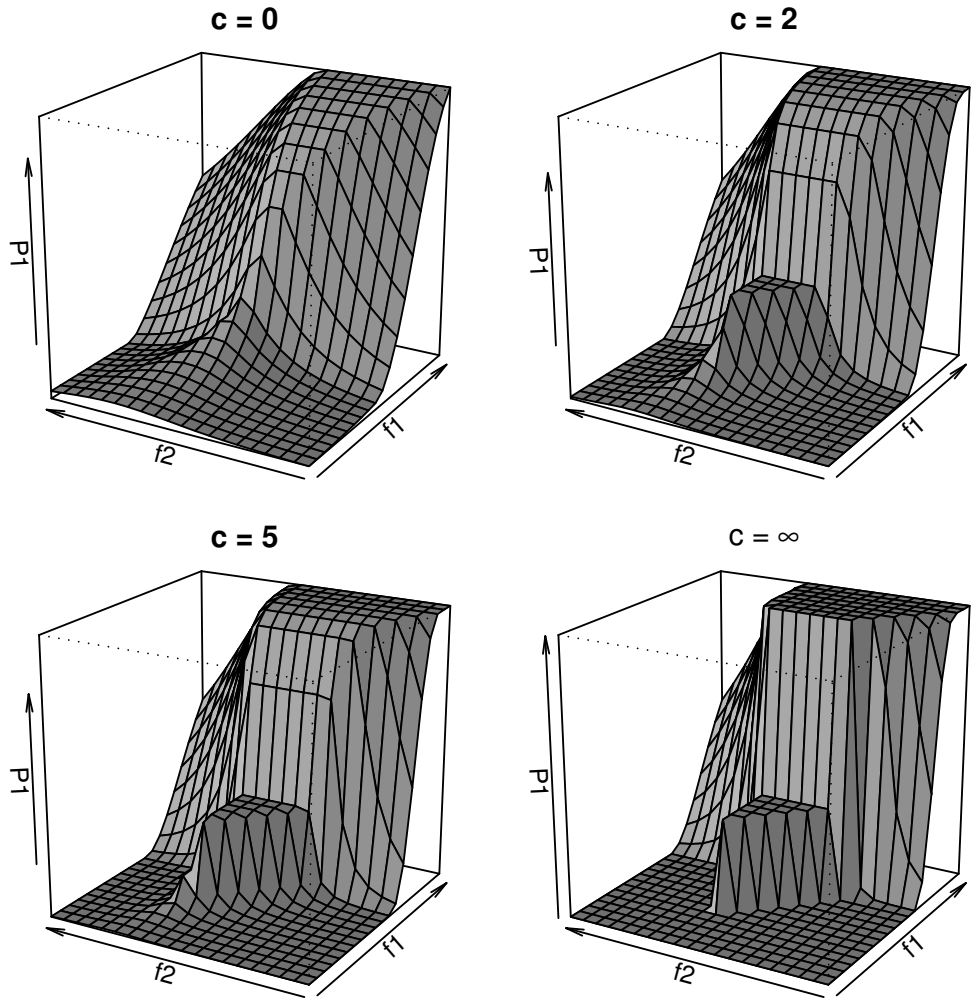



Figure 4: Visualization of the relationship between $f_1^*$, $f_2^*$ and $P_1$, with $\gamma = 1$, $a = 1$, and $c \in \{0, 2, 5, \infty\}$. The plots show the transition of the MLUM from soft classification (top left panel) to hard classification (bottom right panel).

$\gamma)\ell(-f_j)] + (1-\gamma)\sum_{j=1}^{k}\ell(-f_j)$. Note that $Q(\mathbf{P},\mathbf{f})$ does not involve $\mathbf{x}$. For any given $\mathbf{P}$, let $\mathbf{f}_\ell^*(\mathbf{P}) = \text{arginf}_{\mathbf{f}\in\mathbf{R}^k} Q(\mathbf{P},\mathbf{f})$, where $\mathbf{R}^k$ is the $k-$dimensional real space. Define $Q^*(\mathbf{P}) = Q\big(\mathbf{P},\mathbf{f}_\ell^*(\mathbf{P})\big)$ to be the optimum value for any given $\mathbf{P}$, and $\Delta Q(\mathbf{P},\mathbf{f}) = Q(\mathbf{P},\mathbf{f}) - Q^*(\mathbf{P})$. The excess $V-$risk is equivalent to $\Delta Q\big(\mathbf{f}(\cdot)\big) = E_{\mathbf{X}}\Delta Q\big(\mathbf{P}(\mathbf{X}),\mathbf{f}(\mathbf{X})\big)$. Zhang (2004a) and Bartlett et al. (2006) showed that in binary problems, one can bound the excess risk by some transformation of $\Delta Q\big(\mathbf{f}(\cdot)\big)$. A similar result for multicategory problems is obtained in Zhang (2004b). To study the functional form of $\Delta Q\big(\mathbf{f}(\cdot)\big)$ in binary problems, Zhang (2004a) proposed to use the Bregman divergence. Here we extend the results in Zhang (2004a) to the multicategory version.

The Bregman divergence of a convex function $g(\cdot)$ is defined as

$$d_g(f_1, f_2) = g(f_2) - g(f_1) - g'(f_1)(f_2 - f_1).$$

Here $g'(\cdot)$ is a subgradient of the convex function $g(\cdot)$. In Theorem 2.2 of Zhang (2004a), it was shown that in the binary case, if $g$ is differentiable, $\Delta Q(P, f) = P d_g\big(f_g^*(P), f\big) + (1-P) d_g\big(-f_g^*(P), -f\big)$. For the MLUM family, we have the following result, which is a generalization of the binary formula.

**Theorem 4** $\Delta Q(\mathbf{P}, \mathbf{f}) = \sum_{j=1}^k \big[P_j \gamma d_\ell(f_j^*, f_j) + (1-P_j)(1-\gamma) d_\ell(-f_j^*, -f_j)\big]$.

Theorem 4 can be used to establish the connection of the convergence rate of the classification function and that of the excess $V-$risk. Suppose the classification function $\mathbf{f}$ varies in a certain space $H$. We can decompose the excess $V-$risk as $\Delta Q(\hat{\mathbf{f}}^n) = [\Delta Q(\hat{\mathbf{f}}^n) - \Delta Q(\mathbf{f}^H)] + [\Delta Q(\mathbf{f}^H)]$, where $\mathbf{f}^H$ is the function that achieves the minimal of $\Delta Q(\mathbf{f}) : \mathbf{f} \in H$ pointwisely. We may refer to the first term in the previous display as the $V-$estimation error and the second term as the $V-$approximation error. When $H$ is rich enough, so that

$$\operatorname*{arginf}_{\mathbf{f} \in H} E_{\mathbf{X},Y}\Big(V\big(\mathbf{f}(\mathbf{X}), Y\big)\Big) = \operatorname*{arginf}_{\mathbf{f} \in \mathbf{R}^k} E_{\mathbf{X},Y}\Big(V\big(\mathbf{f}(\mathbf{X}), Y\big)\Big) = \mathbf{f}^*, \tag{10}$$

the estimator $\hat{\mathbf{f}}^n$ will converge to $\mathbf{f}^*$ as the sample size $n$ grows, under some mild conditions. In this case the $V-$approximation error is zero. We can explore how the convergence rate of $\hat{\mathbf{f}}^n$ affects the convergence rate of the excess $V-$risk, which is essentially the $V-$estimation error. First we introduce some definitions and assumptions. Recall that $a$, $c$ and $\gamma$ are parameters in the MLUM loss function (6).

Let $\mu(\cdot)$ be the regular Lebesgue measure. For any fixed $a$, $c$ and $\gamma$, the distribution $D(\mathbf{X}, Y)$ naturally defines $k$ probability measures on the real line: $\tau_j(B) = P(f_j^* \in B);\ j = 1, \ldots, k$, where $B$ is any Borel measurable set.

**Assumption A:** For any $c \geq 0$, $a > 0$ and $\gamma \in [0,1]$, $\tau_j \ll \mu;\ j = 1, \ldots, k$. Namely, every measure $\tau_j$ is absolutely continuous with respect to the Lebesgue measure $\mu$.

**Assumption B:** For any $c \geq 0$, $a > 0$ and $\gamma \in [0,1]$, $n^q\big(\hat{\mathbf{f}}^n(\mathbf{x}, y) - \mathbf{f}^*(\mathbf{x}, y)\big) \to \mathbf{T}(c, a, \gamma, \mathbf{x}, y)$ in distribution, where $\mathbf{T}(c, a, \gamma, \mathbf{x}, y) = (T_1, \ldots, T_k)^T$ is a multivariate random variable, whose distribution depends on $c$, $a$, $\gamma$, and varies among different $(\mathbf{x}, y)$; $q > 0$ is a constant. Furthermore, suppose that for fixed $c$, $a$, and $\gamma$, $\int_{\mathbf{X},Y} |\sup_{1 \leq j \leq k} T_j|^2 dD(\mathbf{X}, Y) < \infty$.

Now we are in the position to introduce the following theorem.

**Theorem 5** *In the MLUM family with the underlying distribution D, suppose Assumptions A and B are satisfied, and (10) holds. Then for any fixed c, a, and γ,*

$$\Delta Q(\hat{\mathbf{f}}^n) = O(n^{-2q}).$$

Note that Assumption A can be verified if there is no probability mass point in the distribution $D$. Under Assumption A, the expected loss $Q\big(\mathbf{f}(\cdot)\big)$ has bounded second order derivative for a fixed $c$ almost surely. Hence under some conditions, $q = 1/2$ for regular finite dimensional problems,

and $\hat{\mathbf{f}}^n$ is root $n-$consistent. For example, in the literature of sieve estimation with linear learning, a finite dimensional problem enjoys the root $n-$consistency of $\hat{\beta}$ (Shen and Wong, 1994), with a proper choice of the regularization term. In this case, the integrability in Assumption B can be satisfied if $\mathbf{X}$ is bounded. From Theorem 5, we can conclude that the excess $V-$risk is $n-$consistent for any $a$, $c$ and $\gamma$, when the function space is large enough.

**Remark 1.** Assumption A ensures that there is no probability mass point where the LUM loss function $\ell(\cdot)$ is not twice differentiable. Without Assumption A we are not even guaranteed to have convergence for any suitable transformation of $\hat{\mathbf{f}}^n - \mathbf{f}^*$. The square integrable requirement of Assumption B is essential in the sense that it prevents the distribution of $\mathbf{T}$ from diverging with large probability when $(\mathbf{X}, Y)$ varies.

**Remark 2.** The potential problem of non-unique $\mathbf{f}^*$ does not affect the result of Theorem 5. Because $S$ in (8) is convex, any partial derivative of $S$ with respect to $f_j$ is non-decreasing. Suppose there is a flat region $[h_1, h_2]$ of value 0 in the derivative function. Then $f_j^*$ must be within $[h_1, h_2]$ and the second order partial derivative with respect to $f_j$ is 0 in $(h_1, h_2)$. Because the MLUM loss function is continuously differentiable, either the first order derivative is not differentiable on the boundary of $(h_1, h_2)$, which we assume probability 0, or it is differentiable with derivative 0. Note that if the first order derivative of a convex function $\psi(\cdot)$ is 0 within $(h_1, h_2)$, then for any $t_1, t_2 \in (h_1, h_2)$, and any other $t_3$, $d_\psi(t_1, t_3) = d_\psi(t_2, t_3)$. This means that the choice of $f_j^*$ is not essential, as long as the empirical minimizer $\hat{f}_j$ approaches $[h_1, h_2]$.

The situation when the $V-$approximation error vanishes is rare. When the $V-$approximation error is nonzero, in other words,

$$\inf_{\mathbf{f} \in H} E_{\mathbf{X},Y}\Big(V\big(\mathbf{f}(\mathbf{X}), Y\big)\Big) > \inf_{\mathbf{f} \in \mathbf{R}^k} E_{\mathbf{X},Y}\Big(V\big(\mathbf{f}(\mathbf{X}), Y\big)\Big), \tag{11}$$

Theorem 5 is not applicable, because the excess $V-$risk does not converge to 0 anymore. Then we are interested in the convergence rate of the $V-$estimation error, $\Delta Q(\hat{\mathbf{f}}^n) - \Delta Q(\mathbf{f}^H)$. First, we need to modify Assumption B as follows.

**Assumption B':** For any $c \geq 0$, $a > 0$ and $\gamma \in [0, 1]$, $n^q\big(\hat{\mathbf{f}}^n(\mathbf{x}, y) - \mathbf{f}^*(\mathbf{x}, y)\big) \to \mathbf{T}(c, a, \gamma, \mathbf{x}, y)$ in distribution, where $\mathbf{T}(c, a, \gamma, \mathbf{x}, y) = (T_1, \ldots, T_k)^T$ is a multivariate random variable, whose distribution depends on $c$, $a$, $\gamma$, and varies among different $(\mathbf{x}, y)$; $q > 0$ is a constant. Furthermore, suppose that for fixed $c$, $a$, and $\gamma$, $\int_{\mathbf{X},Y} |\sup_{1 \leq i \leq k} T_i| dD(\mathbf{X}, Y) < \infty$, and $\int_{\mathbf{X},Y} |\sup_{1 \leq i \leq k} f_i^*| + |\sup_{1 \leq i \leq k} f_i^H| dD(\mathbf{X}, Y) < \infty$.

**Theorem 6** *In the MLUM family with the underlying distribution D, suppose Assumptions A and B' are satisfied, and (11) holds. Then for any fixed c, a, and* $\gamma$*,*

$$\Delta Q(\hat{\mathbf{f}}^n) - \Delta Q(\mathbf{f}^H) = O(n^{-q}).$$

From Theorem 6, we can see that when the theoretical minimizer does not belong to the function class $H$, the convergence rate of the excess $V$-risk is the same as $\hat{\mathbf{f}}^n$. When $q = 1/2$ under some mild conditions, the excess $V$-risk is also root $n$-consistent.

## 4. Computational Algorithm

In this section, we discuss how to implement (4) with the MLUM family. The MLUM loss (6) is convex and first-order differentiable, and one can apply standard tools to solve the optimization problem, for example, the OPTIM function in R. Here we propose to use the well-known cyclic coordinate descent algorithm (Tseng, 2001; Friedman et al., 2010).

Next we present the coordinate descent algorithm to solve the following MLUM optimization problem

$$\min_{\mathbf{f}} \frac{1}{n}\sum_{i=1}^{n}\big[\gamma\ell\big(f_y(\mathbf{x}_i)\big)+(1-\gamma)\sum_{j\neq y}\ell\big(-f_j(\mathbf{x}_i)\big)\big]+\lambda J(\mathbf{f})$$
$$\text{subject to } \sum_{j=1}^{k} f_j(\mathbf{x})=0.$$

We choose the first $(k-1)^{th}$ elements of $\mathbf{f}$ as free parameters, and let $\sum_{j=1}^{k} f_j(\mathbf{x})=0$ for all $\mathbf{x}$. For simplicity we focus on the linear learning with $f_j(\mathbf{x})=\mathbf{x}^T\beta_j+b_j, j=1,\ldots,k$, and the $L_2$ penalty with $J(\mathbf{f})=\frac{1}{2}\sum_{j=1}^{k}\|\beta_j\|^2$. Extensions to kernel learning and other convex and separable types of regularization term are relatively straightforward and are not included here.

We now describe the coordinate descent algorithm in detail. Denote the solution at the $m^{th}$ step by $\Xi^{(m)}=(\beta_1^{(m)},\ldots,\beta_k^{(m)})^T$ and $\mathbf{b}=(b_1^{(m)},\ldots,b_k^{(m)})^T$. At the $(m+1)^{th}$ step, define $\tilde{\mathbf{f}}_{i,j,-p}=(\mathbf{x}_i^T\beta_1^{(m+1)}+b_1^{(m+1)},\ldots,\mathbf{x}_i^T\beta_{j-1}^{(m+1)}+b_{j-1}^{(m+1)},W_{i,1},\ldots,\mathbf{x}_i^T\beta_{k-1}^{(m)}+b_{k-1}^{(m)},W_{i,2})^T$ for $i=1,\ldots,n$, $j=1,\ldots,k-1$ and $p=1,\ldots,d$, where $d$ is the dimension of the problem. Here $W_{i,1}=\sum_{q<p}x_{iq}\Xi_{j,q}^{(m+1)}+\sum_{q>p}x_{iq}\beta_{j,q}^{(m)}+b_j^{(m)}$, and $W_{i,2}$ is determined by other components such that the component sum of $\tilde{\mathbf{f}}_{i,j,-p}$ is 0. Set

$$\Xi_{j,p}^{(m+1)}=\operatorname*{argmin}_{z}\frac{\lambda}{2}\Big(z^2+[\sum_{s=1}^{p-1}\Xi_{j,s}^{(m+1)}+z+\sum_{s=p+1}^{d}\Xi_{j,s}^{(m)}]^2\Big)+\frac{1}{n}\sum_{i=1}^{n}V(\tilde{\mathbf{f}}_{i,j,-p}+x_{i,j}z,y_i). \tag{12}$$

The optimization of (12) involves a one-dimensional update and can be solved efficiently. Once we finish updating the entire $j^{th}$ row of $\Xi^{(m+1)}$, we update the intercept $b_j$ using a similar idea as (12) without the regularization term. We continue the iteration until convergence.

The above coordinate descent method can be summarized as the following pseudo-code.

Step 1 Start with $\beta_j=(0,\ldots,0)^T$ and $b_j=0$, for all $j=1,\ldots,k$. We keep $\beta_k=-\beta_1-\beta_2-\cdots-\beta_{k-1}$ and $b_k=-b_1-b_2-\cdots-b_{k-1}$.

Step 2 At the beginning of the $m^{th}$ loop, suppose we have $(\beta_1,\ldots,\beta_k)$ and $(b_1,\ldots,b_k)$ as the intermediate results.

- 2.1 Update, in orders, the elements of $\beta_1$.
- 2.2 After the entire vector $\beta_1$ has been updated, update the intercept $b_1$.
- 2.3 Update the vectors $\beta_2,\ldots,\beta_{k-1}$ and the intercepts $b_2,\ldots,b_{k-1}$, in the same manner as above.

Step 3 Repeat Step 2 until convergence.

## 5. Simulated Examples

In this section, we use six simulated examples to demonstrate the numerical performance of the MLUM family. Our unified algorithm, discussed in Section 4, greatly facilitates a systematic exploration and comparison of different types of classifiers. The MLUM also provides class conditional probability estimation as a by-product. The simulated examples we consider here are different scenarios in which the behavior of different classifiers varies from one to another. Since we know the underlying data generation schemes for simulated examples, the results can shed some light on the behavior of MLUM and offer some insights on the choice of methods in different situations.

We divide this section into four subsections. Section 5.1 contains two examples in which the hard classifier works the best. Section 5.2 consists of two examples that favor the soft classifier. Section 5.3 includes two examples where the MLUM with $c = 1$, corresponding to a new multicategory DWD, can outperform both hard and soft classifiers, which was not observed in the binary LUM case in Liu et al. (2011). Note that this multicategory DWD is different from the version proposed in Huang et al. (2013). Section 5.4 provides some summary on the effect of hard versus soft classifiers and gives some insight on the choice of classification methods. For each simulation, we apply the linear learning with the $L_2$ penalty, and the corresponding tuning parameter $\lambda$ is selected by minimizing the classification error over a separate tuning set using a grid search.

In practice, a systematic tuning procedure for selection of the optimal $(a, c, \gamma)$ is needed. The three-dimensional tuning can be very time consuming, and we suggest a fast and effective tuning procedure for the MLUM family. Similar to the binary case, the choice of $a$ in the MLUM loss has little impact on the performance of the classifiers, so we fix $a$ at 1. Interestingly, we find that the behavior of MLUM with $c = 1$ can outperform the hard ($c \to \infty$) and soft ($c = 0$) classifiers in certain cases. Based on our numerical experience, we suggest to choose $\gamma$ from $\{0, 0.25, 0.5, 0.75, 1\}$. In particular, we propose to tune the MLUM with $c \in \{0, 1, 1000\}$ and $\gamma \in \{0, 0.25, 0.5, 0.75, 1\}$, holding $a$ fixed at 1. We call this procedure the "tuned MLUM".

For each example, we evaluate both classification performance and probability estimation. We use the test error to measure classification accuracy, on a test set with size $10^6$. To explore the effect of different $c$ independently, we let $c$ vary in $\{0, 1, 10, 100, 1000\}$, and let the tuning procedure automatically choose the best $\gamma$ from $\{0, 0.25, 0.5, 0.75, 1\}$. We call this procedure "tuned $\gamma$, fixed $c$". Similarly, with $\gamma$ varying in $\{0, 0.1, 0.2, \ldots, 0.9, 1\}$, we select the best $c$ from $\{0, 1, 1000\}$, and call the resulting classifier "tuned $c$, fixed $\gamma$", to observe the effect of $\gamma$. For comparison, we also include the results of Losses 1-4 mentioned in Section 2.2, RMSVM, Multicategory Penalized Logistic Regression (MPLR), along with the tuned MLUM, in Table 7. Here for the MPLR, we replace the loss function $\ell$ by the logistic loss $V(f, y) = \log(1 + e^{-yf})$, while keeping the convex combination so that the classifier is tuned with different $\gamma$ values. We would like to point out that this MPLR is new and we refer it as the "tuned MPLR", analogous to the tuned MLUM. Note that in most examples, the tuned MLUM performs better than the other methods, and is recommended.

We conduct 1000 replications for each simulation, and report the average test errors. For probability estimation, we use the criterion of the mean absolute error (MAE), $E_{\mathbf{X}}|p(\mathbf{X}) - \hat{p}(\mathbf{X})|$. The MLUMs with $c \in \{0, 1, 1000\}$ and $\gamma \in \{0, 0.25, 0.5, 0.75, 1\}$ are fit to explore the pattern of probability estimation accuracy. Through 1000 replicates, MAEs with their standard errors are reported. As the number of covariates for Examples 1, 2, 4 and 5 is 2, we plot the corresponding marginal distributions of $\mathbf{x}$ for typical training samples in Figure 5.

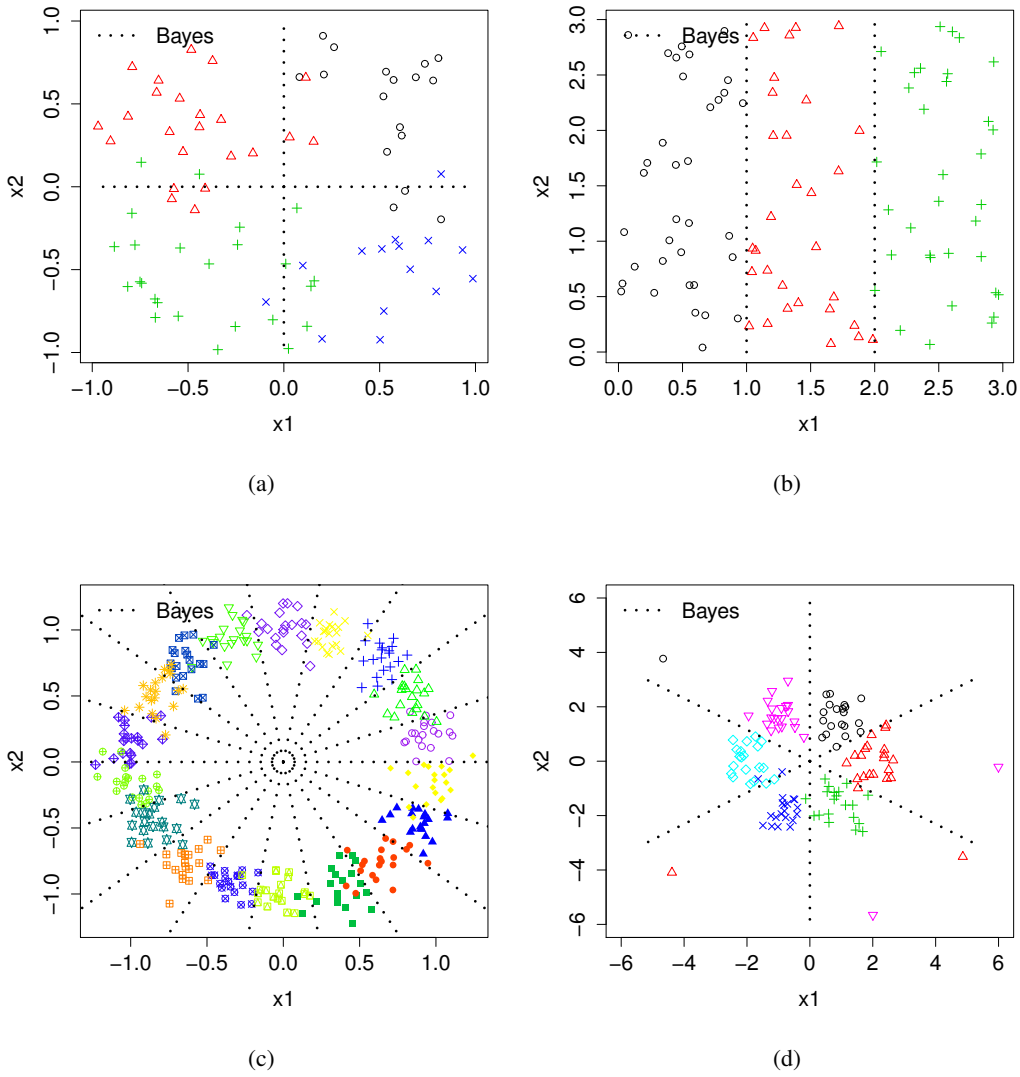



Figure 5: Plots of typical samples for Examples 1(a), 2(b), 4(c) and 5(d) respectively.

### 5.1 Hard Classification Better

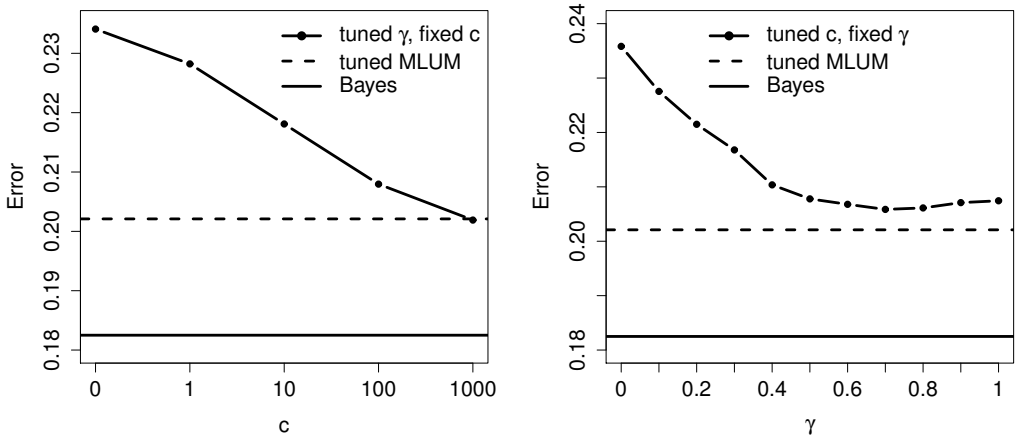
In this section, we generate the data such that the underlying probability functions are step functions. In this case, estimation of the conditional probability function is challenging, and we expect hard classifiers to perform better than the soft ones, because the former bypasses probability estimation.

**Example 1:** We generate two dimensional data uniformly on $[-1,1]^2$, with four classes. Conditional on $X_1$ and $X_2$, the class output is generated as follows. Let $Y=1$ if $X_1>0.2$ and $X_2>0.2$, $Y=2$ if $X_1<-0.2$ and $X_2>0.2$, $Y=3$ if $X_1<-0.2$ and $X_2<-0.2$ and $Y=4$ if $X_1>0.2$ and $X_2<-0.2$. When $X_1>0.2$ and $X_2\in[-0.2,0.2]$, $P(Y=1)=P(Y=4)=1/2$. Similarly, when $X_1<-0.2$ and $X_2\in[-0.2,0.2]$, $P(Y=2)=P(Y=3)=1/2$, and when $X_2>0.2$ and $X_1\in[-0.2,0.2]$, $P(Y=1)=$

$P(Y=2)=1/2$, and when $X_2<-0.2$ and $X_1\in[-0.2,0.2]$, $P(Y=2)=P(Y=4)=1/2$. When the data points fall in $[-0.2,0.2]^2$, the probabilities of being in four classes are equal. In this example we use 80 data points for training and another 80 for tuning.

This is a multicategory generalization of Example 2 in Liu et al. (2011). The underlying conditional class probability function of this example is a step function. Thus class probabilities are difficult to estimate in this case, and the classification accuracy of soft classifiers may be sacrificed by probability estimation.

The test errors of this example are reported in Figure 6. As we expect, hard classifiers outperform soft ones. With tuned $C$, MLUM with $\gamma>0.5$ works better than those with $\gamma\leq 0.5$. We would like to point out that, with finite $c$, the classification performance of MLUM differs from RMSVM, because the MLUM with finite $c$ is always Fisher consistent, while the RMSVM does not guarantee consistency for $\gamma>0.5$. Also, the tuned MLUM is more flexible and achieves the best classification accuracy. The MAEs are reported in Table 1, and the soft classifier gives the best probability estimation.




(a) The test error of Example 1 with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Example 1 with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

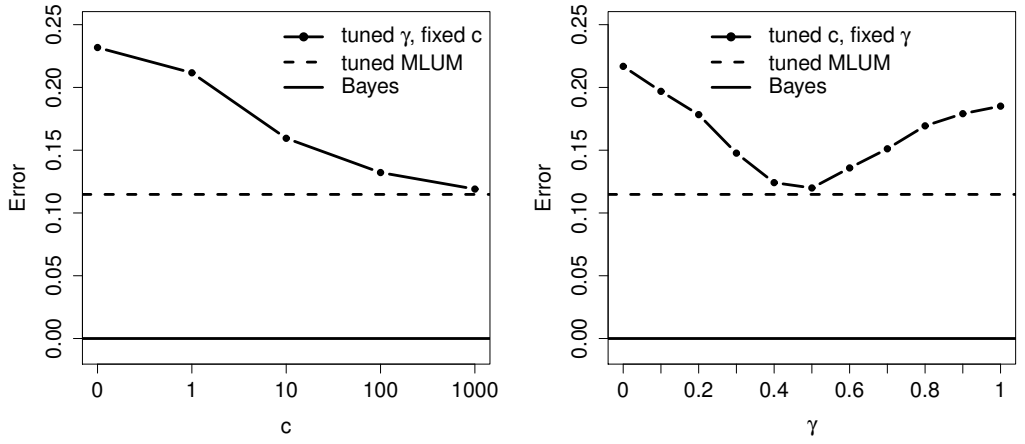
Figure 6: Classification error rates in Example 1. The left panel shows that the hard classifier performs better than the other ones in this example. The right panel shows that the test error is minimized with $\gamma$ around 0.7.

**Example 2:** We generate a three-class problem in this example, and $\mathbf{X}$ is two dimensional. The first class is uniformly distributed in the square $x_1\in[0,1],x_2\in[0,3]$, the second class is uniform from $x_1\in[1,2],x_2\in[0,3]$, and the third class is uniform from $x_1\in[2,3],x_2\in[0,3]$. There are 60 training data points and 60 tuning ones, respectively.

| MAE | $\gamma = 0$ | $\gamma = 0.25$ | $\gamma = 0.5$ | $\gamma = 0.75$ | $\gamma = 1$ |
|---|---|---|---|---|---|
| $c = 0$ | 0.1226 | 0.1208 | 0.1165 | 0.1050 | 0.0491 |
| | (0.005844) | (0.001845) | (0.002189) | (0.003622) | (0.009019) |
| $c = 1$ | 0.1422 | 0.1368 | 0.1328 | 0.1215 | 0.0935 |
| | (0.005240) | (0.003535) | (0.005132) | (0.003987) | (0.004582) |
| $c = 1000$ | 0.1424 | 0.1387 | 0.1372 | 0.1321 | 0.1267 |
| | (0.002661) | (0.005278) | (0.001347) | (0.004159) | (0.008138) |

Table 1: The MAEs for different $c$ and $\gamma$ in Example 1.

We report the classification errors in Figure 7, and the MAEs in Table 2. Based on the results, both test errors and MAEs suggest that the hard classifier performs significantly better than the others. Interestingly, the soft classifier gives worse probability estimation than hard classifiers in this example. Here the parameter $\gamma = 0.5$ works better than other values, which is consistent with the findings in Liu and Yuan (2011).




(a) The test error of Example 2 with $c$ in $\{0, 1, 10, 100, 1000\}$ and $\gamma$ tuned.

(b) The test error of Example 2 with $\gamma$ in $\{0, 0.1, \ldots, 0.9, 1\}$ and $c$ tuned.

Figure 7: Classification error rates in Example 2. The hard classifier works the best, as is suggested by the left panel. The right panel shows that $\gamma = 0.5$ is optimal.

### 5.2 Soft Classification Better

In this section we generate the data so that the conditional probability function is smooth and relatively easy to estimate. In such cases the accurate probability estimation may help the soft classifier to build more accurate classification boundaries.

| MAE | $\gamma = 0$ | $\gamma = 0.25$ | $\gamma = 0.5$ | $\gamma = 0.75$ | $\gamma = 1$ |
|---|---|---|---|---|---|
| $c = 0$ | 0.2113 | 0.1969 | 0.1933 | 0.1887 | 0.1892 |
| | (0.006877) | (0.006228) | (0.004803) | (0.001011) | (0.009567) |
| $c = 1$ | 0.1790 | 0.1734 | 0.1662 | 0.1693 | 0.1674 |
| | (0.001738) | (0.005951) | (0.003742) | (0.006418) | (0.003838) |
| $c = 1000$ | 0.1551 | 0.1523 | 0.1469 | 0.1430 | 0.1422 |
| | (0.002738) | (0.007453) | (0.007681) | (0.001394) | (0.008392) |

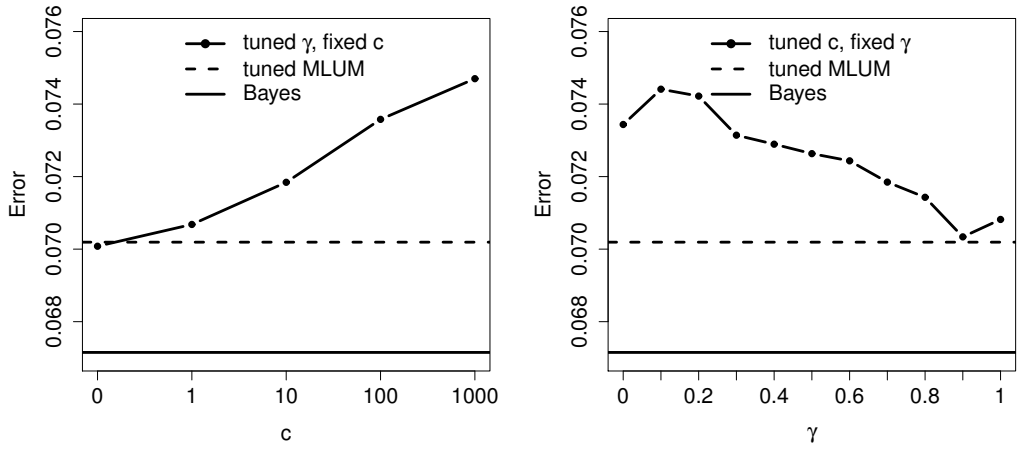
Table 2: The MAEs for different $c$ and $\gamma$ in Example 2.

**Example 3:** This example involves a three-dimensional feature space with eight classes. In particular,

$$P(Y = j) = 1/8, P(\mathbf{X}|Y = j) \sim N(\mu_j, 0.5^2 I_2);\ j = 1, \ldots, 8,$$

where the means $\mu_j = (1,1,1)^T$, $(1,1,-1)^T$, $(1,-1,1)^T$, $(1,-1,-1)^T$, $(-1,1,1)^T$, $(-1,1,-1)^T$, $(-1,-1,1)^T$, $(-1,-1,-1)^T$, for $j = 1, \ldots, 8$, respectively. Because the number of classes is large, we generate 160 observations for training, and another 160 for tuning.

The classification performance is reported in Figure 8, and the MAEs are reported in Table 3. This is a case in which the soft classifier works the best both in terms of classification accuracy as well as probability estimation.




(a) The test error of Example 3 with $c$ in $\{0, 1, 10, 100, 1000\}$ and $\gamma$ tuned.

(b) The test error of Example 3 with $\gamma$ in $\{0, 0.1, \ldots, 0.9, 1\}$ and $c$ tuned.

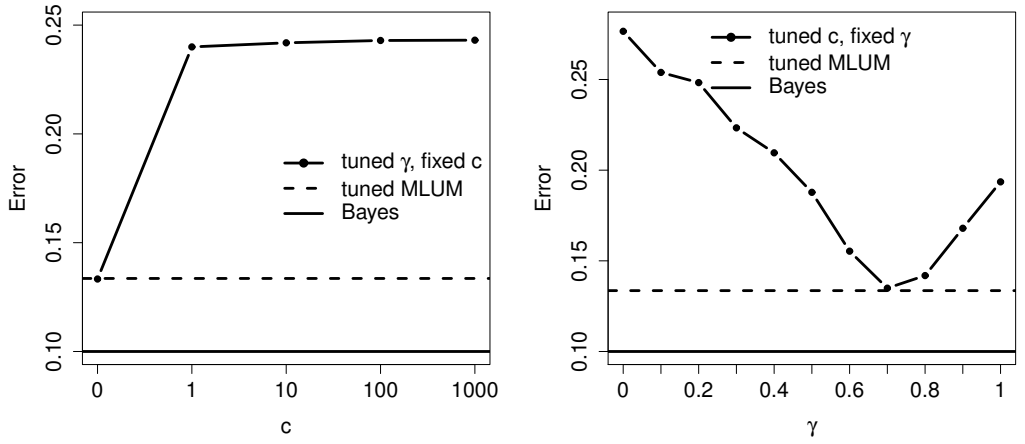
Figure 8: Classification error rates in Example 3. The left panel shows that this is an example in which soft classifier works the best. The right panel illustrates that the test error is minimized with $\gamma = 0.9$.

| MAE | $\gamma = 0$ | $\gamma = 0.25$ | $\gamma = 0.5$ | $\gamma = 0.75$ | $\gamma = 1$ |
|---|---|---|---|---|---|
| $c = 0$ | 0.2362 | 0.2183 | 0.1956 | 0.1617 | 0.0887 |
| | (0.005048) | (0.008325) | (0.007328) | (0.004206) | (0.000884) |
| $c = 1$ | 0.2629 | 0.2438 | 0.2230 | 0.1785 | 0.0867 |
| | (0.003554) | (0.003873) | (0.006405) | (0.001282) | (0.000671) |
| $c = 1000$ | 0.2867 | 0.2540 | 0.2432 | 0.1990 | 0.1073 |
| | (0.003513) | (0.001959) | (0.007493) | (0.002192) | (0.000560) |

Table 3: The MAEs for different $c$ and $\gamma$ in Example 3.

**Example 4:** In this example we have 2 covariates, and the number of classes is 20. Each marginal distribution of $\mathbf{X}|Y = j;\ j = 1,\ldots,20$ follows a $N(\mu_j, \sigma^2 I_2)$ distribution. Here we choose $\mu_j$ such that $\mu_j;\ j = 1,\ldots,20$ are evenly spaced on the unit circle. We choose $\sigma$ such that the Bayes error is 0.1, and we use 400 training data points and another 400 for tuning.

The test errors are reported in Figure 9, and the MAEs in Table 4. In this example, the soft classifier significantly dominates the others in terms of prediction accuracy. The tuned MLUM method always chooses $c = 0$ throughout the 1000 simulations. The MLUM family yields the best accuracy with $\gamma$ approximately 0.7. The MAEs for different $c$ and $\gamma$ do not differ much, possibly due to the large number of classes.




(a) The test error of Example 4 with $c$ in $\{0, 1, 10, 100, 1000\}$ and $\gamma$ tuned.

(b) The test error of Example 4 with $\gamma$ in $\{0, 0.1, \ldots, 0.9, 1\}$ and $c$ tuned.

Figure 9: Classification error rates in Example 4. The left panel shows the soft classifier works reasonably well in this example. When other choices of $c$ fail, tuned MLUM automatically selects $c = 0$ and thus keeps a good performance. In the right panel, when $\gamma = 0.7$, the classifier has a minimum error rate.

| MAE | $\gamma = 0$ | $\gamma = 0.25$ | $\gamma = 0.5$ | $\gamma = 0.75$ | $\gamma = 1$ |
|---|---|---|---|---|---|
| $c = 0$ | 0.0801 | 0.0799 | 0.0812 | 0.0813 | 0.0806 |
| | (0.001627) | (0.009215) | (0.001278) | (0.006627) | (0.007840) |
| $c = 1$ | 0.0832 | 0.0836 | 0.0829 | 0.0834 | 0.0822 |
| | (0.007646) | (0.006726) | (0.001157) | (0.008513) | (0.006180) |
| $c = 1000$ | 0.0829 | 0.0831 | 0.0821 | 0.0819 | 0.0816 |
| | (0.008364) | (0.001699) | (0.002103) | (0.003235) | (0.002697) |

Table 4: The MAEs for different $c$ and $\gamma$ in Example 4.

### 5.3 MLUM with $c = 1$ Better

Liu et al. (2011) showed that among many examples, the classifier that works the best in the LUM family appears to be either soft or hard classifiers. In the MLUM family, however, we observe that the MLUM with $c = 1$ (a new multicategory DWD) can sometimes yield the best performance in terms of classification accuracy. In particular, we explore the effect of outliers in Examples 5 and 6. In these two examples, we add a small percentage of noisy data into the originally clean data sets, and observe that soft classifiers can be very sensitive to outliers in terms of classification accuracy, while MLUMs with $c \geq 1$ appear to be quite robust.

**Example 5:** In this example, there are 95% data points from a six-class distribution with

$$P(Y = j) = 1/6, P(\mathbf{X}|Y = j) \sim N(\mu_j, 0.5^2 I_2);\ j = 1, \ldots, 6,$$

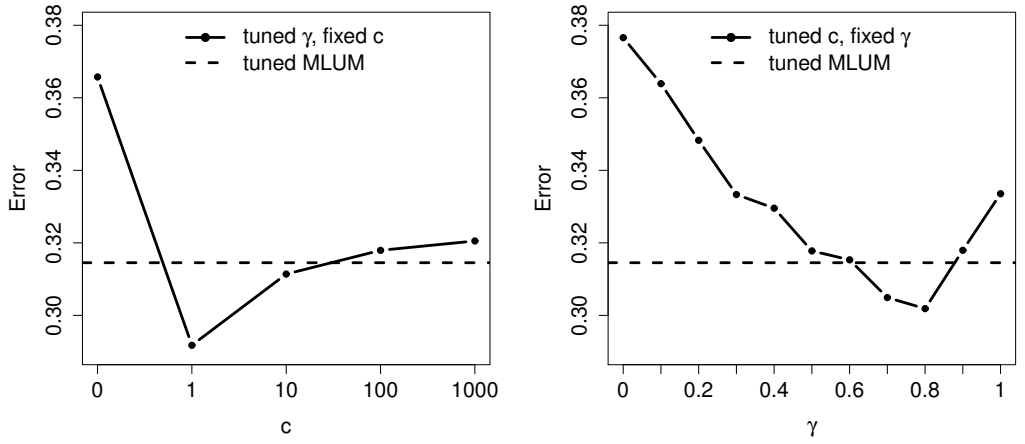
where $\mu_j = (1, \sqrt{3})^T,\ (2,0)^T,\ (1, -\sqrt{3})^T,\ (-1, -\sqrt{3})^T,\ (-2,0)^T,\ (-1, \sqrt{3})^T$, for $j = 1, \ldots, 6$, respectively. The rest 5% instances are outliers, which are uniformly distributed on the circle $x_1^2 + x_2^2 = 36$ with their labels randomly chosen. We generate independent training and tuning sets, both of size 100.

From the test errors in Figure 10, we see that MLUM with $c = 0$ is less stable than the other $c$ values, and the best classifier is $c = 1$. From the MAEs in Table 5, we see that the probability estimation with $c = 1$ is more accurate than the others.

| MAE | $\gamma = 0$ | $\gamma = 0.25$ | $\gamma = 0.5$ | $\gamma = 0.75$ | $\gamma = 1$ |
|---|---|---|---|---|---|
| $c = 0$ | 0.2328 | 0.2321 | 0.2300 | 0.2292 | 0.2303 |
| | (0.003625) | (0.006968) | (0.009631) | (0.005956) | (0.014856) |
| $c = 1$ | 0.2312 | 0.2307 | 0.2266 | 0.2241 | 0.2257 |
| | (0.010180) | (0.004729) | (0.007306) | (0.004518) | (0.001417) |
| $c = 1000$ | 0.2432 | 0.2416 | 0.2397 | 0.2397 | 0.2399 |
| | (0.005403) | (0.004657) | (0.011136) | (0.014401) | (0.006237) |

Table 5: The MAEs for different $c$ and $\gamma$ in Example 5.

**Example 6:** In this example, we add noisy observations to the data in Example 4. As the classification boundary is very sensitive to the outliers, we only have 1% noisy instances, whose distribution is the same as that of Example 5. Compared to Figure 9, Figure 11 shows that $c = 0$ is more sensitive to noise than the other $c$ values, while $c = 1$ is the most robust one. Table 6 shows that the probability estimation with $c = 1$ is the best, as in Example 5.




(a) The test error of Example 5 with $c$ in $\{0, 1, 10, 100, 1000\}$ and $\gamma$ tuned.

(b) The test error of Example 5 with $\gamma$ in $\{0, 0.1, \ldots, 0.9, 1\}$ and $c$ tuned.

Figure 10: Classification error rates in Example 5. As in Example 5, the prediction accuracy with $c = 1$ is the highest. The MLUM with $\gamma = 0.8$ works better than the others.

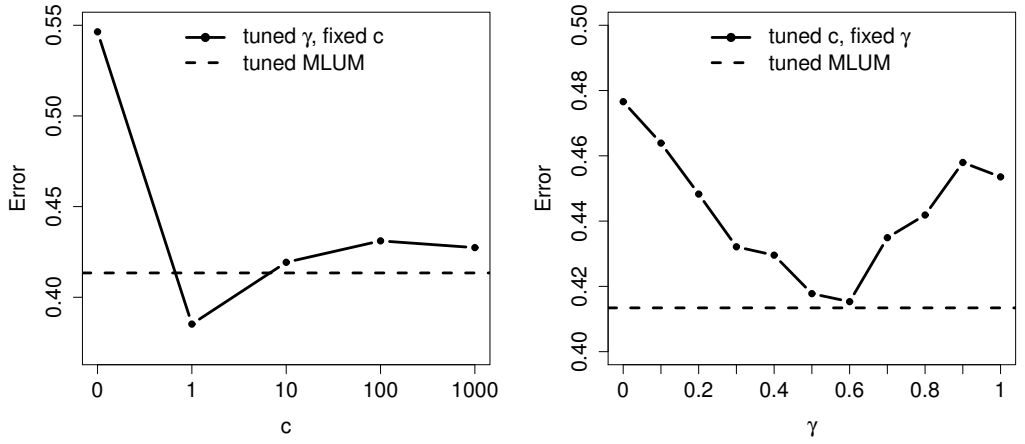
| MAE | $\gamma = 0$ | $\gamma = 0.25$ | $\gamma = 0.5$ | $\gamma = 0.75$ | $\gamma = 1$ |
|---|---|---|---|---|---|
| $c = 0$ | 0.1294 | 0.1267 | 0.1245 | 0.1258 | 0.1267 |
| | (0.007833) | (0.004012) | (0.009447) | (0.005301) | (0.006370) |
| $c = 1$ | 0.1198 | 0.1135 | 0.1131 | 0.1142 | 0.1155 |
| | (0.004456) | (0.010214) | (0.010776) | (0.007559) | (0.003054) |
| $c = 1000$ | 0.1326 | 0.1299 | 0.1276 | 0.1268 | 0.1270 |
| | (0.003148) | (0.002562) | (0.004809) | (0.004215) | (0.003149) |

Table 6: The MAEs for different $c$ and $\gamma$ in Example 6.

### 5.4 Summary of Simulation Results

Our simulated examples provide some insights on hard versus soft classification methods, and suggest that no single classifier works universally the best in all cases. Varying from soft to hard, the patterns of classification performance can differ significantly, for different settings. Our simulation studies showed different cases in which hard ($c = \infty$), soft ($c = 0$) and in-between ($c = 1$) classifiers work the best, respectively.

When the underlying conditional probability function is a step function, probability estimation can be a difficult problem for soft classifiers, and the prediction accuracy may be compromised. In Examples 1 and 2, we see that the hard classification method performs better than the others, because it directly focuses on the boundary estimation while bypassing the probability estimation. This finding is consistent with the binary case in Liu et al. (2011).




(a) The test error of Example 6 with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Example 6 with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 11: Classification error rates in Example 6. In contrast to Example 4, with noisy points in the data set, $c = 0$ performs worse than the other $c$ values. The right panel suggests that the MLUM works well with $\gamma$ around 0.6.

In contrast to Examples 1 and 2, for Examples 3 and 4, the underlying conditional probability functions are relatively smooth. Soft classifiers with $c = 0$ can build more accurate classification boundaries through estimation of the probability functions. This is also observed in the binary case in Liu et al. (2011).

One new observation is that the MLUM with $c = 1$ may work the best in some situations. This was not reported in the binary LUM cases (Liu et al., 2011). Interestingly, the soft classifier is quite vulnerable to potential outliers in the data. In Example 5, we add 5% outliers to the clean samples, and the soft classifier performs the worst among the others. In Example 4 we see that soft classifier outperforms the other methods by around 10% in terms of classification accuracy. However in Example 6 when we add only 1% outliers to the data, the classification accuracy of the soft classifier is reduced by over 40% while those of the others are just around $15-20\%$. This indicates that trying to estimate class conditional probabilities when data contain outliers may severely reduce the accuracy of the classification boundary estimation.

Another observation is that in the simulated data sets, the optimal $\gamma$ is always in or close to the recommended set $\{0,0.25,0.5,0.75,1\}$. Therefore the tuned MLUM procedure is sufficient enough to avoid tuning on the entire interval $\gamma \in [0,1]$. When it is uncertain about which classifier to use, we propose to apply our tuned MLUM method. As shown in the simulation studies, the tuned MLUM automatically selects the near optimal classifier from a rich family, and has advantages in terms of classification accuracy. Lastly, Table 7 reports comparisons of five MSVM methods, as discussed in Section 2.2, and MPLR, with our proposed tuned MLUM. The results show that the tuned MLUM delivers the most accurate classifier in most cases.

| | MSVM 1 | MSVM 2 | MSVM 3 | MSVM 4 | RMSVM | MPLR | MLUM |
|---|---|---|---|---|---|---|---|
| Ex 1 | 0.2062 | 0.2077 | **0.2019** | 0.2123 | 0.2057 | 0.2113 | 0.2021 |
| Ex 2 | 0.1223 | 0.1475 | 0.1361 | 0.1246 | 0.1190 | 0.2377 | **0.1147** |
| Ex 3 | 0.0716 | 0.0716 | 0.0719 | 0.0717 | 0.0715 | 0.0706 | **0.0701** |
| Ex 4 | 0.2411 | 0.2439 | 0.2338 | 0.2410 | 0.2411 | 0.1419 | **0.1335** |
| Ex 5 | 0.3519 | 0.3568 | 0.3638 | 0.3697 | 0.3329 | 0.3719 | **0.3145** |
| Ex 6 | 0.4528 | 0.4237 | 0.4497 | 0.4611 | 0.4323 | 0.5321 | **0.4134** |

Table 7: The classification error rates for all simulated examples, for MSVMs with Losses 1-4 defined in Section 2.2, the RMSVM, the tuned MPLR, and the tuned MLUM.

## 6. Real Data Examples

This section empirically tests the performance of the MLUM family, on two types of data sets. The first type of data examples includes several benchmark data sets available on the UCI machine learning website. The second one is a recent microarray gene expression data set in cancer research. For all the real data sets, we apply the MLUM family in the same way as in the simulation part, and we repeat the procedures 1000 times to evaluate the performance.

### 6.1 Benchmark Data

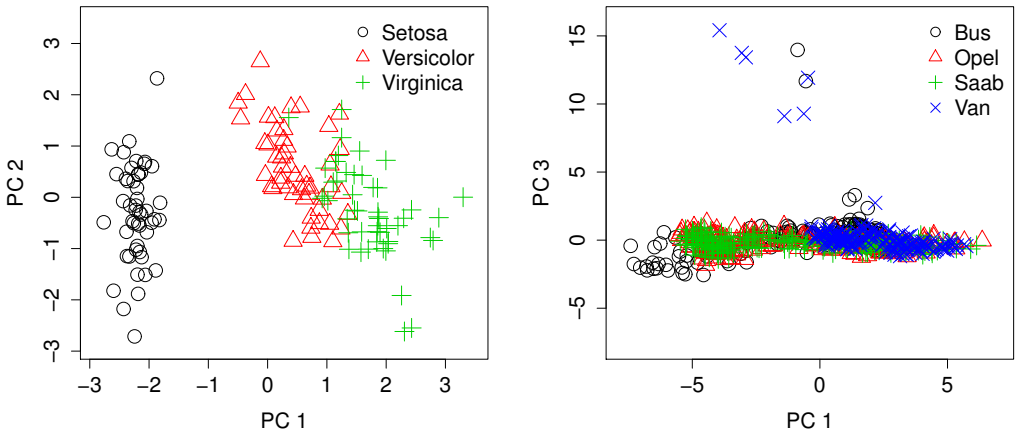
We consider seven benchmark data sets in this section: Breast Tissue (Breast), Dermatology, Image Segmentation (Image), Iris, Vehicle Silhouettes (Vehicle), Vertebral Column (Vertebral) and Wine. In each case, we perform a four-fold cross validation on roughly $4/5$ of the data set, and test the performance on the remaining $1/5$. For Iris and Image data, we apply linear learning. For the Vehicle data set, we apply the second order polynomial kernel learning. The Gaussian kernel is used for the Breast, Dermatology, Wine and Vertebral data sets, with the $\sigma$ parameter value being the median of all pairwise distances between one category and the other. For all data sets, we standardize the attributes before further analysis.

For illustration, we summarize the data sets in Table 8. As we observed in the simulation studies that potential outliers may decrease the accuracy of soft classifiers, we perform Principal Component Analysis (PCA) on each data set and examine if there are any obvious potential outliers on the PCA projected plots. From the results of the real examples, we observe that soft classifiers tend to have relatively worse performance on data sets with potential outliers. For demonstration, we report the PCA plots for the Iris and Vehicle data in Figure 12. The Iris data set doesn't appear to have any obvious outliers and the soft classifier works very well there. In the Vehicle data, there appears to have several potential outliers, and the soft classifier turns out to be the worst in terms of classification accuracy.

*Breast:* The data set contains 10 variables and 22, 21, 14, 15, 16 and 18 instances for the following 6 breast diseases respectively: carcinoma, fibro-adenoma, mastopathy, glandular, connective, and adipose. The classification errors in Figure 13 suggest that hard classification works better, and $\gamma = 1$ would be the best choice when $c$ is tuned. The tuned MLUM performs slightly worse than the RMSVM, but the difference is not large.

| Name | $n$ | $d$ | $k$ | Kernel | Outlier | Best $c$ |
|---|---|---|---|---|---|---|
| Breast | 106 | 10 | 6 | Gaussian | Yes | 1000 |
| Dermatology | 366 | 34 | 6 | Gaussian | No | 1000 |
| Image | 210 | 19 | 7 | Linear | Yes | 1 |
| Iris | 150 | 4 | 3 | Linear | No | 100 |
| Vehicle | 946 | 18 | 4 | $2^{nd}$ poly. | Yes | 1000 |
| Vertebral | 310 | 6 | 3 | Gaussian | Yes | 1 |
| Wine | 178 | 13 | 3 | Gaussian | No | 0 |

Table 8: Summary of the benchmark data sets in Section 6.1.




(a) PCA plot for the Iris data, PC1 vs PC2.

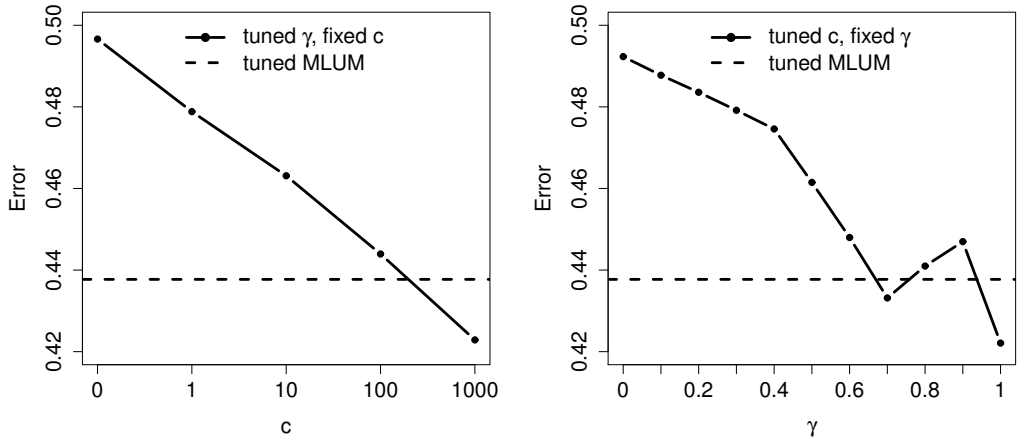
(b) PCA plot for the Vehicle data, PC1 vs PC3.

Figure 12: PCA projection plots for Iris (left panel) and Vehicle (right panel) data. The plots indicate the Iris data appear to be quite clean, while there may exist some outliers for the Vehicle data.

*Dermatology:* There are 6 classes in the Dermatology data set, namely, psoriasis, seboreic dermatitis, lichen planus, pityriasis rosea, cronic dermatitis and pityriasis rubra pilaris, with 112, 61, 72, 49, 52 and 20 instances respectively. There are 34 covariates measured for each observation. The classification results are reported in Figure 14. With $c \to \infty$, the classification accuracy is the best. In addition, $\gamma = 0.2$ performs better than other values.

*Image:* In the Image data set, there are 7 classes available with 30 observations in each group. The class names are brickface, cement, foliage, grass, path, sky and window. There are 19 covariates in the data set. We report the test errors in Figure 15. In this case, $c = 1$ performs the best, and $\gamma = 0.9$ is the optimal choice when $c$ is tuned.

*Iris:* There are three classes in the Iris data set, namely, Iris-setosa, Iris-versicolor and Iris-virginica. Within each class the number of observations is 50. There are four covariates. The test errors are reported in Figure 16. In this example, the DWD ($c = 1$) performs the worst among all classifiers,




(a) The test error of Breast data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Breast data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 13: Classification error rates in the Breast data. The hard classification method dominates the others in terms of classification accuracy, as shown in the left panel. The right panel suggests that the classification error is the best when $c$ is tuned with $\gamma = 1$.

in contrast to the simulated Example 1 where the DWD dominates. Interestingly, hard and soft classifiers perform similarly in terms of classification accuracy. For the effect of $\gamma$, the MLUM family with $\gamma = 1$ appears to work the best.

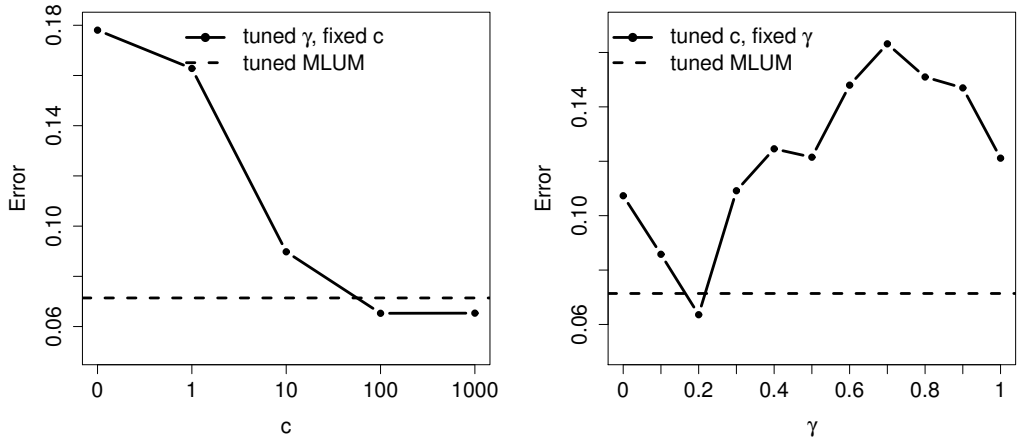
*Vehicle:* The Vehicle data set contains 4 classes of observations, namely, bus, opel, saab and van. There are $218, 212, 217, 199$ instances in the four groups, respectively. There are 18 variables associated with the data. The test errors in Figure 17 show that the hard classifier performs better than the other methods, and when $c$ is automatically chosen, $\gamma = 0.8$ dominates other values.

*Vertebral:* The three classes involved in the Vertebral data set are normal, disk hernia and spondilolysthesis. The numbers of instances are 100, 60 and 150, respectively. Six covariates are present in the data set. Figure 18 illustrates the MLUM performs quite consistently for $c \geq 1$, with the best classifier being the MLUM with $c = 1$. In terms of the behavior of $\gamma$, the optimal choice is $\gamma = 0.9$.

*Wine:* The wine data set measures 3 classes (59, 71 and 48 each) of wine with 13 attributes. In Figure 19, we see that soft classifier has the most accurate prediction result, and $\gamma = 0$ is the best choice when $c$ is tuned.

### 6.2 Glioblastoma Multiforme Cancer Data

In this section we apply the MLUM family to a cancer data set and explore its performance. The data set was previoulsy used by Verhaak et al. (2010). They investigated Glioblastoma Multiforme (GBM) cancer on the genetic data of patients from The Cancer Genome Atlas Research Network. There are 356 observations in the data set, which consists four classes, namely, Proneural, Neural, Classical and Mesenchymal. The numbers of patients in each class are 97, 56, 92 and 111, respec-




(a) The test error of Dermatology data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Dermatology data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 14: Classification error rates in the Dermatology data. The hard classifier works the best, as shown in the left panel. The right panel shows that when $c$ is tuned, $\gamma = 0.2$ works better than the others.

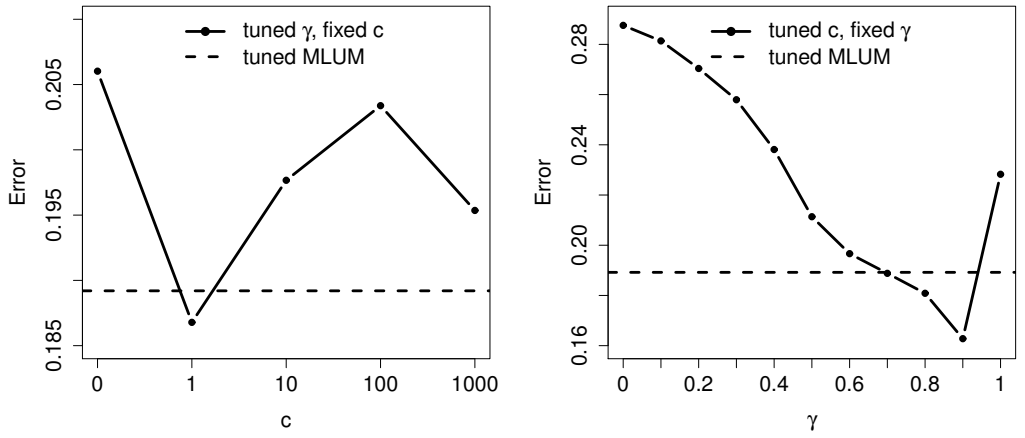
tively. There are 23285 genes in the data set. In each replication, the numbers of observations in the train, tune and test data are chosen to be roughly the same. To reduce computational cost, for each replication we pick 500 genes which have the largest median absolute deviation values in the training data set.

To visualize the data, we plot the first two projected principal component directions of the data in Figure 20. From Figure 20, we can see the classes Proneural and Mesenchymal are relatively far from each other, and Classical and Neural are in between. Biologically, Neural and Proneural are more similar to each other than the other classes. We report the MLUM results in Figure 21. From the plots, we can see that the MLUM with $\gamma = 0.75$ performs uniformly the best in terms of classification accuracy, and hard classifiers dominate the rest.

Lastly, we report the comparison of the tuned MLUM versus other methods in Table 9. Overall, the tuned MLUM performs the best, as in the simulated examples.

## 7. Discussion

Multicategory classification is commonly seen in practice. In this paper, we generalize the binary LUM family to the simultaneous multicategory classifiers, namely, the MLUM family. The MLUM is very general and it includes many popular multicategory classification methods as special cases. In particular, the MLUM family includes both soft and hard classifiers, and provides a platform to explore the transition behavior from soft to hard classification problems. In theoretical studies, we show that the MLUM is always Fisher consistent for any finite $c$, and can provide class con-




(a) The test error of Image data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

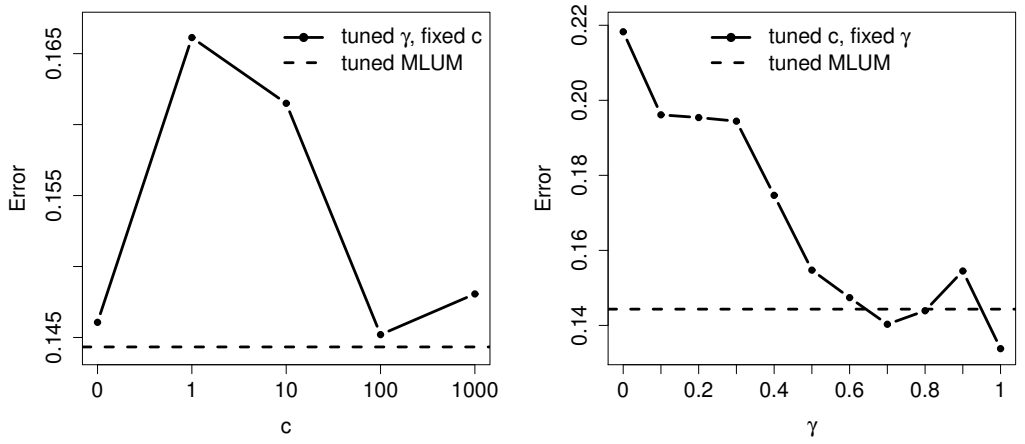
(b) The test error of Image data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 15: Classification error rates in the Image data. Left panel shows that $c = 1$ performs the best on the Image data. The right panel shows $\gamma = 0.9$ dominates other values.

| | MSVM 1 | MSVM 2 | MSVM 3 | MSVM 4 | RMSVM | MPLR | MLUM |
|---|---|---|---|---|---|---|---|
| Breast | 0.4497 | 0.4531 | 0.4391 | 0.4623 | **0.4331** | 0.4951 | 0.4377 |
| Dermatology | 0.0907 | 0.1496 | 0.1053 | 0.1398 | 0.1048 | 0.1688 | **0.0713** |
| Image | 0.1985 | 0.2083 | 0.1999 | 0.2255 | 0.2100 | 0.1978 | **0.1892** |
| Iris | 0.1557 | 0.1647 | 0.1487 | 0.1480 | 0.1605 | 0.1498 | **0.1443** |
| Vehicle | 0.2519 | 0.2535 | 0.2489 | 0.2598 | 0.2419 | 0.2680 | **0.2331** |
| Vertebral | **0.1741** | 0.1931 | 0.1860 | 0.1951 | 0.1856 | 0.3359 | 0.1766 |
| Wine | 0.0437 | 0.0743 | 0.0507 | 0.0670 | 0.0891 | 0.0348 | **0.0361** |
| GBM | 0.2291 | 0.2478 | 0.2351 | 0.2478 | 0.2394 | 0.2340 | **0.2287** |

Table 9: The classification error rates for all real data examples, for MSVMs with Losses 1-4 defined in Section 2.2, the RMSVM, the tuned MPLR and the tuned MLUM.

ditional probability estimation. We explore some asymptotic behavior of the MLUM family, and demonstrate that the convergence rate of the excess $V-$risk is closely related to the convergence rate of the classification function $\hat{\mathbf{f}}$, as well as the size of the function class space. The numerical examples show that hard and soft classifiers behave quite differently in various settings, and they help to shed some light on the choice between the two. In particular, the numerical results suggest that the underlying probability function and potential outliers may have a significant effect on the performance of soft, hard and in-between classifiers. Furthermore, for practical applications, we propose an automatic tuning procedure for the MLUM. We numerically demonstrate that the tuned MLUM outperforms several other multicategory techniques and should be a competitive addition to the existing classification toolbox.




(a) The test error of Iris data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Iris data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 16: Classification error rates in the Iris data. The left panel shows the test error of soft and hard classifiers are roughly comparable, while $c = 1$ (DWD) is the worst. The right panel indicates $\gamma = 1$ is the optimal choice, if $c$ is tuned.

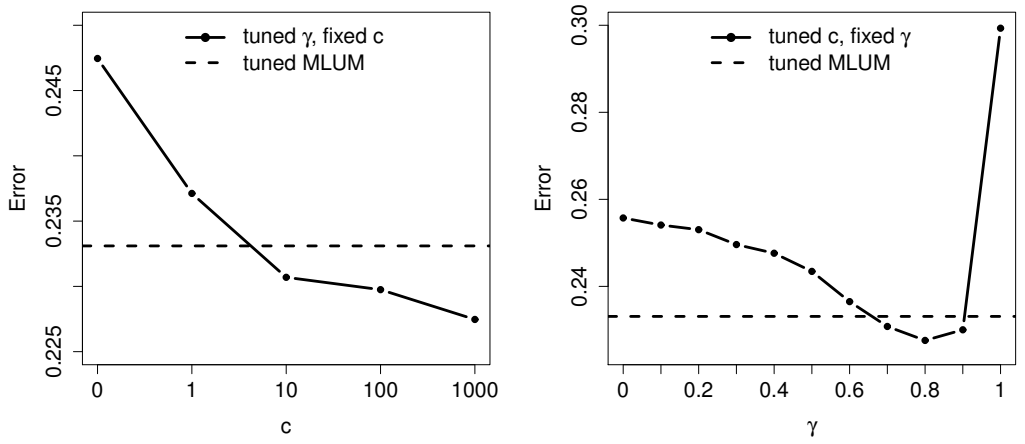
## Acknowledgments

The authors would like to thank the Action Editor Professor Saharon Rosset and three reviewers for their constructive comments and suggestions, and Derek Y. Chiang for helpful discussions. The authors are supported in part by NIH/NCI grant R01 CA-149569.

## Appendix A. Proof of Proposition 1

For brevity we only prove the kernel learning case, and the proof for the linear learning is analogous. By representer theorem, we have that $g_{j,\mathcal{H}}$ can be written as $g_{j,\mathcal{H}}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_{i,j} K(\mathbf{x}_i, \mathbf{x})$, where $K(\cdot,\cdot)$ is the associated reproducing kernel, and $\mathbf{x}_i$; $i = 1,\ldots,n$ are the training data points. With the kernel learning, the penalty becomes $\frac{1}{2}\sum_{j=1}^{k} \alpha_j^T K \alpha_j$, where $\alpha_j = (\alpha_{1,j},\ldots,\alpha_{n,j})^T$; $j = 1,\ldots,k$, and with a little abuse of notation, $K$ is the gram matrix with the $(i,j)^t h$ element $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$. One can verify that the sum-to-zero constraint here is equivalent to that $\sum_{j=1}^{k} \alpha_{i,j} = 0$, for all $i$.

Because the loss $V$ depends on $\mathbf{f}$ only through $f_i - f_j$, it suffices to prove that for a given solution $\hat{\alpha}_j = (\hat{\alpha}_{1,j},\ldots,\hat{\alpha}_{n,j})$; $j = 1,\ldots,k$ such that $\sum_{j=1}^{k} \hat{\alpha}_{i,j} = 0$ for all $i$, $\sum_{j=1}^{k} \hat{\alpha}_j^T K \hat{\alpha}_j < \sum_{j=1}^{k} (\hat{\alpha}_j - \mathbf{z})^T K (\hat{\alpha}_j - \mathbf{z})$. Here $\mathbf{z}$ is any fixed vector of length $n$ that is not $\mathbf{0}$. To this end, we see that $\sum_{j=1}^{k} \hat{\alpha}_j^T K \hat{\alpha}_j - \sum_{j=1}^{k} (\hat{\alpha}_j - \mathbf{z})^T K (\hat{\alpha}_j - \mathbf{z})$ can be simplified as $2(\hat{\alpha}_1^T + \cdots + \hat{\alpha}_k^T) K\mathbf{z} - k\mathbf{z}^T K \mathbf{z}$. Note that $\hat{\alpha}_1^T + \cdots + \hat{\alpha}_k^T = \mathbf{0}$, and $K$ is positive definite, so that $2(\hat{\alpha}_1^T + \cdots + \hat{\alpha}_k^T) K\mathbf{z} - k\mathbf{z}^T K\mathbf{z} < 0$, and this completes the proof. ■




(a) The test error of Vehicle data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Vehicle data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 17: Classification error rates in the Vehicle data. The hard classifier works better than the other ones, as shown in the left panel. The right panel suggests the best $\gamma$ is 0.8. Note that the test error significantly increases when $\gamma$ moves from 0.9 to 1.

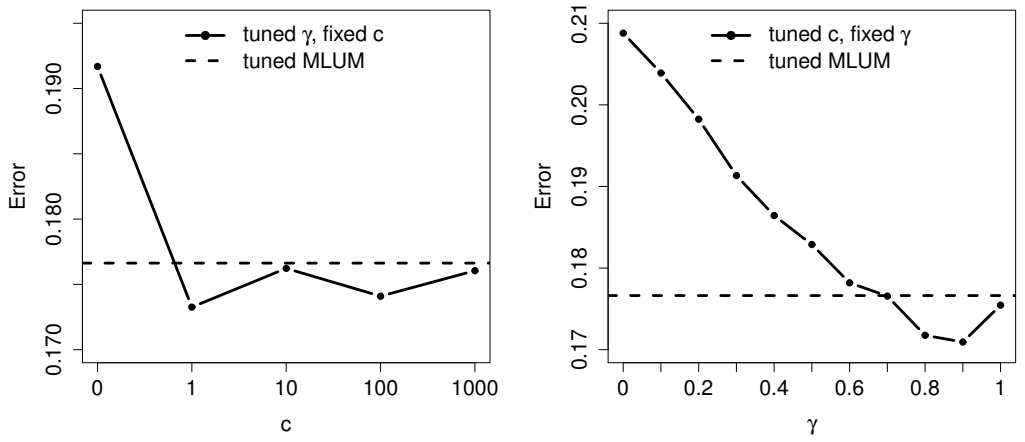
## Appendix B. Proof of Lemma 1

We prove by contradiction. Without loss of generality, assume that $P_1 > P_2$, and $f_1^* < f_2^*$. Now let $\mathbf{f} = (f_2^*, f_1^*, f_3^*, \ldots, f_k^*)$, and we want to show that $S(\mathbf{f},\mathbf{x}) < S(\mathbf{f}^*,\mathbf{x})$. One can verify that $S(\mathbf{f},\mathbf{x}) - S(\mathbf{f}^*,\mathbf{x}) = (P_2 - P_1)\big(\gamma\delta^+ + (1-\gamma)\delta^-\big)$, where $\delta^+ = \ell(f_1^*) - \ell(f_2^*)$ and $\delta^- = \ell(-f_2^*) - \ell(-f_1^*)$. Since the $\ell$ function is monotonely decreasing, and for $u < 0$ the monotonicity is strict, both $\delta^+$ and $\delta^-$ are non-negative, with at least one of them being non-zero. Thus $S(\mathbf{f},\mathbf{x}) < S(\mathbf{f}^*,\mathbf{x})$, and $\mathbf{f}^*$ is not the minimizer of $S$. ■

## Appendix C. Proof of Theorem 2

Assume, without loss of generality, that $P_1 > P_2 > \cdots > P_k$, and we need to show that $f_1^* > f_2^* \geq \cdots \geq f_k^*$. We prove by contradiction. Because of Lemma 1, $f_1^* \geq f_2^*$. Suppose $f_1^* = f_2^*$, and let $\mathbf{f} = (f_1^* + \varepsilon, f_2^* - \varepsilon, f_3^*, \ldots, f_k^*)$, where $\varepsilon > 0$ is sufficiently small. We show that $S(\mathbf{f},\mathbf{x}) < S(\mathbf{f}^*,\mathbf{x})$. For simplicity in notation, let $f = f_1^* = f_2^*$. After some calculation we have

$$\begin{aligned} S(\mathbf{f},\mathbf{x}) - S(\mathbf{f}^*,\mathbf{x}) = (1-\gamma)[\delta_1 - \delta_2] + P_1\big[\gamma(-\delta_4) + (1-\gamma)(-\delta_1)\big] \\ + P_2\big[\gamma(\delta_3) + (1-\gamma)(\delta_2)\big] \end{aligned} \tag{13}$$




(a) The test error of Vertebral data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Vertebral data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 18: Classification error rates in the Vertebral data. In the left panel, we can see the MLUM classifiers work roughly the same for $c \geq 1$, with $c = 1$ being the optimal. The best $\gamma$ is 0.9, as is suggested by the right panel.

where

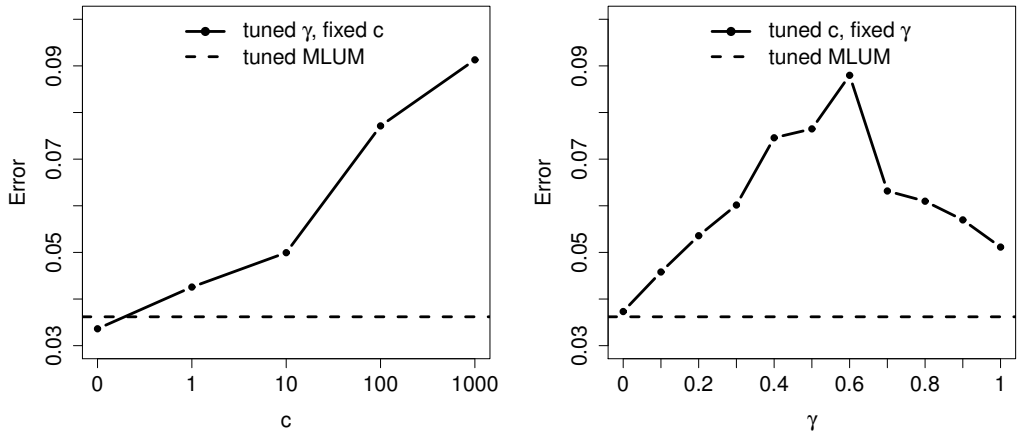
$$\begin{cases} \delta_1 = \ell(-f-\varepsilon) - \ell(-f), \\ \delta_2 = \ell(-f) - \ell(-f+\varepsilon), \\ \delta_3 = \ell(f-\varepsilon) - \ell(f), \\ \delta_4 = \ell(f) - \ell(f+\varepsilon). \end{cases}$$

Since $\ell(\cdot)$ is differentiable, we may rewrite the above equations as:

$$\begin{cases} \delta_1 = -\varepsilon\ell'(-f) + o(\varepsilon), \\ \delta_2 = -\varepsilon\ell'(-f) + o(\varepsilon), \\ \delta_3 = -\varepsilon\ell'(f) + o(\varepsilon), \\ \delta_4 = -\varepsilon\ell'(f) + o(\varepsilon). \end{cases}$$

and so (13) becomes $\varepsilon(P_1 - P_2)[\gamma\ell'(f) + (1-\gamma)\ell'(-f)] + o(\varepsilon)$. Because $\ell(\cdot)$ is convex and strictly monotonely decreasing, $\ell'(\cdot) < 0$, thus $S(\mathbf{f},\mathbf{x}) - S(\mathbf{f}^*,\mathbf{x}) < 0$. ■

**Remark 3.** The reason why the hinge loss is not Fisher consistent when $\gamma > 1/2$ becomes more clear from the proof of Theorem 2. For the hinge loss, we should replace the derivatives with the corresponding left/right derivatives in the previous argument. When $f = 1$, $\delta_4 = 0$, $|P_1(1-\gamma)\delta_1|$ does not necessarily dominate $P_2\big[\gamma(\delta_3) + (1-\gamma)(\delta_2)\big]$ with $\gamma > 1/2$. By allowing differentiability in the loss function, we overcome the disadvantage of non-Fisher consistency.




(a) The test error of Wine data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of Wine data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 19: Classification error rates in the Wine data. The left panel suggests that soft classification method performs better in terms of classification accuracy than the others. With $\gamma = 0$, the MLUM method works the best than the other $\gamma$ values, which is shown in the right panel.

## Appendix D. Proof of Theorem 3

Due to the constraint $\sum_{j=1}^{k} f_j = 0$, the degree of freedom for $S$ in (8) is only $k-1$. Rewrite $S$ as

$$S = (1-\gamma)\sum_{j=1}^{k} \ell(-f_j) + \sum_{j=1}^{k} P_j\big[\gamma\ell(f_j) - (1-\gamma)\ell(-f_j)\big].$$

Let the first $k-1$ components of $\mathbf{f}$ be the free parameters. Take partial derivative of $S$ with respect to $f_1,\ldots,f_{k-1}$, and we have

$$0 = \frac{\partial S}{\partial f_j}|_{\mathbf{f}=\mathbf{f}^*} = -F^*(j) + F^*(k) + P_j E^*(j) - P_k E^*(k);\ j = 1,\ldots,k-1,$$

where $E^*$ and $F^*$ are defined in an obvious manner. Let $\mathbf{P}_{k-1} = (P_1,\ldots,P_{k-1})^T$, we can rewrite the above equations as $M\mathbf{P}_{k-1} - \mathbf{b} = 0_{k-1}$, or $M\mathbf{P}_{k-1} = \mathbf{b}$ where $\mathbf{b} = \big([(1-\gamma)\ell'(-f_1^*) + \gamma\ell'(f_k^*)],\ldots,[(1-\gamma)\ell'(-f_{k-1}^*) + \gamma\ell'(f_k^*)]\big)^T$ and $M = \mathrm{diag}\big(E^*(1),\ldots,E^*(k-1)\big) + E^*(k)J_{k-1}$. Here $J_{k-1}$ is the $k-1$ by $k-1$ matrix with every element being 1. Since $E(\cdot) < 0$, $1 + E^*(k)\sum_{j=1}^{k-1} E^*(j) > 0$, and we apply the Sherman-Morrison formula (which guarantees the invertibility of $M$), to obtain $\mathbf{P}_{k-1} = M^{-1}\mathbf{b}$. Let $P_k = 1 - \sum_{j=1}^{k-1} P_j$. The equations (9) follow after some simplification and substituting $\hat{\mathbf{f}}$ for $\mathbf{f}^*$. Note that from the form of (9), we conclude the choice of the free parameters in $\mathbf{f}$ is not essential. ■

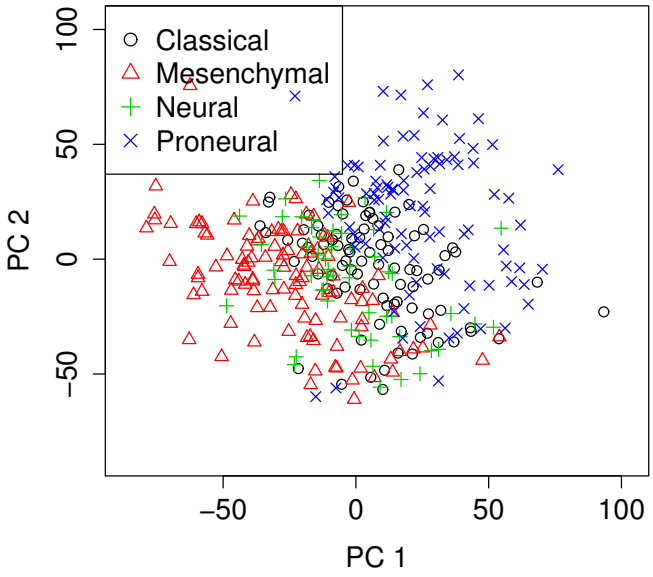



Figure 20: PCA plot of the GBM data, PC 1 vs PC 2.

## Appendix E. Proof of Theorem 4

By definition,

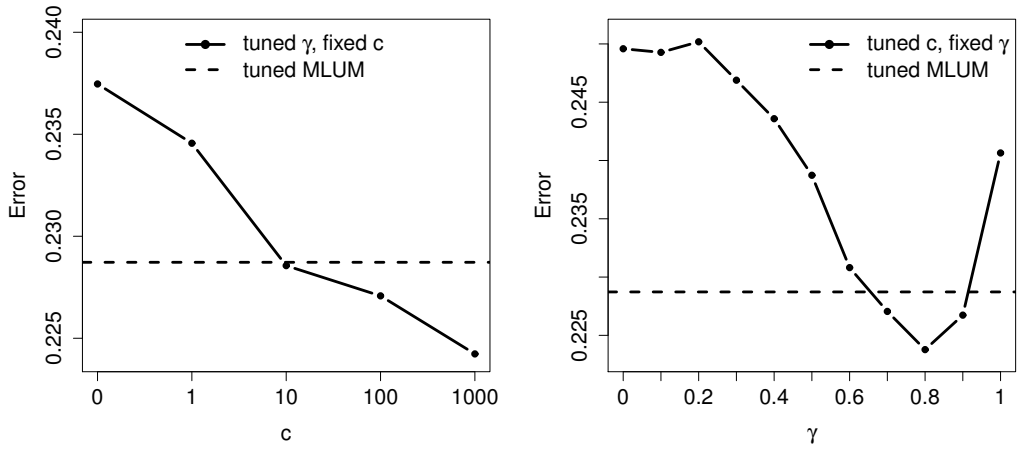
$$\Delta Q(\mathbf{P},\mathbf{f}) = Q(\mathbf{P},\mathbf{f}) - Q^*(\mathbf{P}).$$

We can rewrite the RHS of the display as

$$(1-\gamma)\sum_{j=1}^{k}\ell(-f_j) + \sum_{j=1}^{k}P_j[\gamma\ell(f_j) - (1-\gamma)\ell(-f_j)]$$
$$-(1-\gamma)\sum_{j=1}^{k}\ell(-f_j^*) - \sum_{j=1}^{k}P_j[\gamma\ell(f_j^*) - (1-\gamma)\ell(-f_j^*)].$$

With adding and subtracting, rearrange to obtain that the above display is equivalent to

$$(1-\gamma)\sum_{j=1}^{k}[\ell(-f_j) - \ell(-f_j^*) - \ell'(-f_j^*)(-f_j+f_j^*)] + (1-\gamma)\sum_{j=1}^{k}\ell'(-f_j^*)(-f_j+f_j^*)$$
$$+\sum_{j=1}^{k}P_j[\gamma\{\ell(f_j) - \ell(f_j^*) - \ell'(f_j^*)(f_j-f_j^*)\} - (1-\gamma)\{\ell(-f_j) - \ell(-f_j^*) - \ell'(-f_j^*)(-f_j+f_j^*)\}]$$
$$+\sum_{j=1}^{k}P_j[\gamma\ell'(f_j^*)(f_j-f_j^*) - (1-\gamma)\ell(-f_j^*)(-f_j+f_j^*)].$$




(a) The test error of GBM data with $c$ in $\{0,1,10,100,1000\}$ and $\gamma$ tuned.

(b) The test error of GBM data with $\gamma$ in $\{0,0.1,\ldots,0.9,1\}$ and $c$ tuned.

Figure 21: For the GBM data, hard classifier works the best, as shown in the left panel. The right panel illustrates that the optimal choice of $\gamma$ is 0.8.

In combination with the Bregman divergence, observe that the above is essentially RHS of Theorem 4 plus $(1-\gamma)\sum_{j=1}^{k}\ell'(-f_j^*)(-f_j+f_j^*)+\sum_{j=1}^{k}P_j[\gamma\ell'(f_j^*)(f_j-f_j^*)-(1-\gamma)\ell(-f_j^*)(-f_j+f_j^*)]$, so it suffices to show the latter, denoted by $W$, equals 0.

Using the same notation as in Theorem3, we have

$$W=\sum_{j=1}^{k}[P_j\gamma\ell'(f_j^*)(f_j-f_j^*)+(1-P_j)(1-\gamma)\ell'(-f_j^*)(-f_j+f_j^*)]$$
$$=\sum_{j=1}^{k}[-F^*(j)+P_jE^*(j)](f_j-f_j^*).$$

Because of the sum-to-zero constraint, the above display is equivalent to

$$W=\sum_{j=1}^{k-1}[-F^*(j)+F^*(k)+P_jE^*(j)-P_kE^*(k)](f_j-f_j^*).$$

Note that $-F^*(j)+F^*(k)+P_jE^*(j)-P_kE^*(k)$ is just the $j^{th}$ element of $\bigtriangledown Q^*(\mathbf{P})|_{\mathbf{f}=\mathbf{f}^*}$, choosing the first $k-1$ elements in $\mathbf{f}$ as free parameters and taking partial derivatives. Thus $W=0$, and the desired result follows. ■

## Appendix F. Proof of Theorem 5

Expand the Bregman divergence in Theorem 4 into the Taylor series form:

$$\Delta Q(\mathbf{P},\mathbf{f}) = \sum_{i=1}^{k} \big[P_i\gamma[\frac{\ell^{(2)}(f_i^*)}{2}(\hat{f}_i - f_i^*)^2 + o((\hat{f}_i - f_i^*)^2)]$$
$$+(1-P_i)(1-\gamma)[\frac{\ell^{(2)}(-f_i^*)}{2}(\hat{f}_i - f_i^*)^2 + o((\hat{f}_i - f_i^*)^2)]\big].$$

Note that the second order derivative of $\ell$ is bounded for every $a$ and finite $c$. Because $\tau_i \ll \mu, i = 1,\ldots,k$, we can multiply both sides by $n^{2q}$, and take expectation to obtain

$$n^{2q}\Delta Q(\hat{\mathbf{f}}^n) = O\big(\int_{\mathbf{X},Y} |\sup_{1\le j\le k} T_j|^2 dD(\mathbf{X},Y)\big).$$

Because of Assumption B, the RHS is bounded, and the desired result follows. ■

## Appendix G. Proof of Theorem 6

Note that $(\hat{f}_i - f_i^*)^2 - (f_i^H - f_i^*)^2 = (\hat{f}_i - f_i^H)(\hat{f}_i + f_i^H - 2f_i^*)$. $(\hat{f}_i - f_i^H)$ is $n^q$ consistent, and $|\hat{f}_i + f_i^H - 2f_i^*| \le |\hat{f}_i - f_i^H| + 2(|f_i^H| + |f_i^*|) \to 2(|f_i^H| + |f_i^*|)$. The rest of the proof is analogous to that of Theorem 5. ■

## References

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

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.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, 1992. ISBN 0-89791-497-X.

C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

K. Crammer, Y. Singer, N. Cristianini, J. Shawe-taylor, and B. Williamson. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

H. Huang, Y. Liu, Y. Du, C. Perou, D. N. Hayes, M. Todd, and J. S. Marron. Multiclass distance weighted discrimination. *Journal of Computational and Graphical Statistics*, 2013. Forthcoming.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

X. Lin, G. Wahba, D. Xiang, F. Gao, R. Klein, and B. Klein. Smoothing spline anova models for large data sets with bernoulli observations and the randomized gacv. *Annals of Statistics*, 28(6): 1570–1600, 2000.

Y. Liu. Fisher consistency of multicategory support vector machines. In *Eleventh International Conference on Artificial Intelligence and Statistics*, pages 289–296, 2007.

Y. Liu and X. Shen. Multicategory ψ-learning. *Journal of the American Statistical Association*, 101 (474):500–509, 2006.

Y. Liu and M. Yuan. Reinforced multicategory support vector machines. *Journal of Computational and Graphical Statistics*, 20(4):901–919, 2011.

Y. Liu, X. Shen, and H. Doss. Multicategory ψ-learning and support vector machine: computational tools. *Journal of Computational and Graphical Statistics*, 14(1):219–236, 2005.

Y. Liu, H. H. Zhang, and Y. Wu. Soft or hard classification? large margin unified machines. *Journal of the American Statistical Association*, 106(493):166–177, 2011.

J. S. Marron, M. Todd, and J. Ahn. Distance weighted discrimination. *Journal of the American Statistical Association*, 102(480):1267–1271, 2007.

S. Y. Park, Y. Liu, D. Liu, and P. Scholl. Multicategory composite least squares classifiers. *Statistical Analysis and Data Mining*, 3(4):272–286, 2010.

X. Shen and W. H. Wong. Convergence rate of sieve estimates. *Annals of Statistics*, 22(2):580–615, 1994.

X. Shen, G. C. Tseng, X. Zhang, and W. H. Wong. On ψ-learning. *Journal of the American Statistical Association*, 98(463):724–734, 2003.

I. Steinwart and C. Scovel. Fast rates for support vector machines using gaussian kernels. *Annals of Statistics*, 35(2):575–607, 2007.

Y. Tang and H. H. Zhang. Multiclass proximal support vector machines. *Journal of Computational and Graphical Statistics*, 15(2):339–355, 2006.

A. Tewari and P. L. Bartlett. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8:1007–1025, 2007.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

R. G. Verhaak, K. A. Hoadley, E. Purdom, V. Wang, Y. Qi, M. D. Wilkerson, C. R. Miller, L. Ding, T. Golub, J. P. Mesirov, G. Alexe, M. Lawrence, M. O'Kelly, P. Tamayo, B. A. Weir, S. Gabriel, W. Winckler, S. Gupta, L. Jakkula, H. S. Feiler, J. G. Hodgson, C. D. James, J. N. Sarkaria, C. Brennan, A. Kahn, P. T. Spellman, R. K. Wilson, T. P. Speed, J. W. Gray, M. Meyerson, G. Getz, C. M. Perou, D. N. Hayes, and Cancer Genome Atlas Research Network. Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in pdgfra, idh1, egfr, and nf1. *Cancer Cell*, 17(1):98–110, 2010.

G. Wahba. Support vector machines, reproducing kernel hilbert spaces and the randomized GACV. In *Advances in Kernel Methods Support Vector Learning*, pages 69–87. MIT Press, 1999.

G. Wahba. Soft and hard classification by reproducing kernel hilbert space methods. In *Proceedings of the National Academy of Sciences*, volume 99, pages 16524–16530, 2002.

J. Wang, X. Shen, and Y. Liu. Probability estimation for large margin classifiers. *Biometrika*, 95 (1):149–167, 2008.

L. Wang and X. Shen. On $l_1$-norm multi-class support vector machines: methodology and theory. *Journal of the American Statistical Association*, 102(478):595–602, 2007.

J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the Seventh European Symposium on Artificial Neural Networks*, volume 4, pages 219–224, 1999.

T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5:975–1005, 2004.

Y. Wu and Y. Liu. Robust truncated-hinge-loss support vector machines. *Journal of the American Statistical Association*, 102(479):974–983, 2007.

T. Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 32:56–85, 2004a.

T. Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5:1225–1251, 2004b.

J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005.

J. Zhu, H. Zou, S. Rosset, and T. Hastie. Multi-class adaboost. *Statistics and Its Interface*, 2(3): 349–360, 2009.

H. Zou, J. Zhu, and T. Hastie. New multicategory boosting algorithms based on multicategory fisher-consistent losses. *Annals of Applied Statistics*, 2(4):1290–1306, 2008.

Journal of Machine Learning Research 14 (2013) 1387-1415 Submitted 1/11; Revised 12/12; Published 5/13

# Finding Optimal Bayesian Networks Using Precedence Constraints*

**Pekka Parviainen** PEKKAPA@KTH.SE
*Science for Life Laboratory*
*School of Computer Science and Communication*
*Royal Institute of Technology (KTH)*
*Tomtebodavägen 23A*
*17121 Solna, Sweden*

**Mikko Koivisto** MIKKO.KOIVISTO@CS.HELSINKI.FI
*Helsinki Institute for Information Technology*
*Department of Computer Science*
*University of Helsinki*
*Gustaf Hällströmin katu 2b*
*00014 Helsinki, Finland*

**Editor:** Chris Meek

## Abstract

We consider the problem of finding a directed acyclic graph (DAG) that optimizes a decomposable Bayesian network score. While in a favorable case an optimal DAG can be found in polynomial time, in the worst case the fastest known algorithms rely on dynamic programming across the node subsets, taking time and space $2^n$, to within a factor polynomial in the number of nodes $n$. In practice, these algorithms are feasible to networks of at most around 30 nodes, mainly due to the large space requirement. Here, we generalize the dynamic programming approach to enhance its feasibility in three dimensions: first, the user may trade space against time; second, the proposed algorithms easily and efficiently parallelize onto thousands of processors; third, the algorithms can exploit any prior knowledge about the precedence relation on the nodes. Underlying all these results is the key observation that, given a partial order $P$ on the nodes, an optimal DAG compatible with $P$ can be found in time and space roughly proportional to the number of ideals of $P$, which can be significantly less than $2^n$. Considering sufficiently many carefully chosen partial orders guarantees that a globally optimal DAG will be found. Aside from the generic scheme, we present and analyze concrete tradeoff schemes based on parallel bucket orders.

**Keywords:** exact algorithm, parallelization, partial order, space-time tradeoff, structure learning

## 1. Introduction

During the last two decades, Bayesian networks (BNs) have become one of the most popular and powerful frameworks for modeling various aspects of intelligent reasoning, such as degrees of belief, causality, and responsibility (Pearl, 1988, 2000; Chockler and Halpern, 2004). While the conceptual basis of BNs can be regarded as satisfactory to a large extent, there remain computational bottlenecks that currently limit the utilization of BNs in large and combinatorially complex do-

*. A preliminary version of this work appeared in UAI 2009 (Parviainen and Koivisto, 2009). Part of the work was done while the first author (P.P.) was affiliated with Helsinki Institute for Information Technology at University of Helsinki. All correspondence should be addressed to the first author.

©2013 Pekka Parviainen and Mikko Koivisto.

mains. Underlying many of these bottlenecks is the graphical structure of the model, a *directed acyclic graph* (DAG). In particular, when one is supposed to learn a BN from data (Verma and Pearl, 1990; Spirtes and Glymour, 1991; Cooper and Herskovits, 1992; Heckerman et al., 1995), in principle, one has to exhaust the space of all possible DAGs, which can be enormous and does not easily factorize due to the acyclicity constraint. Indeed, when formalized in a natural manner, the problem is known to be NP-hard (Chickering, 1996; Chickering et al., 2004). Consequently, much of the machine learning research on BNs has focused on tractable special cases or other restrictive assumptions and, of course, on various heuristics.

However, the continued increase in computational resources and the advances in algorithmic techniques have recently turned many researchers' attention to exact means for learning BNs from data. Common to such endeavors is that the problem is cast as optimization (or sometimes as integration) of a scoring function that assigns each possible DAG a real number reflecting how well the DAG fits the given data. Furthermore, it is assumed that the scoring function decomposes into a sum of local terms, each local term depending on a child node and its parent nodes. For this optimization problem, techniques similar to the classic dynamic programming (DP) treatment of the traveling salesman problem (Bellman, 1962; Held and Karp, 1962) have yielded exponential-time algorithms that solve instances of up to around 30 nodes with feasible worst-case runtime guarantees (Ott and Miyano, 2003; Koivisto and Sood, 2004; Silander and Myllymäki, 2006; Singh and Moore, 2005). Note that, while heuristic search algorithms may *often* find an optimal DAG in instances of this size, that certainly does not happen *always* and there is usually no practical way to verify or falsify a claim of optimality. Exact algorithms completely avoid that major uncertainty concerning the quality of the algorithm's output.[1] Besides, they provide tools for the design and analysis of heuristic methods that better scale up to larger instances. There are obvious interests in extending the scope of exact algorithms.

The existing DP algorithms suffer from two major shortcomings: they have huge memory requirements and they do not parallelize efficiently. For example, a streamlined implementation of the DP algorithm, by Silander and Myllymäki (2006), is able to handle 29 nodes in around 10 hours, but, needing almost 100 gigabytes of hard disk aside a main memory of a few gigabytes. As such, the algorithm cannot handle larger instances. However, supposing the memory requirement was not an issue and that the algorithm could be run in parallel on thousands of processors, much larger instances could be solved. For example, a 40-node instance would take just about the same 10 hours if given $2^{11} = 2048$ processors, or about a week if given one hundred processors. Unfortunately, it is notoriously difficult to save space in related DP algorithms without an increase in the running time (Bellman, 1962; Bodlaender et al., 2006, 2012). Thus, a plausible goal is to trade as little time as possible for a fair amount of space and parallelization capacity.

In this article, we introduce new algorithmic schemes that address the issues of memory requirement and parallelization of the DP algorithms for the BN learning problem. Our contributions stem from the following observation: Suppose we know a priori some *precedence constraints* that an optimal DAG we are searching for must obey; that is, if node $u$ is constrained to precede node

1. The reader may wonder whether the search for an optimal DAG is important at all. Might not noise in the data, limitations of the score function, etc., dampen any advantage of the optimality procedure over, say, heuristic local search? Yes, they might. When that is the case, the research, indeed, ought to focus on removing the corresponding bottlenecks, that is, improve data quality, the model, etc., which is a matter of more practical case-by-case studies. But we also believe there is room for research in settings where "optimal" is truly more desirable than "arbitrary", and research on BNs is not an exception.

$v$, then the DAG must not contain a directed path from $v$ to $u$. In an extreme case, the precedence constraints specify a linear order on the nodes, and more generally, the constraints specify a partial order. The key observation is that the DP algorithms can be extended to exploit the precedence constraints to save both time and space. Specifically, the time and space requirements can be made to grow roughly linearly in the number of node subsets that are closed under taking predecessors, called *ideals* (or downsets) of the partial order. As the number of ideals can be much smaller than the number of all node subsets, depending on the width of the partial order, there is potential for a significant improvement over the existing DP algorithms, which are ignorant of precedence constraints. To realize this potential, we need to consider partial orders that have only few ideals. The second part of our observation addresses this requirement: Instead of assuming a single given partial order, we may consider multiple partial orders that together cover all possible DAGs, that is, every DAG is compatible with at least one of the partial orders. Each partial order amounts to a subproblem that can be solved independently of the others, until only finally the best of the solutions to the subproblems is returned. Because of the independence of the subproblems, the computations can be run in parallel on as many processors as there are partial orders. We are left with the freedom to choose either many partial orders, each with only few ideals (the extreme is to consider all linear orders), or just a few partial orders, each with many ideals, or something in between. This freedom allows us to trade time for space and parallelization in a smooth fashion, to adapt to the available resources.

We begin the remainder of this article in Section 2 by formulating the optimization problem in question more carefully and by reviewing the basic DP algorithm. Section 3 illustrates the problem setting by describing a simple scheme, called the *two-bucket scheme*, that enables trading time for space and parallelization. While that scheme is *per se* rather inefficient, it serves as a base for the developments that follow in later sections. Specifically, Section 4 combines the two-bucket scheme with divide and conquer to get efficient tradeoff in the small-space regime. While the resulting algorithms are, admittedly, of merely theoretical interest, they connect to and extend what is known about the polynomial-space solvability of related classic problems, such as the traveling salesman problem (Savitch, 1970; Gurevich and Shelah, 1987; Björklund and Husfeldt, 2008; Bodlaender et al., 2006, 2012). Section 5, on the other hand, generalizes the two-bucket scheme into a generic partial order approach, constituting the main conceptual and technical contribution of this work. The generic approach being quite abstract, its more concrete implications are derived in Section 6 for a particular class of partial orders, namely (parallel compositions of) bucket orders. Some of the combinatorial analyses concerning bucket orders build on our on-going work on related permutation problems—some results have been published in a preliminary form in a conference proceedings (Koivisto and Parviainen, 2010) that we will cite in Section 6. Finally, in Section 7, we summarize our main findings and discuss how they advance the state of the art in learning BNs from data.

After the publication of a preliminary version of this work (Parviainen and Koivisto, 2009), several notable related results have been announced by other groups. Branch-and-bound methods exploiting local constraints are proposed by de Campos et al. (2009, 2011) and Etminani et al. (2010). Jaakkola et al. (2010) and Cussens (2011) have developed integer linear programming techniques and achieved rather encouraging empirical results. Malone et al. (2011) and Yuan et al. (2011) propose tighter implementations of the DP algorithm, yielding moderate savings in memory and time usage. Finally, Tamada et al. (2011) present a parallelization scheme with somewhat involved communication routines. We discuss the merits and limitations of these methods and their relation to our work more thoroughly in Section 7.

## 2. Preliminaries

This section presents the basic terminology and formulations needed in later sections. In particular, we formalize the computational problem of finding an optimal Bayesian network, accompanied with some remarks concerning the representation of the input. Then we review a dynamic programming algorithm that sets the technical and conceptual baseline for the developments in Sections 3–6.

### 2.1 The Optimal Bayesian Network Problem

A Bayesian network is a multivariate probability distribution that obeys a structural representation in terms of a directed acyclic graph (DAG) and a corresponding collection of univariate conditional probability distributions. For our purposes, it is crucial to treat the DAG, that is, the *network structure*, explicitly, whereas the conditional probabilities will enter our formalism only implicitly. Formally, a DAG is a pair $(N, A)$, where $N$ is the *node set* and $A \subseteq N \times N$ is the *arc set*. A node $u$ is said to be a *parent* of a node $v$ if the arc $uv$ is in $A$. The *parent set* of a node $v$ consists of the parents of $v$ and is denoted by $A_v$. When there is no ambiguity about the node set we identify the DAG with its arc set. We denote the cardinality of $N$ by $n$. The *number of parents* or *indegree* of node $v$ in $A$ is simply $|A_v|$. A node that is not a parent of any node is called a *sink* of the DAG.

We formulate the task of learning a Bayesian network from data as a generic optimization problem over DAGs on a given node set $N$. Specifically, we assume that each DAG $A$ is associated with a real number $f(A)$ that specifies how well the Bayesian networks with structure $A$ fit the given data. In particular, the *scoring function* $f$ can take any of the various forms derived under different paradigms for statistical inference, chiefly, the Bayesian, the maximum-likelihood, and the minimum description length paradigms. For examples of concrete scoring functions and their justifications, see, for instance, de Campos's (2006) review and the next paragraph. The optimization problem becomes algorithmically interesting when the function $f$ has some structure. To this end, we make the usual assumption that the scoring function is *decomposable*, that is, for each node $v$ there exist a *local scoring function* $f_v$ such that

$$f(A) = \sum_{v \in N} f_v(A_v)\,,$$

for all DAGs $A$ on $N$; each $f_v$ is a function from the subsets of $N \setminus \{v\}$ to real numbers. We call $f(A)$ and $f_v(A_v)$ the *score* of $A$ and the *(local) score* of $A_v$, respectively.

**Example 1 (the Bayesian Dirichlet (BD) score)** Heckerman et al. (1995) define the *Bayesian Dirichlet (BD) metric* as the joint probability distribution over the DAG and the data:

$$p(A) \prod_{v \in N} \prod_{x=1}^{q_v} \frac{\Gamma(m'_{ux})}{\Gamma(m'_{ux} + m_{ux})} \prod_{y=1}^{r_v} \frac{\Gamma(m'_{uxy} + m_{uxy})}{\Gamma(m'_{uxy})}\,,$$

where $p(A)$ is the prior of the structure $A$, $r_v$ is the number of possible values of node $v$ and $q_v$ the number of possible value configurations of the parents of $v$ in $A$. Furthermore, $m_{uxy}$ is the number of data records in which $u$ has value $x$ and the parents of $u$ in $A$ have value configuration $y$. Here we assume some arbitrary but fixed labeling of the values and value configurations by numbers $1, 2, 3, \ldots$. The numbers $m'_{uxy}$ are nonnegative reals specified by the modeler, and $m_{ux}$ and $m'_{ux}$ are obtained by taking the sum of the $m_{uxy}$ and $m'_{uxy}$, respectively, over the range of $y$. The gamma

function $\Gamma$ appears in the expression as Dirichlet priors on the local conditional distributions are marginalized out.

The *BD score* is obtained by taking a logarithm of the BD metric. Whether the BD score is decomposable or not depends on the choice of the prior $p(A)$. Often the prior is assigned such that it factorizes into a product $\prod_v \rho_v(A_v)$, with, for instance $\rho_v(A_v) = c\kappa^{|A_v|}$ for some constants $c$ and $\kappa$ independent of $A_v$ but possibly dependent on $v$ (Heckerman et al., 1995). With such a prior, the BD score is easily seen to yield a decomposable scoring function.

**Definition 1 (the OBN problem)** Given a decomposable scoring function $f$ as input, the *optimal Bayesian network* problem is to output a DAG $A$ that maximizes $f(A)$.

Concerning the representation of the problem input, some remarks are in order. First, while the notion of decomposability concerns the mere existence of local scoring functions, we naturally assume that the functions $f_v$ are given explicitly as input. In practice, the values $f_v(A_v)$ are computed for every relevant $A_v$ based on the data and the chosen scoring function. Second, we allow $f_v(A_v)$ to take the value $-\infty$ to indicate that $A_v$ cannot be the parent set of $v$. In fact, we assume that a collection of *potential parent sets*, denoted as $\mathcal{F}_v$, is given as input, with the understanding that outside that collection the values are $-\infty$. Third, we remark that the size of $\mathcal{F}_v$ may often be much less than the theoretical maximum $2^{n-1}$, due to several potential reasons:

(a) The *maximum number of parents* is set to $k$, a parameter specified by the modeler.

(b) The parents are assumed to be contained in a predetermined (small) *set of candidates*.

(c) One can safely *ignore a parent set that has a subset with a better score* (observed by computing the score or by analytical bounds specific to the scoring function).

We note that (a) and (b) are often assumed by the modeller but do not hold in general, and are not assumed in the sequel, whereas (c) always holds. Common to (a) and (b) is that they yield a *downward closed* collection of parent sets, that is, a collection that is closed with respect to set inclusion. While the same does not hold for (c), it is plausible to expect that the pruned collection is not much smaller than the *downward closure* of the collection, obtained by taking all members of the collection and their subsets: In theory, the *downward closure* can be larger than the pruned collection by a factor of about $2^k$, where $k$ is the size of the largest parent set; for a small constant $k$, this factor is not large. In practice, however, we have observed significantly smaller factors, typically not exceeding 2.[2] Motivated by this proximity, we for technical ease take the downward closure of the pruned collection as the collection $\mathcal{F}_v$. For the state of the art in pruning parent sets we refer to the results of de Campos and Ji (2011).

These issues become relevant when it comes to representing the input in a data structure that takes relatively little space but enables fast fetching of local scores. If only (a) and (b) hold, then a

2. We have examined the pruned parent set collections of 52 data sets made available by James Cussens at `www.cs.york.ac.uk/aig/sw/gobnilp/data/`. In 21 of the data sets, the maximum number of parents was set to 3, and in the rest it was set to 2. We found that the factor—defined as the size of the downward closure of the pruned collection divided by the size of the pruned collection—varied (over the data sets and nodes) from 1.0 to 4.2. The median value over the nodes varied (over the data sets) from 1.0 to 1.8. The maximum over the nodes varied from 1.0 to 4.2 and was below 2.7 for all but one data set; the value 4.2 was due to having 5 members in the pruned collection while 21 members in the closure. This suggests that larger factors occur only when the closure is small, in which case the size of the closure is not a bottleneck in the computations.

simple array representation would suffice, because the regularity enables efficient indexing. However, to allow for general downward-closed collections we will work with the following *augmented representation*. We assume that each $f_v$ is provided as an array of tuples $(Z, f_v(Z), U)$, where $Z \in \mathcal{F}_v$ and $U$ consists of the nodes $u$ outside $Z$ satisfying $Z \cup \{u\} \in \mathcal{F}_v$. We further assume that the array is ordered lexicographically by $Z$ (with respect to an arbitrary but fixed ordering of the nodes). This representation is motivated by the following observation concerning set interval queries, which we will need in Section 5.2.

The *interval* from set $X$ to set $Y$, denoted as $[X,Y]$, is the collection $\{Z : X \subseteq Z \subseteq Y\}$.

**Proposition 2 (interval queries)** *Given a downward closed collection $\mathcal{F}_v$ in the augmented representation and sets $X,Y \subseteq N \setminus \{v\}$, the scores $f_v(Z)$ for all $Z \in \mathcal{F}_v$ in the interval $[X,Y]$ can be listed in $O(n)$ time per score.*

**Proof** First, search for the tuple $(X, f_v(X), U)$. If no tuple is found, then stop and list no sets—this is correct, because a downward closed collection intersects the interval $[X,Y]$ only if $X$ belongs to the collection. Otherwise let $Z = X$, list $f_v(Z)$, and proceed recursively to the tuples of $Z' = Z \cup \{u\}$ for each $u \in U$ that belongs to $Y \setminus Z$ and succeeds the maximum node in $Z \setminus X$. Clearly, the algorithm visits all the desired tuples exactly once. For each visited tuple, locating it takes $O(\log |\mathcal{F}_v|))$ time using binary search, and traversing through the nodes in $U$ takes $O(n)$ time. Since $\log |\mathcal{F}_v| = O(n)$, the claimed time bound follows. ■

It is worth noting that the simple array representation, for cases (a) and (b), is more efficient. Compared to the augmented representation, it takes less space and enables faster interval queries, both by a factor linear in $n$. We leave the verification of this to the reader.

As suggested by the above discussion, we gauge the time and space requirements of an algorithm by the number of basic operations it executes and the maximum storage size it needs at any point of its execution, respectively. More specifically, by basic operations we refer not only to addition and comparison of real numbers but also to indexing arrays by nodes or node subsets. The storage size is assumed to be constant for real numbers and individual nodes.

### 2.2 Dynamic Programming

The OBN problem can be solved by dynamic programming across the node subsets (Ott and Miyano, 2003; Koivisto and Sood, 2004; Silander and Myllymäki, 2006; Singh and Moore, 2005). The idea of the algorithm can be described in terms of node orderings, as follows. Suppose $\hat{A}$ is an optimal DAG, that is, it maximizes the scoring function $f$. Because $\hat{A}$ is acyclic, there is at least one topological ordering of the nodes that is compatible with $\hat{A}$, that is, if $uv$ is an arc in $\hat{A}$, then $u$ precedes $v$ in the topological ordering. We will call any such ordering an *optimal linear order* on the nodes. Now, vice versa, if an optimal linear order is given, finding an optimal DAG is relatively simple: for each node $v$, independently, find a best-score parent set[3] among its predecessors in $O(|\mathcal{F}_v|)$ time. Now, when such an optimal linear order is not given, we basically need to consider all possible linear orders on the nodes. The key observation is that two linear orders imply the same best-score

3. Finding a best-score parent set is computationally hard for the usual local scoring functions, assuming the scoring function is not given by an explicit array of numbers but by an implicit expression, like the one in Example 1, that is efficient to evaluate for a given data set (Koivisto, 2006). In the worst case, there is little hope for doing better than examining all possible sets up to size logarithmic in the number of data points.

for node $v$ whenever the set of predecessors of $v$ are the same in the orders. Thus the algorithm only needs to tabulate the best cumulated scores for the node subsets. In the following paragraphs we present a dynamic programming algorithm that proceeds in two phases. We omit a rigorous proof of correctness. See Section 5 for a proof concerning a generalization of the algorithm.

In the *first phase*, the algorithm computes the best-score for each node $v$ and set of predecessors $Y \subseteq N \setminus \{v\}$, defined as

$$\hat{f}_v(Y) = \max_{X \subseteq Y} f_v(X) \,.$$

The direct computation of $\hat{f}_v(Y)$ for any fixed $v$ and $Y$ requires $2^{|Y|}$ basic operations. Thus, the total number of basic operations scales as $n \sum_{i=0}^{n-1} \binom{n-1}{i} 2^i = n3^{n-1}$. However, this can be significantly lowered by the following observation.

**Lemma 3 (Ott and Miyano 2003)** *If $v \in N$ and $Y \subseteq N \setminus \{v\}$, then*

$$\hat{f}_v(Y) = \max\Big\{ f_v(Y), \max_{u \in Y} \hat{f}_v(Y \setminus \{u\}) \Big\} \,.$$

This recurrence allows us to proceed levelwise, that is, in increasing cardinality of $Y$. If the values $\hat{f}_v(X)$ have already been computed for all $X \subset Y$ and stored in an array, computing $\hat{f}_v(Y)$ takes no more than $n$ comparisons. Recall we assume that indexing by subsets takes only constant time. Thus, the values $\hat{f}_v(Y)$ for all $v$ and $Y$ can be computed in $O(n^2 2^n)$ time. We discuss the space requirement later.

In the *second phase*, the algorithm goes through all node subsets $Y \subseteq N$, tabulating the maximum score over all DAGs on $Y$, denoted as $g(Y)$. In particular, $g(N)$ is the maximum score over all DAGs on $N$. One easily finds the recurrence

$$g(Y) = \max_{v \in Y} \Big\{ g(Y \setminus \{v\}) + \hat{f}_v(Y \setminus \{v\}) \Big\} \,, \tag{1}$$

with $g(\emptyset) = 0$. In other words, $g(Y \setminus \{v\}) + \hat{f}_v(Y \setminus \{v\})$ is the maximum score over all DAGs on $Y$ such that $v$ is the "last node", that is, the parents of $v$ are selected from $Y \setminus \{v\}$. Given the values $\hat{f}_v(Y)$ computed in the first phase, the values $g(Y)$ can be computed in $O(n2^n)$ basic operations. So the time requirement of the whole algorithm is $O(n^2 2^n)$.

Like the time requirement, also the space requirement is dominated by the first phase, the management of the values $\hat{f}_v(Y)$. If all intermediate results are kept in memory, the space requirement is $O(n2^n)$. However, this can be reduced to $O(\sqrt{n}2^n)$ by merging the two phases of the algorithm (Bellman, 1962; Ott and Miyano, 2003; Malone et al., 2011): Note that the computation of both $g(Y)$ and $\hat{f}_v(Y)$ requires information only about the sets of size $|Y| - 1$. Therefore, we can proceed levelwise and compute both phases for one level at a time. Thereby, at any level $\ell$ we need to keep in memory the values $g(Y)$ and $\hat{f}_v(Y)$ only for $O\big(\binom{n}{\ell} + \binom{n}{\ell-1}\big)$ sets $Y$ of size $\ell$ and $\ell - 1$. Hence the space requirement is $O(n\binom{n}{\lceil n/2 \rceil})$.

Once the optimal score has been computed, an optimal DAG can be constructed with negligible time and space overheads by back tracing, as follows. Starting with $Y = N$, first a node $v \in Y$ such that $g(Y) = g(Y \setminus \{v\}) + \hat{f}_v(Y \setminus \{v\})$ can be found in $O(n)$ time, since the required values of $g$ and $\hat{f}_v$ have already been computed and are available. An optimal parent set for $v$ can then be found by brute-force in $O(|\mathcal{F}_v|)$ time. If the values of $g$ and $\hat{f}_v$ are kept in memory for all node subsets,

then this step can be just repeated $n$ times. Otherwise, if the values of $g$ and $\hat{f}_v$ are kept in memory only for the node subsets at the last two levels, one can resolve the entire problem on the smaller node subset $Y \setminus \{v\}$ to recompute the needed values of $g$ and $\hat{f}_v$. While this is repeated $n$ times, the overall running time becomes not more than roughly two-fold, since the smaller instances are solved exponentially faster.

## 3. Two-Bucket Scheme

In this section we present a simple scheme for solving the OBN problem with less space, albeit more slowly. The idea is to guess the set of, say, $s$ first nodes of an optimal linear order and then solve separately the independent subproblems on those $s$ first nodes and on the last $n-s$ nodes. More formally and in terms of DAGs, the key observation is the following: Let $\hat{A}$ be an optimal DAG on the node set $N$. Fix an integer $s$ with $n/2 \leq s \leq n$. Since $\hat{A}$ is acyclic, there exists a partition of $N$ into two sets $N_0$ and $N_1$ of size $s$ and $n-s$, respectively, such that every arc between $N_0$ and $N_1$ in $\hat{A}$ is directed from $N_0$ to $N_1$. In other words, the parents of any node in $N_0$ are from $N_0$, while a node in $N_1$ may have parents from both $N_0$ and $N_1$. Thus, one can find $\hat{A}$—strictly speaking, the associated optimal score $f(\hat{A})$—by trying out all possible ordered partitions $(N_0, N_1)$ of $N$, with $|N_0| = s$ and $|N_1| = n-s$, and solving the recurrences

$$g_0(Y) = \max_{v \in Y} \left\{ g_0(Y \setminus \{v\}) + \hat{f}_v(Y \setminus \{v\}) \right\}, \tag{2}$$

for $\emptyset \subset Y \subseteq N_0$ with $g_0(\emptyset) = 0$, and

$$g_1(Y) = \max_{v \in Y} \left\{ g_1(Y \setminus \{v\}) + \hat{f}_v(N_0 \cup Y \setminus \{v\}) \right\}, \tag{3}$$

for $\emptyset \subset Y \subseteq N_1$ with $g_1(\emptyset) = 0$. The score of $\hat{A}$ is obtained as the maximum of $g_0(N_0) + g_1(N_1)$ over all the said partitions $(N_0, N_1)$.

We notice that the two subproblems are independent of each other given the partition $(N_0, N_1)$, and thus they can be solved separately. Applying the algorithm of the previous section, the computation of $g_0$ takes $O(2^s n^2)$ time and $O(2^s n)$ space.

Computing $g_1$ can be more expensive, since evaluating the term $\hat{f}_v(N_0 \cup Y \setminus \{v\})$ requires considering all possible subsets of $N_0$ as parents of $v$, in addition to a subset of $Y \setminus \{v\}$. To this end, at each $X_1 \subseteq N_1 \setminus \{v\}$ define

$$f'_v(X_1) = \max \{ f_v(X) : X \cap N_1 = X_1 , X \in \mathcal{F}_v \}$$

and

$$\hat{f}'_v(Y_1) = \max_{X_1 \subseteq Y_1} f'_v(X_1) .$$

Observe that computing $f'_v(X_1)$ for all $X_1$ takes $O((F + 2^{n-s})n)$ time in total, where $F$ is the size of $\mathcal{F}_v$. Now, because $\hat{f}_v(N_0 \cup Y_1) = \hat{f}'_v(Y_1)$, the algorithm of the previous section again applies to compute $g_1$, running in $O((F + 2^{n-s})n^2 + 2^{n-s}n^2)$ time and $O(2^{n-s}n)$ space in total. Since there are $\binom{n}{s}$ possible partitions $(N_0, N_1)$, we have the following:

**Proposition 4** *OBN can be solved in $O(\binom{n}{s}(2^s + F)n^2)$ time and $O((2^s + F)n)$ space for any $s = n, n-1, \ldots, n/2$, assuming each node has at most $F$ potential parent sets.*

When $F$ is $O(2^s)$ we can derive simpler asymptotic bounds. For instance, putting $s = 4/5n$ yields $O(2.872^n)$ time and $O(1.742^n)$ space. In general, this approach yields a smooth time-space tradeoff for space bounds between $O(2^{n/2}n)$ and $O(2^n n)$ (see Figure 4 in Section 7). However, within this space complexity range a more efficient scheme exists, as we will show in Section 6. Furthermore, with space less than $O(2^{n/2}n)$ the above scheme is not applicable.

## 4. Divide and Conquer Scheme

The partitioning idea from the previous section can be applied recursively, as described next. To solve the subproblems, namely computing $g_0(N_0)$ and $g_1(N_1)$ via the recurrences (2–3), with less space, we apply the partitioning technique again. The problem of computing $g_0(N_0)$ is of the same form as the original problem, and can thus be treated in a straightforward manner. The computation of $g_1(N_1)$ is only little more involved. Like before, for any fixed integer $s$ satisfying $0 \leq s \leq |N_1|$, there exists a partitioning of the node set $N_1$ into subsets $N_{10}$ and $N_{11}$ of size $s$ and $|N_1| - s$, respectively, such that every arc between $N_{10}$ and $N_{11}$ in an optimal DAG $\hat{A}$ is directed from $N_{10}$ to $N_{11}$. So, one can compute $g_1(N_1)$ by trying out all possible partitions $(N_{10}, N_{11})$ of $N_1$, with $|N_{10}| = s$ and $|N_{11}| = |N_1| - s$, and solving the recurrences

$$g_{10}(Y) = \max_{v \in Y} \Big\{ g_{10}(Y \setminus \{v\}) + \hat{f}_v((N_0 \cup Y \setminus \{v\}) \Big\},$$

for $\emptyset \subset Y \subseteq N_{10}$ with $g_{10}(\emptyset) = 0$, and

$$g_{11}(Y) = \max_{v \in Y} \Big\{ g_{11}(Y \setminus \{v\}) + \hat{f}_v(N_0 \cup N_{10} \cup Y \setminus \{v\}) \Big\},$$

for $\emptyset \subset Y \subseteq N_{11}$ with $g_{11}(\emptyset) = 0$. The score $g_1(N_1)$ is obtained as the maximum of $g_{10}(N_{10}) + g_{11}(N_{11})$ over all the said partitions $(N_{10}, N_{11})$. In general, one can apply partitioning recursively, say to depth $d$, and then solve the remaining subproblems by dynamic programming.

For an analysis of the time and space requirements, it is convenient to assume a balanced scheme: in every step of the recursion, the node set in question is partitioned into two sets of about equal sizes. For simplicity, assume $n$ is a power of 2. Then, at depth $d \geq 0$ of the recurrence, the node set in each subproblem in question is of size $s = n/2^d$. Hence, each subproblem can be solved in time and space within a polynomial factor to $2^s$, assuming the number of parents per node is polynomial in $n$; more precisely, if each node has at most $F$ potential parent sets, the routines described in the previous sections take $O((2^s n + F)n)$ time and $O((2^s + F)n)$ space. Because each subproblem of size $2s$ is divided into $2\binom{2s}{s} \leq 2^{2s}$ subproblems of size $s$, the total number of subproblems of size $s$ that need to be solved is at most $2^n 2^{n/2} 2^{n/4} \cdots 2^{2s} = 2^{2n-2s}$.

**Theorem 5** *OBN can be solved in $O(2^{2n-2s}(2^s + F)n^2)$ time and $O((2^s + F)n)$ space for any $s = n, n/2, n/4, \ldots$, assuming each node has at most $F$ potential parent sets.*

Choosing an $s \geq 0$ such that $2^s \leq F < 2^{s+1}$, gives a theoretically interesting implication:

**Corollary 6** *OBN can be solved in $O(4^n n^2/F)$ time and $O(nF)$ space, assuming each node has at most $F$ potential parent sets.*

In particular, when $F$ grows polynomially in $n$, we have a polynomial-space algorithm whose running time scales roughly as $4^n$. Analogous results are known for a number of related permutation problems, such as the traveling salesman problem, the minimum fill-in problem, the pathwidth problem, the cutwidth problem, the optimal linear arrangement problem, and the feedback arc set problem (Gurevich and Shelah, 1987; Björklund and Husfeldt, 2008; Bodlaender et al., 2012). The divide and conquer technique underlying all these results can be attributed to Savitch (1970).

## 5. The Partial Order Approach

This section generalizes the two-bucket scheme in a different direction than the divide and conquer scheme. Informally speaking, we replace a two-bucket partition by a partial order and, accordingly, the consideration of all fixed-size partitions by the consideration of sufficiently many partial orders so as to "cover" the linear orders.

We begin by introducing the needed concepts. Then the following three subsections generalize the two-phase dynamic programming algorithm. While Sections 5.2 and 5.3 present, respectively, the first and the second phase of the dynamic programming algorithm in a logical order, the treatment of the former is strongly motivated by the latter. The reader may prefer taking a look at the second phase first. We end this section with some remarks about choosing an efficient system of partial orders and about implications to parallel computation.

### 5.1 Partial Order Concepts

The following paragraphs introduce some concepts related to partial orders that will be needed in the remainder of this paper; for a more thorough treatment, an interested reader may refer, for instance, to the book by Davey and Priestley (2003).

Partial orders are binary relations that can be viewed as specializations of DAGs, for they inherit the acyclicity property of DAGs but also require additional properties of reflexivity and transitivity. But partial orders can also be viewed as extensions of DAGs, for the additional properties can be achieved by augmenting a DAG with appropriate arcs. Formally, a *partial order* $P$ on *ground set* $M$ is a subset of $M \times M$ such that for all $x, y, z \in M$ it holds that $xx \in P$ (reflexivity), $xy \in P$ and $yx \in P$ implies $y = x$ (antisymmetry), $xy \in P$ and $yz \in P$ implies $xz \in P$ (transitivity). A partial order $P$ is a *linear order* if, in addition, for all $x, y \in M$ it holds that $xy \in P$ or $yx \in P$ (totality). A linear order $Q$ is a *linear extension* of a partial order $P$ if $P \subseteq Q$. If $xy \in P$ and $x \neq y$, we say that $x$ is *smaller* than $y$ and that $y$ is *larger* than $x$; we denote by $P_y$ the set of all elements that are smaller than $y$. An element $x$ is *maximal* if no element is larger than $x$, and *minimal* if no element is smaller than $x$. The *trivial order* on $M$ is the "diagonal" partial order $\{xx : x \in M\}$. If the ground set consists of a single element, we call the partial order a *singleton order*. Because a partial order $P$ determines its ground set $M$, there will usually be no need to refer to the structure $(M, P)$ known as a *partially ordered set (poset)*. By writing $xy \in P$ instead of $xPy$ we emphasize the similar treatment of DAGs and partial orders. A notion of compatibility between DAGs and partial orders will be central in our developments:

**Definition 7 (compatibility)** A DAG $A$ and a partial order $P$ are said to be *compatible* with each other if there exists a partial order $Q$ that is a superset of both $A$ and $P$.

One of the key concepts we need is an *ideal* of a partial order. Informally, an ideal is a subset of the ground set that is closed under taking smaller elements. In the literature, ideals are also

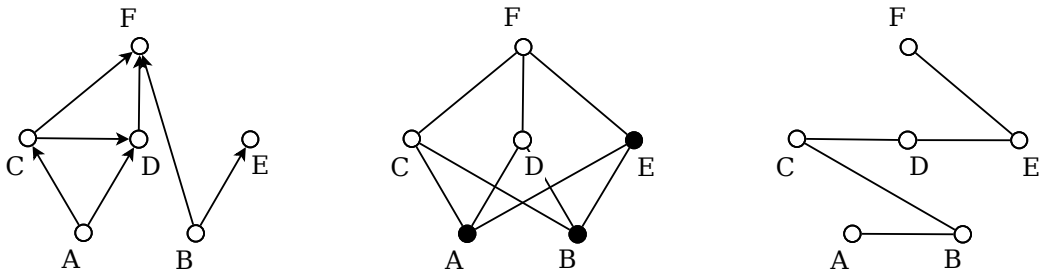



Figure 1: Three binary relations on the set $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D},\mathrm{E},\mathrm{F}\}$. (a) A DAG. (b) A Hasse diagram of a partial order compatible with the DAG. The partial order has 12 ideals, namely $\emptyset$, $\{\mathrm{A}\}$, $\{\mathrm{B}\}$, $\{\mathrm{A},\mathrm{B}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{C}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{D}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{E}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{E}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{D},\mathrm{E}\}$, $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D},\mathrm{E}\}$, and $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D},\mathrm{E},\mathrm{F}\}$. One of the ideals is marked by black dots. (c) A Hasse diagram of a linear order that is an extension (superset) of both the DAG and the partial order.

called *order ideals* or *downsets*. We will show that, when maximizing a decomposable scoring function over DAGs that are compatible with a given a partial order, the basic dynamic programming algorithm can be trimmed to run only across the ideals. We illustrate this soon in Example 2 and give a general treatment in Sections 5.2 and 5.3.

**Definition 8 (ideal)** Let $P$ be a partial order on $M$. A subset $I$ of $M$ is called an *ideal* of $P$ if $y \in I$ and $xy \in P$ imply that $x \in I$. We denote the set of all ideals of $P$ by $\mathcal{I}(P)$.

Figure 1 illustrates some of the above defined concepts. Note that we visualize a partially ordered set $(M, P)$ by its *transitive reduction*, that is, the graph obtained by removing from $P$ all pairs $xy$ for which $x = y$ or there exist a $z$ that is larger than $x$ and smaller than $y$. We draw a transitive reduction on the plane using a Hasse diagram: a pair $xy$ in the reduction corresponds to a line segment such that $x$ is positioned below or to the left of $y$.

**Example 2 (dynamic programming across ideals)** Consider maximizing a decomposable scoring function over DAGs that are compatible with the partial order given in Figure 1(b). Because the score is decomposable, an optimal DAG $\hat{A}$ must have a sink node $v \in N = \{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D},\mathrm{E},\mathrm{F}\}$ such that the parent set $\hat{A}_v$ of $v$ maximizes the local score over all subsets of $N \setminus \{v\}$, and the remainder of the DAG maximizes the score over DAGs on $N \setminus \{v\}$—this is the essence of the basic dynamic programming recurrence (1). For concreteness, the reader may think of the DAG of Figure 1(a) as the optimal DAG. The key observation we can make here is that, due to the precedence constraints given by the partial order, not all nodes $v \in N$ need to be considered as the possible sink node. Indeed, since F is the only maximal element in the partial order, it has to be a sink of any DAG compatible with the partial order. Thus, in the dynamic programming recurrence it suffices to consider only one out of the six cases, namely recursing on the subproblem of maximizing the score over DAGs on $N \setminus \{\mathrm{F}\} = \{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D},\mathrm{E}\}$. It is not a coincidence that this is the only ideal of

the partial order of size 5. Continuing the same reasoning to smaller subproblems shows that it is sufficient to tabulate the optimal scores (DAGs) for the ideals of the partial order.

We will work with collections of partial orders that share the same ground set. We call such a collection a *partial order system* on the ground set, or *POS* for short. Our interest is particularly in partial order systems that exhibit sufficient diversity so as to "cover" all linear orders on the ground set. This idea is formalized in the following:

**Definition 9 (cover)** A POS $\mathcal{P}$ on $M$ is said to be a *cover* on $M$ if any linear order on $M$ is an extension of at least one partial order in $\mathcal{P}$.

An extreme example of a cover on a ground set $M$ is the collection $\{T\}$ formed by the trivial order $T$ on $M$. At the other extreme we have the cover that consists of all linear orders on $M$. For yet another example, consider the three partial orders defined by $P = \{\mathrm{AA}, \mathrm{BB}, \mathrm{CC}, \mathrm{AB}, \mathrm{AC}\}$, $Q = \{\mathrm{AA}, \mathrm{BB}, \mathrm{CC}, \mathrm{BA}, \mathrm{BC}\}$, and $R = \{\mathrm{AA}, \mathrm{BB}, \mathrm{CC}, \mathrm{CA}, \mathrm{CB}\}$. We see that the system $\{P, Q, R\}$ is a cover on the ground set $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\}$.

### 5.2 Dynamic Programming: First Phase

We aim at an algorithm that maximizes $f(A)$ over all DAGs $A$ on the node set $N$ subject to the constraint that $A$ is compatible with a given partial order $P$. In this subsection, we modify the first phase of the basic dynamic programming algorithm, described in Section 2.2, accordingly. In the next subsection, we will modify the second phase.

Recall that in the first phase, the task is to compute the values $\hat{f}_v(Y)$ for all nodes $v$ and node subsets $Y \subseteq N \setminus \{v\}$. However, it turns out that, in the second phase, we need these values only for the ideals $Y$ of the given partial order $P$. This gives us an opportunity to save space, provided that we have an appropriate "sparse" variant of the recurrence of Lemma 3 at hand. Next we present such a variant.

Our key insight is the following simple observation concerning collections of subsets. We leave the proof to the reader.

**Lemma 10** *Let $X$ and $Y$ be sets with $X \subseteq Y$. Let*

$$\mathcal{A} = [X, Y] \quad \textit{and} \quad \mathcal{B} = \{Z \subseteq Y : x \notin Z \textit{ for some } x \in X\}.$$

*Then (i) $2^Y = \mathcal{A} \cup \mathcal{B}$ and (ii) $\mathcal{B} = \bigcup_{x \in X} 2^{Y \setminus \{x\}}$.*

In terms of the set functions $f_v$ and $\hat{f}_v$, for an arbitrary $v$, Lemma 10 amounts to the following generalization of Lemma 3 (one obtains Lemma 3 at $X = Y$):

**Lemma 11** *Let $X$ and $Y$ be subsets of $N \setminus \{v\}$ with $X \subseteq Y$. Then*

$$\hat{f}_v(Y) = \max\Big\{ \max_{X \subseteq Z \subseteq Y} f_v(Z), \max_{u \in X} \hat{f}_v(Y \setminus \{u\}) \Big\}.$$

**Proof** By Lemma 10(ii) and the definition of $\hat{f}_v$, the maximum of $f_v(Z)$ over $Z \in \bigcup_{u \in X} 2^{Y \setminus \{u\}}$ equals $\max_{u \in X} \hat{f}_v(Y \setminus \{u\})$. By Lemma 10(i), the larger of $\max_{X \subseteq Z \subseteq Y} f_v(Z)$ and $\max_{u \in X} \hat{f}_v(Y \setminus \{u\})$ equals $\max_{V \subseteq Y} f_v(V)$, which by definition is $\hat{f}_v(Y)$. ■

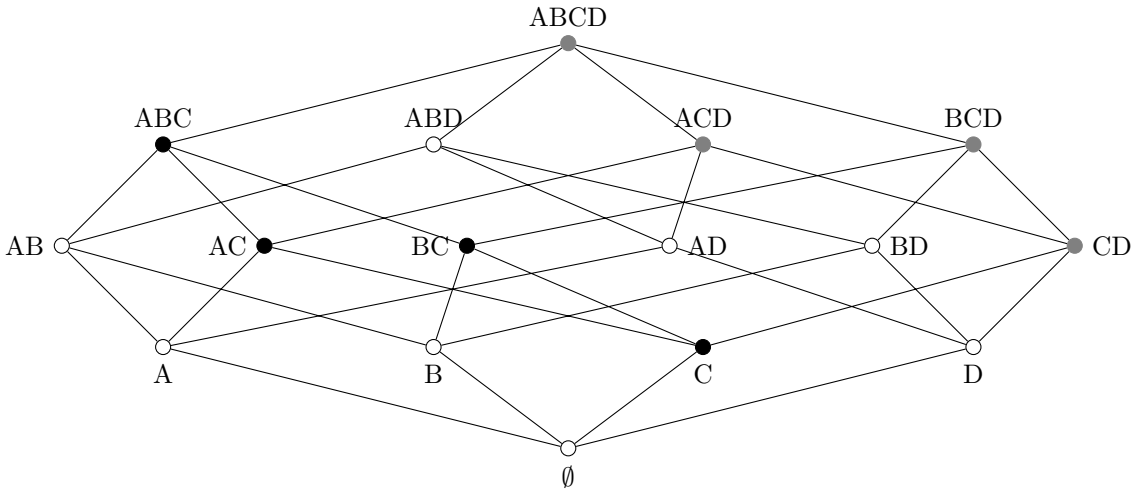



Figure 2: Tails of the ideals $\{\mathrm{A},\mathrm{B},\mathrm{C}\}$ and $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}\}$ of the partial order of Figure 1(b), marked in black and gray, respectively, on the subset lattice of $\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}\}$. An interval from one set to another consists of the sets that are along a shortest path between the two sets in a Hasse diagram of the lattice. For example, the interval $[\{\mathrm{C}\},\{\mathrm{A},\mathrm{B},\mathrm{C}\}]$ consists of the sets $\{\mathrm{C}\}$, $\{\mathrm{A},\mathrm{C}\}$, $\{\mathrm{B},\mathrm{C}\}$, and $\{\mathrm{A},\mathrm{B},\mathrm{C}\}$. In the figure, each subset is referred to by a sequence of its elements.

Lemma 11 leaves us the freedom to choose a suitable node subset $X$ for each set of interest $Y$. For this choice, we make use of the fact that in the second phase of dynamic programming, as it will turn out in the next subsection, we need the values $\hat{f}_v(Y)$ only for sets $Y$ that are ideals of $P$. Thus, our goal is to choose $X$ such that $Y \setminus \{u\} \in \mathcal{I}(P)$ for all $u \in X$. To this end, we let $X$ consist of all such nodes in $Y$ that have no larger node in $Y$ (w.r.t. $P$). Accordingly, for $Y \in \mathcal{I}(P)$ define

$$\check{Y} = \{u \in Y : uv \notin P \text{ for all } v \in Y \setminus \{u\}\}.$$

Furthermore, define the *tail* of $Y$ as the interval

$$\mathcal{T}_Y = [\check{Y}, Y];$$

see Figure 2 for an illustration. By letting $X = \check{Y}$ and noting that $f_v(Z) = -\infty$ for $Z \notin \mathcal{F}_v$, we may rephrase the equation in Lemma 11 as

$$\hat{f}_v(Y) = \max\Big\{ \max_{Z \in \mathcal{T}_Y \cap \mathcal{F}_v} f_v(Z),\ \max_{u \in \check{Y}} \hat{f}_v(Y \setminus \{u\}) \Big\}. \tag{4}$$

The next two lemmas show that $\check{Y}$ indeed has the desired property (in a maximal sense) and that the tails of different ideals $Y$ are pairwise disjoint and thus optimally cover the subsets of $N$.

**Lemma 12** *Let $Y \in \mathcal{I}(P)$ and $u \in Y$. Then $Y \setminus \{u\} \in \mathcal{I}(P)$ if and only if $u \in \check{Y}$.*

**Proof** "If": Let $u \in \check{Y}$. Let $st \in P$. By the definition of $\mathcal{I}(P)$ we need to show that $t \in Y \setminus \{u\}$ implies $s \in Y \setminus \{u\}$. So, suppose $t \in Y \setminus \{u\}$, hence $t \in Y$. Now, since $Y \in \mathcal{I}(P)$, we must have $s \in Y$. It remains to show that $s \neq u$. But this holds because $ut \notin P$ by the definition of $\check{Y}$.

"Only if": Let $u \notin \check{Y}$. Then we have $uv \in P$ for some $v \in Y \setminus \{u\}$. But $u \notin Y \setminus \{u\}$ and $v \in Y \setminus \{u\}$, implying $Y \setminus \{u\} \notin \mathcal{I}(P)$ by the definition of $\mathcal{I}(P)$. ■

**Lemma 13** *Let $Y$ and $Y'$ be distinct ideals of $P$. Then the tails of $Y$ and $Y'$ are disjoint.*

**Proof** The lemma states that there does not exist any nonempty $Z$ such that $Z \in \mathcal{T}_Y \cap \mathcal{T}_{Y'}$. Suppose the contrary that $Z \in \mathcal{T}_Y \cap \mathcal{T}_{Y'}$. By symmetry we may assume that $Y \setminus Y'$ contains an element $w$. Thus $w \notin Z$, because $Z \subseteq Y'$. Because $\check{Y} \subseteq Z$, we have $w \notin \check{Y}$. By the definition of $\check{Y}$ we conclude that for every $u \in Y \setminus \check{Y}$ there exists $v \in \check{Y}$ such that $uv \in P$. Therefore, in particular there exists $v \in \check{Y}$ such that $wv \in P$. Since $w \notin Y'$ and $Y'$ is an ideal of $P$ it follows by definition of an ideal that $v \notin Y'$. On the other hand, $v \in \check{Y}$ and $\check{Y} \subseteq Z \subseteq Y'$ implies that $v \in Y'$: contradiction. ■

We we will use these lemmas in Section 5.4 to perform the recurrence of Lemma 11 over the ideals of the partial order and their tails.

### 5.3 Dynamic Programming: Second Phase

Recall that the second phase of the basic dynamic programming algorithm is captured by the recurrence (1), which concerns all subsets of $N$ that can begin some linear order on $N$, that is, all the $2^n$ subsets. Now that we restrict our attention to DAGs that are compatible with the given partial order $P$, we may restrict the recurrence to only those sets that can begin some linear extension of $P$, that is, to the ideals of $P$. Formally, define the function $g^P$ by $g^P(\emptyset) = 0$ and for nonempty $Y \in \mathcal{I}(P)$ recursively:

$$g^P(Y) = \max_{\substack{v \in Y \\ Y \setminus \{v\} \in \mathcal{I}(P)}} \left\{ g^P(Y \setminus \{v\}) + \hat{f}_v(Y \setminus \{v\}) \right\}. \tag{5}$$

We can show that $g^P(N)$ equals the maximum score over the DAGs compatible with $P$:

**Lemma 14** *Let $P$ be a partial order on $N$. Then*

$$g^P(N) = \max\{f(A) : A \text{ is compatible with } P\}.$$

*Furthermore, if $\mathcal{P}$ is a cover on $N$, then*

$$\max_{P \in \mathcal{P}} g^P(N) = \max_A f(A),$$

*where $A$ runs through all DAGs on $N$.*

**Proof** Let $P$ be a partial order on $N$. For any subset $Y \subseteq N$ denote by $P[Y]$ the induced partial order $\{xy \in P : x, y \in Y\}$. We show by induction on the size of $Y$ that $g^P(Y)$ equals the maximum score $f(A')$ over the DAGs $A' \subseteq Y \times Y$ compatible with $P[Y]$, assuming $Y$ is an ideal of $P$. Here the score $f(A')$ is naturally defined as $\sum_{v \in Y} f_v(A'_v)$.

For the base case, consider an arbitrary singleton $\{v\} \in \mathcal{I}(P)$. Clearly, there is exactly one DAG $A'$ on $\{v\}$ and it is compatible with $P[\{v\}] = \{vv\}$; the score of the DAG is $f(A') = f_v(\emptyset) = \hat{f}_v(\emptyset)$. This is precisely what the recurrence (5) gives, as $g^P(\emptyset) = 0$.

Suppose then that the recurrence (5) holds for all proper subsets of a subset $Y \subseteq N$. Without any loss of generality, we assume $Y = N$ for notational convenience. Now, write

$$\begin{aligned}
\max_{A \text{ is compatible with } P} f(A) &= \max_{L \supseteq P} \max_{A \subseteq L} \sum_{v \in N} f_v(A_v) \\
&= \max_{L \supseteq P} \sum_{v \in N} \max_{A_v \subseteq L_v} f_v(A_v) \\
&= \max_{L \supseteq P} \sum_{v \in N} \hat{f}_v(L_v) \\
&= \max_{v \in N} \Big\{ \hat{f}_v(N \setminus \{v\}) + \max_{L' \supseteq P[N \setminus \{v\}]} \sum_{u \in N \setminus \{v\}} \hat{f}_u(L'_u) \Big\} \\
&= \max_{v \in N} \Big\{ \hat{f}_v(N \setminus \{v\}) + g^{P[N \setminus \{v\}]}(N \setminus \{v\}) \Big\} .
\end{aligned}$$

Here $L$ and $L'$ run trough the respective linear extensions and $A$ through the DAGs satisfying the mentioned condition. The first identity holds by the definition of compatibility and the decomposability of the scoring function $f$; the second one by the distributive law; the third one by the definition of $\hat{f}_v$; the fourth one because addition distributes over maximization and the fact that some node $v$ is the last one in the linear order $L$ and that the induced linear order $L'$ on the remaining nodes is an extension of the induced partial order $P[N \setminus \{v\}]$; the fifth one by the induction assumption (and the third identity). Finally, it suffices to notice that $g^P(Y) = g^{P[Y]}(Y)$ for $Y \in \mathcal{I}(P)$, since clearly a subset $X$ of $Y$ is an ideal of $P$ if and only if $X$ is an ideal of $P[Y]$.

For the second statement, it suffices to observe that

$$\max_{P \in \mathcal{P}} \max_{L \supseteq P} \sum_{v \in N} \hat{f}_v(L_v) = \max_{L} \sum_{v \in N} \hat{f}_v(L_v) = \max_{A} f(A) ,$$

since $\mathcal{P}$ is a cover on $N$. ■

### 5.4 Dynamic Programming: First and Second Phase Merged

We now merge the ingredients given in the previous two subsections into an algorithm for evaluating $g^P$ using the recurrence (5) and Lemma 11, for a fixed $P \in \mathcal{P}$. In Algorithm 1 below, $g^P[Y]$ and $\hat{f}_v[Y]$ denote program variables that correspond to the respective target values $g^P(Y)$ and $\hat{f}_v(Y)$ to be computed. Also, recall that $\mathcal{F}_v$ denotes the collection of potential parent sets for node $v$.

**Algorithm 1:**

1. Let $g^P[\emptyset] \leftarrow 0$.
2. For each $v \in N$, let $\hat{f}_v[\emptyset] \leftarrow f_v(\emptyset)$.
3. For each nonempty $Y \in \mathcal{I}(P)$, in increasing order of cardinality:
   (a) let
   $$g^P[Y] \leftarrow \max_{v \in \check{Y}} \left\{ g^P[Y \setminus \{v\}] + \hat{f}_v[Y \setminus \{v\}] \right\};$$
   (b) for each $v \in Y$, let $\hat{f}_v[Y]$ be the larger of
   $$\max_{Z \in \mathcal{T}_Y \cap \mathcal{F}_v} f_v(Z) \text{ and } \max_{u \in \check{Y}} \hat{f}_v[Y \setminus \{u\}].$$

**Lemma 15** *Algorithm 1 correctly computes $g^P$, that is, $g^P[Y] = g^P(Y)$ for all $Y \in \mathcal{I}(P)$.*

**Proof** Observe first that the algorithm correctly computes $\hat{f}_v$ for each $v \in N$, that is, after the execution of the algorithm we have $\hat{f}_v[Y] = \hat{f}_v(Y)$ for all $v \in Y \in \mathcal{I}(P)$. To this end, it suffices to note that step 3(b) implements the recurrence of Lemma 11 as rephrased in (4).

Then notice that step 3(a) implements the recurrence (5). Indeed, by Lemma 12, the condition "$v \in Y$ and $Y \setminus \{v\} \in \mathcal{I}(P)$" of (5) is equivalent to "$v \in \check{Y}$" of step 3(a), given that $Y \in \mathcal{I}(P)$ (guaranteed in step 3). This completes the proof. ■

To solve the OBN problem it suffices to run Algorithm 1 for every partial order in a system that is a cover on $N$. The appropriate partial order system of course varies with the problem instance, particularly with the number of nodes $n$. In the following statement of the time and space complexity of OBN, we do not fix any particular way to choose and construct the needed partial order system, but we simply refer to any appropriate system. Thus, the complexity results are nonuniform with this respect.

**Theorem 16 (main)** *OBN can be solved in $O\big(\big[\sum_{P \in \mathcal{P}} (|\mathcal{I}(P)| + F)\big] n^2\big)$ time and $O\big(\big[\max_{P \in \mathcal{P}} |\mathcal{I}(P)| + Fn\big] n\big)$ space, assuming $\mathcal{P}$ is a cover of $N$ and each $\mathcal{F}_v$ is downward closed and of size at most $F$.*

**Proof** By Lemmas 14 and 15, it suffices to run Algorithm 1 for each $P \in \mathcal{P}$.

The time requirement of Algorithm 1 is dominated by steps 3(a) and 3(b). Given $Y$, the set $\check{Y}$ can be constructed in time $O(n^2)$ by removing from $Y$ each element that is not maximal in $Y$. Thus, the contribution of step 3(a) in the total time requirement is $O(|\mathcal{I}(P)|n^2)$.

We then analyze the time requirement of step 3(b), for fixed $v$. By Proposition 2, the maximization of the local scores over $\mathcal{T}_Y \cap \mathcal{F}_v$ can be done in $O(|\mathcal{T}_Y \cap \mathcal{F}_v|n)$ time. By Lemma 13 the collections $\mathcal{T}_Y \cap \mathcal{F}_v$ are disjoint for different $Y \in \mathcal{I}(P)$. Thus the total contribution to the time requirement is proportional to $|\mathcal{F}_v| \leq F$, for each $v$. Because step 3(b) is executed $|\mathcal{I}(P)|$ times, the total time requirement of step 3(b) is $O(|\mathcal{I}(P)|n^2 + Fn^2)$. Combining the time bounds of steps 3(a) and 3(b) and summing over all members of $\mathcal{P}$ yields the claimed bound $O\big(\big[\sum_{P \in \mathcal{P}} |\mathcal{I}(P)| + F\big] n^2\big)$.

The space requirement for a fixed partial order $P$ is $O(|\mathcal{I}(P)|n)$, since by Lemma 12 the values $g^P[Y]$ and $\hat{f}_v[Y]$ are only needed for $Y \in \mathcal{I}(P)$. In addition, storing the local scores requires $O(Fn)$ space for each node $v$. Therefore, the total space requirement is $O(\lceil \max_{P \in \mathcal{P}} |\mathcal{I}(P)| + Fn \rceil n)$. ■

**Remark 17** Like in the basic DP algorithm, step 3 of Algorithm 1 can be implemented so that, compared to the bound in Theorem 16, a space saving proportional to $\sqrt{n}$ is obtained.

### 5.5 Notes

We end this section with a couple of remarks on the general partial order approach.

The results in this section apply to any POS that is a cover on the node set. However, they do not tell us how to choose a POS that yields, in some sense, the most efficient tradeoff between space and time. Ideally, we would like to have a scheme that given a space bound as a function of the number of nodes $n$, say $s_n$, gives us a POS $\mathcal{P}_n$ such that the space requirement implied by $\mathcal{P}_n$ is $O(s_n)$ while the time requirement bound is minimized. Unfortunately, designing such an optimal scheme seems difficult due to the combinatorial challenge of simultaneously controlling the required covering property of the partial order system and the number of ideals of the members of the system. Note that both counting the linear extensions and the ideals of a given partial order are #P-hard computational problems (Brightwell and Winkler, 1991; Provan and Ball, 1983), suggesting that their mathematical analysis is also not easy. In the next section we give a partial and practical solution to this issue by studying a subclass of series-parallel partial orders, which enables a derivation of concrete, quantitative space-time tradeoff results.

As another remark, we note that the algorithms can be easily parallelized onto $|\mathcal{P}|$ processors, each with its own memory, with negligible communication costs. This is in sharp contrast with the basic dynamic programming algorithm that does not enable such large-scale parallelization. We discuss the recent work by Tamada et al. (2011) in this light in Section 7. The ease of the parallelization stems from the fact that dynamic programming over the ideals can be done independently for each partial order $P$ in $\mathcal{P}$. More precisely, each processor gets a dedicated partial order $P$ (and the local scores) as input and outputs the score $g^P(N)$ and a respective DAG. It then remains to communicate these outputs to a central unit that chooses an optimal one among them.

## 6. Bucket Order Schemes

To apply the partial order approach in practice, it is essential to find partial order systems that provide a good tradeoff between time and space requirements. This section concentrates on a class of partial orders we call *parallel bucket orders*, which turn out to be relatively easy to analyze and seem to yield good tradeoff in practice. They also subsume, for instance, the two-bucket construction of Section 3.

### 6.1 Bucket Orders and Reorderings

Let $P$ and $Q$ be partial orders on disjoint ground sets $M$ and $N$, respectively. We say that a partial order $R$ is a *series composition* of $P$ and $Q$ if $R = P \cup Q \cup \{xy : x \in M, y \in N\}$, and a *parallel composition* of $P$ and $Q$ if simply $R = P \cup Q$. Note that both compositions always yield a partial order, since the ground sets are disjoint. As both composition operations are associative, they read-

ily extend to any finite number of partial orders. Because the series composition operation is not commutative, it is, of course, applied to a sequence of partial orders. A *series-parallel* partial order is defined recursively: a partial order is a series-parallel partial order if it is either a singleton order or a parallel or series composition of two or more series-parallel partial orders. For example, the partial order $\{\mathrm{AA}, \mathrm{BB}, \mathrm{CC}, \mathrm{AB}, \mathrm{AC}\}$ is a series composition of $\{\mathrm{AA}\}$ and $\{\mathrm{BB}, \mathrm{CC}\}$, of which the former is a singleton order and the latter is a parallel composition of two singleton orders. Note that any trivial order is a parallel composition of singleton orders.

We study two special classes of series-parallel partial orders, namely bucket orders and parallel bucket orders. A *bucket order* is a series composition of the trivial orders on some ground sets $B_1, B_2, \ldots, B_\ell$, called *buckets*. The bucket order is said to be of *length* $\ell$ and *type* $|B_1| * |B_2| * \cdots * |B_\ell|$. We may denote the bucket order by $B_1 B_2 \cdots B_\ell$. For instance, the series-parallel partial order in the previous paragraph is a bucket order $\{\mathrm{A}\}\{\mathrm{B}, \mathrm{C}\}$, thus of length two and type $1 * 2$. Likewise, the partial order in Figure 1(b) is a bucket order of length three and type $2 * 3 * 1$, the trivial order on some ground set $M$ is of length one and type $|M|$, and a linear order on $M$ is of length $|M|$ and type $1 * 1 * \cdots * 1$.

By taking a parallel composition of some number of bucket orders we obtain a *parallel bucket order*. Note that the same parallel bucket order may be obtained by different collections of bucket orders. It is, however, easy to observe that for each parallel bucket order $P$ there is a unique collection of bucket orders $P_1, P_2, \ldots, P_p$, called the bucket orders of $P$, such that their parallel composition is $P$ and they are the connected components of $P$.

The following lemma states a well-known result concerning the number of ideals of a series-parallel partial order (see, for example, Steiner's article, 1990, and references therein).

**Lemma 18** *Let $P_1$ and $P_2$ be partial orders on disjoint ground sets. Then (i) the series composition of $P_1$ and $P_2$ has $|\mathcal{I}(P_1)| + |\mathcal{I}(P_2)| - 1$ ideals and (ii) the parallel composition of $P_1$ and $P_2$ has $|\mathcal{I}(P_1)||\mathcal{I}(P_2)|$ ideals.*

The next two lemmas state the number of ideals of a parallel bucket order.

**Lemma 19** *Let $B$ be a bucket order $B_1 B_2 \ldots B_\ell$. Then the number of ideals of $B$ is given by $|\mathcal{I}(B)| = 1 - \ell + 2^{|B_1|} + 2^{|B_2|} + \cdots + 2^{|B_\ell|}$.*

**Proof** Any singleton order has two ideals, namely the empty set and the ground set. Thus, by Lemma 18(ii), the trivial order on the bucket $B_i$ has $2^{|B_i|}$ ideals, for each $i$. Hence, by Lemma 18(i), $B$ has $1 - \ell + 2^{|B_1|} + 2^{|B_2|} + \cdots + 2^{|B_\ell|}$ ideals. ■

We note that the order of buckets does not affect the number of ideals.

**Lemma 20** *Let $P$ be the parallel composition of bucket orders $P_1, P_2, \ldots, P_p$. Then the number of ideals of $P$ is given by $|\mathcal{I}(P)| = |\mathcal{I}(P_1)||\mathcal{I}(P_2)| \cdots |\mathcal{I}(P_p)|$.*

**Proof** Follows immediately from Lemma 18(ii). ■

Let us return to the issue of choosing a POS that is a cover of the node sets and yields a good space-time tradeoff. To this end, we will consider systems of parallel bucket orders of a certain kind, namely systems obtained via "reordering" a fixed parallel bucket order:

**Definition 21 (reordering)** We say two bucket orders are *reorderings* of each other if they have the same ground set and they are of the same type. Furthermore, we say two parallel bucket orders are *reorderings* of each other if their bucket orders can be labelled as $P_1, P_2, \ldots, P_p$ and $Q_1, Q_2, \ldots, Q_p$, respectively, such that $P_i$ is a reordering of $Q_i$ for each $i$. We denote the collection of reorderings of a parallel bucket order $P$ by $\mathcal{R}(P)$.

We note that two bucket orders are reorderings of each other if and only if they are automorphic to each other. However, this equivalence does not hold for parallel bucket orders in general, for reordering only allows shuffling within each bucket order, but not between different bucket orders.

It can be shown that $\mathcal{R}(P)$ is a cover on the ground set of $P$. The proof below slightly simplifies the one we have given earlier (Koivisto and Parviainen, 2010, Theorem 3.1).

**Theorem 22** *Let $P$ be a parallel composition of bucket orders. Then $\mathcal{R}(P)$ is a cover on the ground set of $P$.*

**Proof** Let $L$ be a linear order on $M$. It suffices to construct a parallel bucket order $Q \in \mathcal{R}(P)$ such that $L$ is an extension of $Q$. To this end, let $P_1, P_2, \ldots, P_p$ be the bucket orders of $P$, with respective ground sets $M_1, M_2, \ldots, M_p$. For each $i = 1, 2, \ldots, p$, construct a bucket order $Q_i$ on $M_i$ as follows: Let $m_1 * m_2 \cdots * m_\ell$ be the type of $P_i$. Let $L'$ be the induced order $L[M_i]$. Observe that $L'$ is a linear order. Now, put $Q_i = C_1 C_2 \cdots C_\ell$ where $C_1$ consists of the first $m_1$ elements in the order $L'$, $C_2$ consists of the next $m_2$ elements in the order, and so forth. Observe that $L'$ is an extension of $Q_i$ and that $Q_i$ is a reordering of $P_i$. Finally, let $Q$ be the parallel composition of $Q_1, Q_2, \ldots, Q_p$. Clearly, $Q \in \mathcal{R}(P)$. To complete the proof, note that $L$ is an extension of $Q$, since $xy \in Q$ implies $xy \in Q_i$ for some $i$, whence $xy \in L[M_i] \subseteq L$. ■

When $P$ is a parallel composition of $p$ bucket orders, each of type $b_1 * b_2 * \cdots * b_\ell$, we find it convenient to denote the POS $\mathcal{R}(P)$ by $(b_1 * b_2 * \cdots * b_\ell)^p$. This notation is explicit about the combinatorial structure of the POS, while it ignores the arbitrariness of the labeling of the ground set. When the size of the ground set, $n$, is clear from the context, we may extend the notation $(b_1 * b_2 * \cdots * b_\ell)^p$ to refer to a system $\mathcal{R}(P)$, where $P$ is a parallel composition of $p$ bucket orders of type $b_1 * b_2 * \cdots * b_\ell$ and one trivial order on the remaining $n - p(b_1 + b_2 + \cdots + b_\ell)$ elements. It is instructive to notice that a POS $(b_1 * b_2 * \cdots * b_\ell)^p$ consists of $\binom{b_1+b_2+\cdots+b_\ell}{b_1 \quad b_2 \quad \cdots \quad b_\ell}^p$ different partial orders, since any bucket order of type $b_1 * b_2 * \cdots * b_\ell$ has exactly $\binom{b_1+b_2+\cdots+b_\ell}{b_1 \quad b_2 \quad \cdots \quad b_\ell} = \frac{(b_1+b_2+\cdots+b_\ell)!}{b_1! b_2! \cdots b_\ell!}$ different reorderings.

For an illustration of these concepts, consider a node set $N = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}, \mathrm{F}, \mathrm{G}, \mathrm{H}\}$ partitioned into $N_1 = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}, \mathrm{F}\}$ and $N_2 = \{\mathrm{G}, \mathrm{H}\}$. Let the bucket order on $N_1$ be the one shown in Figure 1(b), denoted as $P_1$, and let the bucket order on $N_2$ be the trivial order $\{\mathrm{GG}, \mathrm{HH}\}$, denoted as $P_2$. Now the reorderings of the parallel composition of $P_1$ and $P_2$ form a POS $(2 * 3 * 1)^1$. By Lemma 19, $P_1$ has $1 - 3 + 2^2 + 2^3 + 2^1 = 12$ ideals and $P_2$ has $1 - 1 + 2^2 = 4$ ideals. By Lemma 20, the total number of ideals of the parallel composition of $P_1$ and $P_2$ is $12 \times 4 = 48$. The number of the partial orders in $(2 * 3 * 1)^1$ is $\binom{6}{2 \quad 3 \quad 1}^1 = 60$.

### 6.2 Bucket Order Schemes

A partial order system $(b_1 * b_2 * \cdots * b_\ell)^p$ is associated with some natural *parameters*, such as the number of parallel bucket orders $p$. When all or some of the parameters are treated as variables

that can take different values, we refer to the implied family of partial order systems as a *bucket order scheme*. Furthermore, we denote the scheme simply by the expression $(b_1 * b_2 * \cdots * b_\ell)^p$, where the parameters and their ranges are assumed to understood from the context. For instance, we may talk about the scheme $(5*5)^p$, understanding that $p$ takes values over its natural range, that is, $p = 1, 2, \ldots, \lfloor n/(5+5) \rfloor$. Another example of a bucket order scheme is the *two-bucket scheme* of Section 3, which corresponds to partial order systems $(s*(n-s))^1$, with $s$ treated as the parameter. Other examples are the *pairwise scheme*, corresponding to systems $(1*1)^p$, with $p$ as the parameter, and the *generalized two-bucket scheme* defined by the systems $(\lceil m/2 \rceil * \lfloor m/2 \rfloor)^{\lfloor n/m \rfloor}$, with $m$ as the parameter.

Via Theorem 16, any fixed bucket order scheme implies parameterized time and space complexity bounds for the OBN problem. Moreover, one scheme may dominate another scheme in the sense of yielding an equal or smaller time bound at any space bound. While it is currently an open problem to characterize bucket order schemes that dominate all other bucket order schemes, our analysis of the so-called space-time product (Koivisto and Parviainen, 2010) suggests that the most efficient trade-off is achieved with the bucket order scheme $(\lceil m/2 \rceil, \lfloor m/2 \rfloor)^p$. The scheme guarantees that the product of the time and space requirements scales roughly as $C^n$, where $C$ equals $4$ or is slightly below $4$ depending on the space bound. If some other scheme resulted in a significantly faster algorithm at any space bound, then that scheme would make a new record also in terms of the space-time product, albeit possibly at a single point. Our prior study (Koivisto and Parviainen, 2010) shows that no such scheme exists among bucket order schemes. When measured by the space-time product, the tradeoff of the scheme $(\lceil m/2 \rceil, \lfloor m/2 \rfloor)^p$ slowly improves when $m$ increases, until $m = 26$, after which the tradeoff starts slowly getting worse. We next examine this scheme in more detail focusing on the range $2 \leq m \leq 26$. See Figure 4 (in Section 7) for the tradeoff curve.

### 6.3 Practical Bucket Order Schemes

A partial order system $(\lceil m/2 \rceil * \lfloor m/2 \rfloor)^p$ consists of $\binom{m}{\lfloor m/2 \rfloor}^p$ partial orders, each having $2^{n-mp}(2^{\lfloor m/2 \rfloor} + 2^{\lceil m/2 \rceil} - 1)^p$ ideals. Plugging these numbers into Theorem 16 gives us the following time and space bounds.

**Corollary 23** *OBN can be solved in $O\big([\binom{m}{\lfloor m/2 \rfloor}^p (I+F)]n^2\big)$ time and $O\big([I+Fn]n\big)$ space, where $I = 2^{n-mp}(2^{\lfloor m/2 \rfloor} + 2^{\lceil m/2 \rceil} - 1)^p$, for any $m = 2, \ldots, n$ and $p = 0, \ldots, \lfloor n/m \rfloor$, assuming each $\mathcal{F}_v$ is downward closed and of size at most $F$.*

We observe that the number of ideals, $I$, dominates both the time and space requirements as long as the input size, roughly $F$, is not too large. Note that pruning the parent sets affects $F$ but has no effect on $I$. Since in our case, $I$ usually grows exponentially in $n$, exponentially large families of potential parent sets can be handled with negligible extra cost. To investigate this issue more carefully, we focus on the case where each node is allowed to have at most $k = \alpha n$ parents, with some *slope* $\alpha \leq 1/2$. How large can the slope $\alpha$ be, yet guaranteeing that the size of the input does not dominate the time and space requirements? Recall that now the term $Fn$ in the space bound can be replaced by $F$.

Next we present some lower bounds for $\alpha$. We note that the largest such slope varies depending on the scheme used. For a moment, let us focus on the $(\lfloor m/2 \rfloor * \lceil m/2 \rceil)^{\lfloor n/m \rfloor}$ scheme, and to simplify calculations, assume $m$ is even and $n$ divisible by $m$. For any fixed $m$, we bound the

largest slope by $\alpha_m$, as follows. It is well-known that $\sum_{i=0}^{\alpha_m n} \binom{n}{i}$, an upper bound for the number of potential parent sets per node, is at most $2^{H(\alpha_m)n}$, where $H$ is the binary entropy function (for a proof, see for example Flum and Grohe (2006, p. 427)). On the other hand, every partial order in the system $(m/2 * m/2)^{n/m}$ has $((2^{m/2+1}-1)^{1/m})^n$ ideals. Thus, the number of ideals dominates the space and time requirements if $2^{H(\alpha_m)n} \leq ((2^{m/2+1}-1)^{1/m})^n$, equivalently, $H(\alpha_m) \leq 1/m \log_2(2^{m/2+1}-1)$. Solving this inequality numerically gives us a bound $\alpha_m$. Table 1 shows the $\alpha_m$ for each even $m \leq 26$.

| $m$ | $\alpha_m$ | $m$ | $\alpha_m$ | $m$ | $\alpha_m$ |
|---|---|---|---|---|---|
| 2 | 0.238 | 12 | 0.139 | 20 | 0.127 |
| 4 | 0.190 | 14 | 0.135 | 22 | 0.125 |
| 6 | 0.167 | 16 | 0.131 | 24 | 0.124 |
| 8 | 0.153 | 18 | 0.129 | 26 | 0.123 |
| 10 | 0.145 | | | | |

Table 1: Bounds on the maximum indegree slopes for the scheme $(m/2 * m/2)^{n/m}$.

Since the partial orders in $(m/2 * m/2)^{n/m}$ have as many or fewer ideals than the partial orders in $(m/2 * m/2)^p$, for $p \leq n/m$, we have the following characterization.

**Corollary 24** *OBN can be solved in* $O(((\binom{m}{m/2}2^{n-mp}(2^{m/2+1}-1))^p n^2)$ *time and* $O(2^{n-mp}(2^{m/2+1}-1)^p n)$ *space for any* $p = 0,1,2,\ldots,\lfloor n/m \rfloor$ *and* $m = 2,4,6,\ldots 26$*, provided that each node has at most* $\alpha_m n$ *parents, with* $\alpha_m$ *as given in Table 1.*

For example, the maximum indegree in 30-node DAGs can be set to $\lfloor 0.238 \times 30 \rfloor = 7$ in the pairwise scheme $(1*1)^p$ and to $\lfloor 0.139 \times 30 \rfloor = 4$ in the scheme $(6*6)^p$. A larger maximum indegree may render the input size dominate the number of ideals in the time and space requirements. In the next subsection we provide some empirical results, which suggest that the bounds in Table 1 are only slightly conservative.

### 6.4 Empirical Results

We have implemented the presented algorithm in the C++ language into a publicly available computer program BOSON (Bucket Order Scheme for Optimal Networks).[4] As the implementation is not fully optimized, the empirical results reported below should be viewed rather as a proof of concept.[5] We tested our implementation varying the number of nodes $n$, bucket order sizes $m$, and the number of parallel bucket orders $p$. The experiments were run on Intel Xeon R5540 processors, each with 32 GB of RAM.

We examined the running time for the limit of 16 GB of memory, letting the number of nodes $n$ vary from 25 to 34, with maximum indegree set to 3. The local scores were taken as given, so

4. BOSON is available at `www.csc.kth.se/~pekkapa/code/boson-1.0.tar.gz`.
5. We found the Silander-Myllymäki implementation (Silander and Myllymäki, 2006) about five times faster when the algorithms were run on the same setting, that is, running our algorithm on the trivial order on the nodes. It is likely that also the space usage can be lowered by a similar small factor by implementing Remark 17. Taken together, such improvements would allow us to deal with networks with two to three additional nodes compared to the present implementation, with the same time and space resources.

computing them is not included in the running time estimates. Since we do not prune potential parent sets based on the local scores, we have them for all possible parent sets of size at most the maximum indegree. Note that the actual scores are irrelevant when measuring the time and space usage. First we estimated the smallest bucket order size $m$ that yields a memory requirement of 16 GB or less. Then we ran Algorithm 1 for a partial order from the POS $(\lceil m/2 \rceil, \lfloor m/2 \rfloor)^1$ and gauged the running time. Finally, the measured running time was multiplied by the cardinality of the POS to get an estimate of the total running time. As all partial orders in the POS yield identical time and space requirements, there is no issue with estimation error or variance. Table 2 shows the results. We observe that, as expected, the time requirement grows rapidly with $n$: An optimal 25-node DAG can be found in about 25 minutes, while a 30-node DAG requires over 16 days of CPU time. Finding an optimal 34-node DAG is feasible using large-scale parallelization: with 1000 processors it takes about 6 days.

| $n$ | $p$ | $m$ | Time per PO | Cover size | Total time |
|---|---|---|---|---|---|
| 25 | 0 | 0 | 0.42 | 1 | 0.42 |
| 26 | 1 | 3 | 0.44 | 3 | 1.34 |
| 27 | 1 | 5 | 0.49 | 10 | 4.9 |
| 28 | 1 | 8 | 0.35 | 70 | 24.8 |
| 29 | 1 | 10 | 0.39 | 252 | 97.7 |
| 30 | 1 | 12 | 0.43 | 924 | 394 |
| 31 | 1 | 14 | 0.49 | 3432 | 1671 |
| 32 | 1 | 16 | 0.53 | 12 870 | 6784 |
| 33 | 1 | 18 | 0.83 | 48 620 | 40 332 |
| 34 | 1 | 20 | 0.78 | 184 756 | 144 930 |

Table 2: Time requirements for finding optimal Bayesian networks, for varying number of nodes when space is limited to 16 GB. Columns: $n$ is the number of nodes; $p$ is the number of parallel bucket orders; $m$ is the size of the balanced bucket order; *time per PO* is the running time (in CPU hours) per partial order; *cover size* is the number of partial orders in the cover; *total time* is the total running times (in CPU hours), that is, time per partial order multiplied by the size of the cover.

Next we studied how different schemes affect the running times and the space usage in practice. We analyzed two specializations of the generalized bucket order scheme $(\lceil m/2 \rceil * \lfloor m/2 \rfloor)^p$: the *practical scheme*, where $p = 1$, and the pairwise scheme, where $m = 2$. The results are shown in Tables 3 and 4. As expected, the practical scheme yields a clearly better space-time tradeoff than the pairwise scheme. This is perhaps even more clearly pronounced in Figure 3, which shows the empirical tradeoffs in the space-time plane along with the analytical bounds (Corollary 23). We see that, in general, the empirical and analytical bounds are in a good agreement, the analytical bounds being slightly conservative for medium $m$ (the practical scheme) and medium $p$ (the pairwise scheme).

We also investigated the influence of the maximum indegree $k$ on various characteristics of our implementation. Note that, in effect, $k$ parameterizes the number of potential parent sets, which in turn determines the computational complexity. Thus the observations readily extend to the case when the number of potential parent sets is reduced using an appropriate pruning proce-

| $m$ | $p$ | Space | Total time | $m$ | $p$ | Space | Total time |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 21248 | 1.01 | 14 | 1 | 331 | 32.51 |
| 2 | 1 | 15936 | 1.13 | 15 | 1 | 248 | 43.97 |
| 3 | 1 | 13280 | 1.41 | 16 | 1 | 166 | 64.35 |
| 4 | 1 | 9296 | 1.75 | 17 | 1 | 124 | 87.79 |
| 5 | 1 | 7304 | 2.13 | 18 | 1 | 83 | 129.65 |
| 6 | 1 | 4980 | 2.80 | 19 | 1 | 62 | 171.93 |
| 7 | 1 | 3818 | 3.57 | 20 | 1 | 41 | 256.61 |
| 8 | 1 | 2573 | 4.96 | 21 | 1 | 31 | 342.92 |
| 9 | 1 | 1950 | 6.88 | 22 | 1 | 21 | 489.88 |
| 10 | 1 | 1307 | 9.06 | 23 | 1 | 16 | 638.48 |
| 11 | 1 | 986 | 12.15 | 24 | 1 | 10 | 976.50 |
| 12 | 1 | 659 | 17.43 | 25 | 1 | 8 | 1444.53 |
| 13 | 1 | 495 | 24.31 | 26 | 1 | 5 | 2600.15 |

Table 3: Running times (in CPU hours) and space usage (in MB) of the practical scheme $(\lceil m/2 \rceil * \lfloor m/2 \rfloor)^1$ with $2 \leq m \leq 26$ and $n = 26$.

| $m$ | $p$ | Space | Total time | $m$ | $p$ | Space | Total time |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 21248 | 1.01 | 2 | 7 | 2836 | 12.79 |
| 2 | 1 | 15936 | 1.13 | 2 | 8 | 2127 | 20.25 |
| 2 | 2 | 11952 | 1.48 | 2 | 9 | 1595 | 32.36 |
| 2 | 3 | 8964 | 2.09 | 2 | 10 | 1197 | 51.66 |
| 2 | 4 | 6723 | 3.19 | 2 | 11 | 897 | 78.96 |
| 2 | 5 | 5042 | 4.97 | 2 | 12 | 673 | 126.52 |
| 2 | 6 | 3782 | 7.90 | 2 | 13 | 505 | 198.88 |

Table 4: Running times (in CPU hours) and space usage (in MB) of the pairwise scheme $(1 * 1)^p$ with $1 \leq p \leq 13$ and $n = 26$.

dure (de Campos and Ji, 2011). We analyzed the scheme $(10 * 10)^1$ with $n = 20$ nodes, varying $k$ from 1 to 8. For interpretation of the results, shown in Table 5, it is useful to note that the partial orders in question have 2047 ideals. For comparison, the (worst-case) input size is 1160 for $k = 3$ and 5036 for $k = 4$. So we conclude that the number of ideals dominates the input size precisely when $k \leq 3$. Now, recall that the bounds in Corollary 24 guarantee this only for $k \leq 2$, indicating that the analytical bounds are not tight but slightly pessimistic. Table 5 shows also, perhaps somewhat surprisingly, that even if the input size for $k = 5$ is more than 3 times the input size for $k = 4$, the total running time less than doubles. This can be explained by the fact that the respective increase in the tail accesses, from 46520 to 160260, does not yet pay off, since the number of computation steps that are not related to the input (nor the maximum indegree) appears to be as large as 358300. For larger $k$, the running time will grow about linearly with the number of tail accesses.

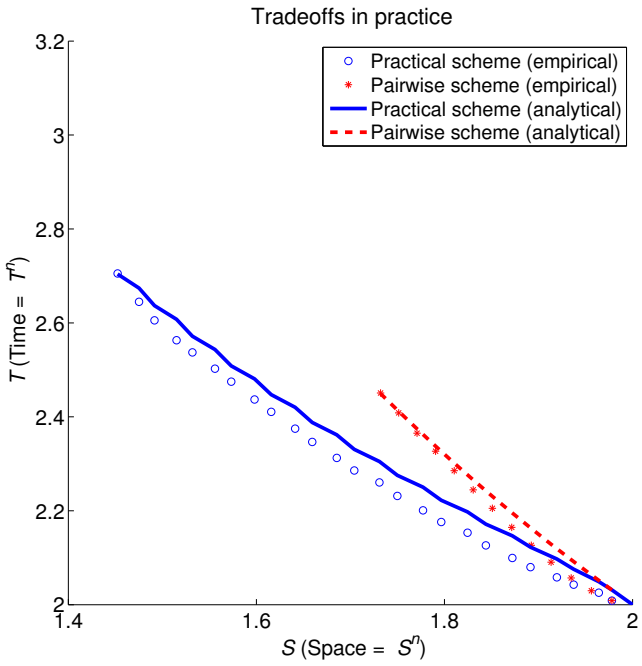



Figure 3: Empirical and analytical time and space requirements of the pairwise scheme and the practical scheme, for $n = 26$ nodes. The empirical time and space requirements were normalized by dividing them, respectively, by the empirical time and space requirement of the basic dynamic programming algorithm (the case of the trivial order). We took the 26th root of the ratio, multiplied by 2, as the normalized value. The analytical bounds were normalized analogously.

## 7. Discussion

While this work is largely theoretical, it is mainly motivated by practical needs for solving larger problem instances to guaranteed optimum. The presented methods address those needs by two means. First, the algorithms truly decrease the space requirement of existing dynamic programming algorithms, which makes it possible to process larger instances on a typical modern computer with some limited amount of RAM, say 16 GB. Second, the algorithms can be easily and efficiently implemented to run in parallel on practically as many processors as available. As large computer clusters and grid computing with thousands of processing units are becoming more common, we believe the presented schemes bring larger networks to within the reach of exact algorithms.

Figure 4 summarizes the relationship of the time and the space requirement of different schemes presented in this work. Compared to the naive two-bucket scheme, the more general bucket order schemes yield significantly more efficient exchange of time and space resources. Our most efficient schemes enable saving space by practically any factor, specified by the user, at the cost of increasing the runtime by about the same factor. It should be noted, though, that none of the schemes enables saving space without an increase in the runtime. Whether more efficient schemes exist or whether considerable space savings are possible even at no increase in the runtime, is a difficult open ques-

| $k$ | Input size | Regular accesses | Tail accesses | Total time |
|---|---|---|---|---|
| 1 | 20 | 210 | 90 | 2.4 |
| 2 | 191 | 1020 | 1350 | 2.5 |
| 3 | 1160 | 3060 | 9840 | 2.8 |
| 4 | 5036 | 6420 | 46500 | 4.3 |
| 5 | 16664 | 10200 | 160260 | 8.5 |
| 6 | 43796 | 13140 | 429480 | 18.3 |
| 7 | 94184 | 14700 | 932160 | 35.5 |
| 8 | 169766 | 15240 | 1687530 | 60.4 |

Table 5: Characteristics of the practical scheme $(10*10)^1$ with $n = 20$ nodes, for varying maximum indegree $k$. Columns: *input size* is the number of parent sets per node; *regular accesses* is the number of accesses to input when the parent set is an ideal; *tail accesses* is the number of accesses to input when the parent set is in the tail of an ideal; *total time* is the running time in hours. Regular and tail accesses are per partial order.

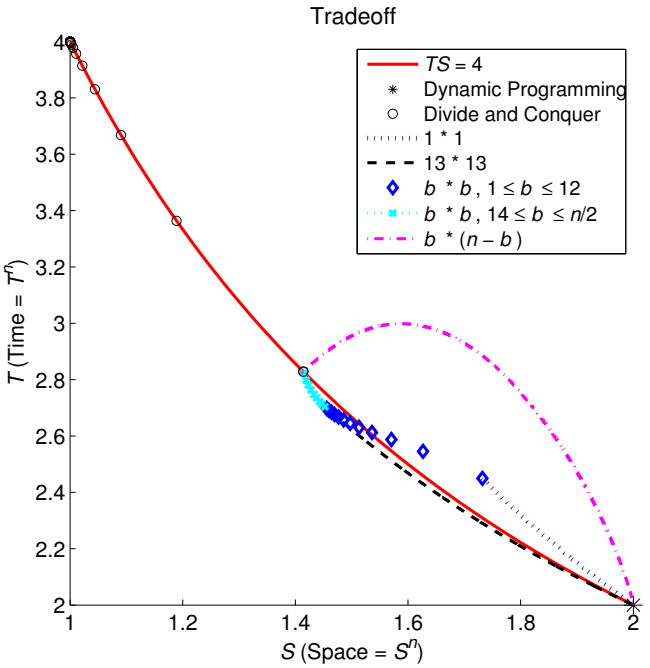



Figure 4: Comparison of time and space requirements of different schemes. The base of the exponential time and space requirement is shown on the vertical and horizontal axis, respectively.

tion. The difficulty is suggested by the fact that, no more efficient schemes are known for the classic traveling salesman problem, a simpler cousin of the BN problem studied here.

We presented the results under the assumption that the local scores are given as input and stored in the memory. While this approach sometimes yields an unnecessary large lower bound for the space requirement, we find the assumption both convenient and most relevant for practical implementations. Namely, the number of potential parent sets, particularly after pruning, is typically relatively small compared to the amount of memory available—the bottleneck regarding memory consumption is the number of ideals of the partial orders. From a theoretical point of view, it is, however, interesting to also consider the alternative approach, in which the local scores are always computed anew from given data (in time polynomial in $n$) when needed in the calculations. For example, this approach combined with the divide and conquer scheme yields an algorithm that takes only polynomial space, albeit the runtime is within a polynomial factor to $8^n$ (with no restrictions on the number of potential parent sets, cf. Corollary 6). It is an intriguing open question whether much faster polynomial-space algorithms exist.

Recently, also other approaches to save space and enhance parallelization have been proposed. Specifically, Malone et al. (2011) and Yuan et al. (2011) present a tighter implementation of the basic DP algorithm and demonstrate moderate space savings (proportional to the squareroot of the number of nodes). Compared to the basic DP algorithm, it enables finding optimal BNs with a couple of more nodes. Tamada et al. (2011) present yet a different approach. They divide the work of the DP algorithm into several subtasks that overlap only little. The subtasks can be solved in parallel, which enables efficient parallelization with a tolerable communication overhead. Compared to our approach, their algorithm has two notable drawbacks, however: First, it has to be run on a supercomputer that enables fast (and synchronized) communication between the processing units, whereas in our approach communication is not an issue. Second, the total amount of memory needed remains large: a 32-node instance was solved using over 800 GB of memory in total (with 256 CPU cores in less than six days) (Tamada et al., 2011).

Whether typical instances met in practice can be solved significantly faster than the present worst-case bounds would suggest, is one of the most central open questions in the research area. There is some hope for an affirmative answer. For an extreme example, consider an instance for which the empty DAG (that is, no arcs) happens to be optimal. That particular instance can be solved by just letting each node take its best-scoring parent set, namely the empty set, with no worries about the acyclicity constraint. This can be done relatively fast if the number of potential parent sets is small, for instance, due to an assumed bound on the number of parents per node. Quite recently, this observation has been taken further using branch-and-bound (de Campos et al., 2009; de Campos and Ji, 2011; Etminani et al., 2010) and linear programming ideas (Jaakkola et al., 2010; Cussens, 2011). The reported results show promising scalability of these methods under some favorable conditions, of which nature has, unfortunately, not yet been satisfactorily characterized. In particular, no worst-case upper bounds for their runtime are known that would be competitive to the bounds of the DP algorithms. What is worse, the methods are known to be unfeasibly slow even for some small benchmark instances that DP algorithms solve in a few minutes. Indeed, an illustrative example is the 19-variable image segmentation data set (from the UCI machine learning repository), on which Cussens's (2011) implementation of the linear programming method does not terminate within two hours, using the BIC scoring that results in a total of 8164 potential parent sets after pruning (B. Malone, personal communication). In comparison, the DP algorithms solve the instance in a few seconds. Note that our schemes further enable solving that instance in very small space, say, using about $2^{15}$ bytes, by increasing the total runtime by a factor around $2^{19-15} = 16$, or by running the algorithm on that many processors in parallel. Nevertheless, the branch-and-bound

and linear programming techniques complement the DP methodology—hybrids between them is a plausible topic of future research.

Finally, it is worth noting that sometimes a "good" partial order of the nodes is actually known to the modeller. By "good" we mean one that encodes prior knowledge about the DAG to be found and that has a relatively small number of ideals. For example, in causal discovery and dynamic Bayesian networks it is quite expected that the precedence order is known for many pairs of nodes. In such a case, one needs to run the presented DP algorithm on just that partial order, thereby using much less time and space than the basic DP algorithm, which essentially cannot exploit given precedence constraints (besides excluding some parent sets for some nodes). In other words, the presented DP algorithm generalizes the basic DP algorithm to fully exploit given precedence constraints, if any.

## Acknowledgments

The authors thank Kustaa Kangas, Petteri Kaski, Brandon Malone, and Teppo Niinimäki for valuable discussions. We also thank the anonymous reviewers of earlier versions of the manuscript for concrete suggestions that enabled us to improve the presentation and clarity of this paper substantially. This work was supported in part by the Academy of Finland, grants 125637, 218153, and 255675.

## References

R. Bellman. Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM*, 9(1):61–63, 1962.

A. Björklund and T. Husfeldt. Exact algorithms for exact satisfiability and number of perfect matchings. *Algorithmica*, 52:226–249, 2008.

H. L. Bodlaender, F. V. Fomin, A. M. C. A. Koster, D. Kratsch, and D. M. Thilikos. On exact algorithms for treewidth. In *Proceedings of the 14th Annual European Symposium on Algorithms (ESA)*, pages 672–683, 2006.

H. L. Bodlaender, F. V. Fomin, A. M. C. A. Koster, D. Kratsch, and D. M. Thilikos. A note on exact algorithms for vertex ordering problems on graphs. *Theory of Computing Systems*, 50(3): 420–432, 2012.

G. Brightwell and P. Winkler. Counting linear extensions. *Order*, 8:225–242, 1991.

D. M. Chickering. *Learning from Data: Artificial Intelligence and Statistics V*, chapter Learning Bayesian networks is NP-Complete, pages 121–130. Springer-Verlag, 1996.

D. M. Chickering, D. Heckerman, and C. Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004.

H. Chockler and J. Y. Halpern. Responsibility and blame: A structural-model approach. *Journal of Artificial Intelligence Research*, 22:93–115, 2004.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

J. Cussens. Bayesian network learning with cutting planes. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 153–160, 2011.

B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2003.

C. P. de Campos and Q. Ji. Efficient structure learning of Bayesian networks using constraints. *Journal of Machine Learning Research*, 12:663–689, 2011.

C. P. de Campos, Z. Zeng, and Q. Ji. Structure learning of Bayesian networks using constraints. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, pages 113–120, 2009.

L. M. de Campos. A scoring function for learning Bayesian networks based on mutual information and conditional independence tests. *Journal of Machine Learning Research*, 7:2149–2187, 2006.

K. Etminani, M. Naghibzadeh, and A. R. Razavi. Globally optimal structure learning of Bayesian networks from data. In *Proceedings of the 20th International Conference on Artificial Neural Networks (ICANN)*, pages 101–106, 2010.

J. Flum and M. Grohe. *Parametrized Complexity Theory*. Springer, 2006.

Y. Gurevich and S. Shelah. Expected computation time for Hamiltonian path problem. *SIAM Journal of Computation*, 16(3):486–502, 1987.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

M. Held and R. M. Karp. A dynamic programming approach to sequencing problem. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962.

T. Jaakkola, D. Sontag, A. Globerson, and M. Meila. Learning Bayesian network structure using LP relaxations. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, JMLR: W&CP, pages 358–365, 2010.

M. Koivisto. Parent assignment is hard for the MDL, AIC, and NML costs. In *Proceedings of the 19th Annual Conference on Learning Theory (COLT)*, pages 289–303, 2006.

M. Koivisto and P. Parviainen. A space–time tradeoff for permutation problems. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 484–492, 2010.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

B. Malone, C. Yuan, E. A. Hansen, and S. Bridges. Improving the scalability of optimal Bayesian network learning with external-memory frontier breadth-first branch and bound search. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 479–488, 2011.

S. Ott and S. Miyano. Finding optimal gene networks using biological constraints. *Genome Informatics*, 14:124–133, 2003.

P. Parviainen and M. Koivisto. Exact structure discovery in Bayesian networks with less space. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 436–443, 2009.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, 1988.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, 2000.

S. Provan and M. O. Ball. On the complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal of Computing*, 12:777–788, 1983.

W. Savitch. Relationship between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4:177–192, 1970.

T. Silander and P. Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 445–452, 2006.

A. P. Singh and A. W. Moore. Finding optimal Bayesian networks by dynamic programming. Technical Report CMU-CALD-05-106, Carnegie Mellon University, June 2005.

P. Spirtes and C. Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9:62–72, 1991.

G. Steiner. On the complexity of dynamic programming with precedence constraints. *Annals of Operations Research*, 26:103–123, 1990.

Y. Tamada, S. Imoto, and S. Miyano. Parallel algorithm for learning optimal Bayesian network structure. *Journal of Machine Learning Research*, 12:2437–2459, 2011.

T. S. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 255–270, 1990.

C. Yuan, B. Malone, and X. Wu. Learning optimal Bayesian networks using A* search. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2186–2191, 2011.

Journal of Machine Learning Research 14 (2013) 1417-1421 Submitted 2/12; Revised 11/12; Published 5/13

# JKernelMachines: A Simple Framework for Kernel Machines

**David Picard** PICARD@ENSEA.FR
*ETIS - ENSEA/CNRS/Université de Cergy Pontoise*
*6 avenue du Ponceau*
*95014 Cergy-Pontoise Cedex, France*

**Nicolas Thome** NICOLAS.THOME@LIP6.FR
**Matthieu Cord** MATTHIEU.CORD@LIP6.FR
*LIP6 - UPMC Univ Paris 6*
*4 place Jussieu*
*75005 Paris, France*

**Editor:** Cheng Soon Ong

## Abstract

*JKernelMachines* is a Java library for learning with kernels. It is primarily designed to deal with custom kernels that are not easily found in standard libraries, such as kernels on structured data. These types of kernels are often used in computer vision or bioinformatics applications. We provide several kernels leading to state of the art classification performances in computer vision, as well as various kernels on sets. The main focus of the library is to be easily extended with new kernels. Standard SVM optimization algorithms are available, but also more sophisticated learning-based kernel combination methods such as Multiple Kernel Learning (MKL), and a recently published algorithm to learn powered products of similarities (Product Kernel Learning).

**Keywords:** classification, support vector machines, kernel, computer vision

## 1. Introduction

Support Vector Machines (SVM) are widely recognized as state-of-the art classifiers for supervised learning problems. When using SVM, the similarity measure has to be a *kernel* function. Efficient libraries already exist implementing SVM optimization algorithms (liblinear, libsvm, weka, SHOGUN, *etc*). However, these solutions are generally limited to vectorial data and simple kernels (linear or Gaussian for example). It is not straightforward to use them when dealing with structured data, for example, sets, graphs, or strings. Designing similarities for such inputs is essential in areas such as bioinformatic and computer vision.

For image classification or object detection tasks, state of the art methods rely on similarity functions based on sets of features (edge, color, texture, *etc*) extracted locally on different image regions. Then, two main strategies can be used to design a similarity between sets of local descriptors, referred as "Bag of Features" (BoF). First, one can rely on the "Bag of Words" (BoW) model, where the local descriptors are quantized using a dictionary, and the projected features are aggregated to produce a histogram of visual words. An alternative to the BoW model is to directly define a kernel function between sets of local descriptors, thus ignoring the quantization step. The feature extraction and projection can also be included in the kernel function (as a first explicit mapping), which then becomes a similarity measure of high complexity. Recently, many kernels on sets have

©2013 David Picard, Nicolas Thome and Matthieu Cord.

been proposed in the computer vision community, for example by Bo and Sminchisescu (2009) or by T. Tuytelaars and Darrell (2011).

To sum it up, designing well adapted kernel functions is attracting a lot of research in the community. *JKernelMachines* is dedicated to facilitate the use of such exotic kernels. It is thus not designed as an end user program, but as a library to be used in computer vision (or bioinformatics, *etc*) pipelines. The goals of the library are to provide:

- An efficient framework for kernels:
  - Easy and intuitive use of the library
  - Very easy and rapid development of new kernels
- An effective SVM machine learning toolbox:
  - Up to date learning algorithms (primal, dual) with kernel combinations (MKL)
  - Good computational performances on current computer vision benchmarks

To fulfill these goals, *JKernelMachines* is designed as a pure Java implementation without any further dependency than a standard JRE. It is licensed under the terms of GPLv3.

## 2. Description of the Library

The backbone of the library is the definition of data types and kernels. In order to use any type of input space, the library makes heavy use of the Java Generics. Kernels are defined on generic input space, with specific implementation delegated to child classes. Classifiers that use a kernel function will end up with the right similarity thanks to polymorphism.

### 2.1 Kernels

Given a sample of generic type T, the library provides a parent class for all kernel functions:

```
public abstract class Kernel<T> implements Serializable {
  /** compute the kernel similarity between two elements of input space  */
  public abstract double valueOf(T t1, T t2);
}
```

To create a new kernel, one basically has to extend this class and implement the valueOf() method. For example, the source code below corresponds to the following kernel on bags of vectors of double:

$$K(\{\mathbf{x}_i\}_i, \{\mathbf{x}_j\}_j) = \sum_{\mathbf{x}_i \in \{\mathbf{x}_i\}_i, \mathbf{x}_j \in \{\mathbf{x}_j\}_j} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2}}.$$

```
public class DoubleBagGaussian extends Kernel<List<double[]>> {
  public double valueOf(List<double[]> l1, List<double[]> l2) {
    double sum = 0., d;
    for(double[] t1 : l1)
      for(double[] t2 : l2) {
        for(d=0., int i=0 ; i<t1.length ; i++){ d += (t2[i]-t1[i])*(t2[i]-t1[i]); }
        sum += Math.exp(-d/2.);
      }
    return sum;
  }
}
```

Currently, *JKernelMachines* provides the following kernels:

- Standard kernels on vector (Linear, Gaussian, etc) for various types (int, double, etc)
- A wide variety of Gaussian kernels ($\chi^2$ distance, subset of selected axes, etc)
- Combination of kernels (weighted sum and weighted product of generic minor kernels)
- Generic kernels on lists (ordered or unordered, weighted or not)
- Kernels defined by a custom Gram matrix and kernels with various caching strategies

### 2.2 Classifiers

Classifiers are also defined on generic data types, and use an instance of a specific Kernel. In order to be able to work on any type of input space, the data type of the samples are encapsulated in a generic class called *TrainingSample*. This class contains the sample of generic type T and the associated label (if the sample is labeled):

```
public class TrainingSample<T> implements Serializable {
  T sample;
  int label;
}
```

Training of an SVM classifier consists of calling the method train with a list of TrainingSample as parameters. For example, an SVM with GaussianKernel on vectors of double would be done as follows:

```
List<TrainingSample<double[]>> train = new ArrayList<TrainingSample<double[]>>();
// add code to feed the list with labeled samples
DoubleGaussL2 k = new DoubleGaussL2();
LaSVM<double[]> lasvm = new LaSVM<double[]>(k);
lasvm.train(train);
```

In this example, the evaluation of a new sample is as simple as:

```
double[] sample;  // add code to fill the sample
double value = lasvm.valueOf(sample);
```

Currently, *JKernelMachines* implements the following algorithms:

- SVM: LaSVM from Bordes et al. (2005), LaSVM-I from Ertekin et al. (2011), SMO from Platt (1999), Transductive SVM using S$^3$VM*Light* from Joachims (1999)
- Density estimator: One-class SVM using SMO, Parzen window estimator
- Multiple kernel: SimpleMKL from Rakotomamonjy et al. (2008), TS-MKL from Kumar et al. (2012), Gradient-descent based $\ell_p$-Norm MKL from Kloft et al. (2011), Product combinations of Gaussian kernels from Picard et al. (2012)
- Fast primal linear SVM on double vectors: Pegasos from Singer and Srebro (2007), SGD from Bottou (2010), SGDQN from Bordes et al. (2009)

Since most of these algorithms are implemented using the Kernel interface, they can be used with any positive semi-definite implementation of the Kernel class. Examples are provided with the library (see the *example* package).

| | Weka | JKernelMachines | | |
|---|---|---|---|---|
| | SMO | LaSVM | SMO | LaSVM-I |
| ionosphere | $86.1 \pm 4.1$ | $92.8 \pm 3.6$ | $91.5 \pm 2.5$ | $90.3 \pm 3.2$ |
| heart | $84.0 \pm 3.8$ | $81.9 \pm 3.8$ | $83.1 \pm 4.3$ | $84.4 \pm 5.0$ |
| breast-cancer | $97.3 \pm 1.3$ | $97.1 \pm 1.0$ | $94.2 \pm 1.8$ | $97.4 \pm 1.5$ |
| german numbers | $75.4 \pm 2.3$ | $75.0 \pm 3.2$ | $68.9 \pm 2.9$ | $75.4 \pm 2.1$ |

Table 1: Mean accuracies for *Weka* and *JKernelMachines* on UCI data sets.

## 3. Validation

We first present results comparing *JKernelMachines* to the well known *Weka* library on common data sets from the UCI repository. We used the example class *CrossValidationExample* of package *fr.lip6.jkernelmachines.example* to produce the results. Following the setup of Rakotomamonjy et al. (2008), 20 random splits were drawn (80% for train, 20% for test).

Table 3 summaries the results. On the *ionosphere* data set, *JKernelMachines* offers significantly better results than *Weka*. On *heart*, *breast cancer* and *german numbers* both libraries gave about the same results. Regarding time, we found *JKernelMachines* implementation of *LaSVM* to be significantly faster than *Weka* (both libraries were fast enough).

Compared to other libraries more centered on kernels, like SHOGUN or OpenKernel, *JKernelMachines* clearly aims at adding new kernels easily and reliably by allowing the user to set any code inside the kernel computation. For instance, SHOGUN allows to use kernels defined by a custom Gram matrix, which has to be computed before the optimization. The custom matrix method may not be easy to handle when the kernel is relying on a parametric feature extraction process, where the parameters are to be varied (e.g., filter banks in signal processing applications). Moreover, in case of online evaluation, a new custom matrix has to be computed for each incoming test sample and manually set as the kernel used by the classifier, whereas a *JKernelMachines* classifier would work without any modification. OpenKernel allows for more complex kernels using base kernels (including n-gram kernels) and combination, but again the data have to be in a specific file format generated in advance which leads to the same difficulties.

*JKernelMachines* was used for PASCAL VOC challenges both in 2009 and in 2010,[1] and gave comparative results to other teams using the same features. The achievement in these challenges is two-folds: first, it validates the implementation of the learning algorithms. Second, it shows the library is usable on real world challenges regarding computing resources. Some of these experiments were published in Picard et al. (2010).

## References

L. Bo and C. Sminchisescu. Efficient match kernel between sets of features for visual recognition. In *Advances in Neural Information Processing Systems*, December 2009.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, September 2005.

A. Bordes, L. Bottou, and P. Gallinari. Sgd-qn: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research*, 10:1737–1754, July 2009.

1. Results can be found at `http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2009/`.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of the 19th International Conference on Computational Statistics*, pages 177–187, Paris, France, August 2010.

S. Ertekin, L. Bottou, and C. L. Giles. Ignorance is bliss: Non-convex online support vector machines. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 33(2):368–381, February 2011.

T. Joachims. Transductive inference for text classification using support vector machines. In *ICML*, pages 200–209, 1999.

M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. Lp-norm multiple kernel learning. *Journal of Machine Learning Research*, 12:953–997, Mar 2011.

A. Kumar, A. Niculescu-Mizil, K. Kavukcuoglu, and H. Daume III. A binary classification framework for two-stage multiple kernel learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1295–1302, New York, NY, USA, July 2012.

D. Picard, N. Thome, and M. Cord. An efficient system for combining complementary kernels in complex visual categorization tasks. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 3877–3880, 2010.

D. Picard, N. Thome, M. Cord, and A. Rakotomamonjy. Learning geometric combinations of gaussian kernels with alternating quasi-newton algorithm. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 79–84, 2012.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods*, pages 185–208. MIT Press, 1999.

A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, November 2008.

Y. Singer and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *In ICML*, pages 807–814, 2007.

K. Saenko T. Tuytelaars, M. Fritz and T. Darrell. The nbnn kernel. In *Proceedings of 13th International Conference on Computer Vision*, 2011.

Journal of Machine Learning Research 14 (2013) 1423-1459 Submitted 4/08; Revised 8/12; Published 5/13

# Asymptotic Results on Adaptive False Discovery Rate Controlling Procedures Based on Kernel Estimators

**Pierre Neuvial*** PIERRE.NEUVIAL@GENOPOLE.CNRS.FR
*Laboratoire Statistique et Génome*
*Université d'Évry Val d'Essonne*
*UMR CNRS 8071 – USC INRA*
*23 boulevard de France*
*91 037 Évry, France*

**Editor:** Olivier Teytaud

## Abstract

The False Discovery Rate (FDR) is a commonly used type I error rate in multiple testing problems. It is defined as the expected False Discovery Proportion (FDP), that is, the expected fraction of false positives among rejected hypotheses. When the hypotheses are independent, the Benjamini-Hochberg procedure achieves FDR control at any pre-specified level. By construction, FDR control offers no guarantee in terms of power, or type II error. A number of alternative procedures have been developed, including plug-in procedures that aim at gaining power by incorporating an estimate of the proportion of true null hypotheses.

In this paper, we study the asymptotic behavior of a class of plug-in procedures based on kernel estimators of the density of the $p$-values, as the number $m$ of tested hypotheses grows to infinity. In a setting where the hypotheses tested are independent, we prove that these procedures are asymptotically more powerful in two respects: (i) a tighter asymptotic FDR control for any target FDR level and (ii) a broader range of target levels yielding positive asymptotic power. We also show that this increased asymptotic power comes at the price of slower, non-parametric convergence rates for the FDP. These rates are of the form $m^{-k/(2k+1)}$, where $k$ is determined by the regularity of the density of the $p$-value distribution, or, equivalently, of the test statistics distribution. These results are applied to one- and two-sided tests statistics for Gaussian and Laplace location models, and for the Student model.

**Keywords:** multiple testing, false discovery rate, Benjamini Hochberg's procedure, power, criticality, plug-in procedures, adaptive control, test statistics distribution, convergence rates, kernel estimators

## 1. Introduction

Multiple simultaneous hypothesis testing has become a major issue for high-dimensional data analysis in a variety of fields, including non-parametric estimation by wavelet methods in image analysis, functional magnetic resonance imaging (fMRI) in medicine, source detection in astronomy, and DNA microarray or high-throughput sequencing analyses in genomics. Given a set of observations corresponding either to a null hypothesis or an alternative hypothesis, the goal of multiple testing is to infer which of them correspond to true alternatives. This requires the definition of risk measures

∗. This work was partly done when the author was affiliated with Laboratoire de Probabilités et Modèles Aléatoires, Université Paris VII-Denis Diderot and INSERM U900/Institut Curie/Mines ParisTech.

©2013 Pierre Neuvial.

that are adapted to the large number of tests performed: typically $10^4$ to $10^6$ in genomics. The False Discovery Rate (FDR) introduced by Benjamini and Hochberg (1995) is one of the most commonly used and one of the most widely studied such risk measure in large-scale multiple testing problems. The FDR is defined as the expected proportion of false positives among rejected hypotheses. A simple procedure called the Benjamini-Hochberg (BH) procedure provides FDR control when the tested hypotheses are independent (Benjamini and Hochberg, 1995) or follow specific types of positive dependence (Benjamini and Yekutieli, 2001).

When the hypotheses tested are independent, applying the BH procedure at level $\alpha$ in fact yields $\mathrm{FDR} = \pi_0\alpha$, where $\pi_0$ is the unknown fraction of true null hypotheses (Benjamini and Yekutieli, 2001). This has motivated the development of a number of "plug-in" procedures, which consist in applying the BH procedure at level $\alpha/\hat{\pi}_0$, where $\hat{\pi}_0$ is an estimator of $\pi_0$. A typical example is the Storey-$\lambda$ procedure (Storey, 2002; Storey et al., 2004) in which $\hat{\pi}_0$ is a function of the empirical cumulative distribution function of the $p$-values.

In this paper, we consider an asymptotic framework where the number $m$ of tests performed goes to infinity. When $\hat{\pi}_0$ converges in probability to $\pi_{0,\infty} \in [\pi_0, 1)$ as $m \to +\infty$, the corresponding plug-in procedure is by construction asymptotically more powerful than the BH procedure, while still providing $\mathrm{FDR} \leq \alpha$. However, as FDR control only implies that the *expected* FDP is below the target level, it is of interest to study the *fluctuations* of the FDP achieved by such plug-in procedures around their corresponding FDR. This paper studies the influence of the plug-in step on the asymptotic properties of the corresponding procedure for a particular class of estimators of $\pi_0$, which may be written as kernel estimators of the density of the $p$-value distribution at 1.

## 2. Background and Notation

In this section, we introduce the multiple testing setting considered in this paper, and define two central concepts: plug-in procedures and criticality.

### 2.1 Multiple Testing Setting

We consider a test statistic $X$ distributed as $F_0$ under a null hypothesis $\mathcal{H}_0$ and as $F_1$ under an alternative hypothesis $\mathcal{H}_1$. We assume that for $a \in \{0,1\}$, $F_a$ is continuously differentiable, and that the corresponding density function, which we denote by $f_a$, is positive. This testing problem may be formulated in terms of $p$-values instead of test statistics. The $p$-value function is defined as $p : x \mapsto \mathbb{P}_{\mathcal{H}_0}(X \geq x) = 1 - F_0(x)$ for one-sided tests and $p : x \mapsto \mathbb{P}_{\mathcal{H}_0}(|X| \geq |x|)$ for two-sided tests. As $F_0$ is continuous, the $p$-values are uniform on $[0,1]$ under $\mathcal{H}_0$. For consistency we denote by $G_0$ the corresponding distribution function, that is, the identity function on $[0,1]$. Under $\mathcal{H}_1$, the distribution function and density of the $p$-values are denoted by $G_1$ and $g_1$, respectively. Their expression as functions of the distribution of the test statistics are recalled in Proposition 1 below in the case of one- and two-sided $p$-values. For two-sided $p$-values, we assume that the distribution function of the test statistics under $\mathcal{H}_0$ is symmetric (around 0):

$$\forall x \in \mathbb{R}, F_0(x) + F_0(-x) = 1\,. \tag{1}$$

Assumption (1) is typically satisfied in usual models such as Gaussian or Laplace (double exponential) models. Under Assumption (1), the two-sided $p$-value satisfies $p(x) = 2\,(1 - F_0(|x|))$ for any $x \in \mathbb{R}$.

**Proposition 1 (One- and two-sided $p$-values)** *For $t \in [0,1]$, let $q_0(t) = F_0^{-1}(1-t)$. The distribution function $G_1$ and the and density function $g_1$ of the p-value under $\mathcal{H}_1$ at $t$ satisfy the following:*

1. *for a one-sided p-value, $G_1(t) = 1 - F_1(q_0(t))$ and $g_1(t) = (f_1/f_0)(q_0(t))$;*
2. *for a two-sided p-value, $G_1(t) = 1 - F_1(q_0(t/2)) + F_1(-q_0(t/2))$ and $g_1(t) = 1/2\,((f_1/f_0)(q_0(t/2)) + (f_1/f_0)(-q_0(t/2)))$.*

The assumption that $f_1$ is positive entails that $g_1$ is positive as well. We further assume that

$$G_1 \text{ is concave.} \tag{2}$$

As $g_1$ is a function of the likelihood ratio $f_1/f_0$ and the non-increasing function $q_0$, Assumption (2) may be characterized as follows:

**Lemma 2 (Concavity and likelihood ratios)**
1. *For a one-sided p-value, Assumption* (2) *holds if and only if the likelihood ratio $f_1/f_0$ is non-decreasing.*
2. *For a two-sided p-value under Assumption* (1)*, Assumption* (2) *holds if and only if $x \mapsto (f_1/f_0)(x) + (f_1/f_0)(-x)$ is non-decreasing on $\mathbb{R}_+$.*

We consider a sequence of independent tests performed as described above and indexed by the set $\mathbb{N}^*$ of positive integers. We assume that either all of them are one-sided tests, or all of them are two-sided tests. This sequence of tests is characterized by a sequence $(\mathbf{H}, \mathbf{p}) = (H_i, p_i)_{i \in \mathbb{N}^*}$ , where for each $i \in \mathbb{N}^*$, $p_i$ is a $p$-value associated to the $i^{th}$ test, and $H_i$ is a binary indicator defined by

$$H_i = \begin{cases} 0 \text{ if } \mathcal{H}_0 \text{ is true for test } i \\ 1 \text{ if } \mathcal{H}_1 \text{ is true for test } i \end{cases} .$$

We also let $m_0(m) = \sum_{i=1}^{m}(1-H_i)$, and $\pi_{0,m} = m_0(m)/m$. Following the terminology proposed by Roquain and Villers (2011), we define the *conditional setting* as the situation where $\mathbf{H}$ is deterministic and $\mathbf{p}$ is a sequence of independent random variables such that for $i \in \mathbb{N}^*$, $p_i \sim G_{H_i}$. This is a particular case of the setting originally considered by Benjamini and Hochberg (1995), where no assumption was made on the distribution of $p_i$ when $H_i = 1$. In the present paper, we consider an *unconditional setting* introduced by Efron et al. (2001), which is also known as the "random effects" setting. Specifically, $\mathbf{H}$ is a sequence of random indicators, independently and identically distributed as $\mathcal{B}(1-\pi_0)$, where $\pi_0 \in (0,1)$, and conditional on $\mathbf{H}$, $\mathbf{p}$ follows the conditional setting, that is, the $p$-values satisfy $p_i|H_i \sim G_{H_i}$. This unconditional setting has been widely used in the multiple testing literature, see, for example, Storey (2003); Genovese and Wasserman (2004); Chi (2007a). In this setting, the $p$-values are independently, identically distributed as $G = \pi_0 G_0 + (1-\pi_0)G_1$, and $m_0(m)$ follows the binomial distribution $Bin(m, \pi_0)$.

**Remark 3** *We are assuming that $\pi_0 < 1$, which implies that the proportion $1 - \pi_{0,m}$ of true null alternatives does not vanish as $m \to +\infty$. While this restriction is natural in the unconditional setting considered in this paper, we note that our results do not apply to the "sparse" situation where $\pi_{0,m} \to 1$ as $m \to +\infty$.*

As $G_0$ is the identity function, the multiple testing model is entirely characterized by the two parameters $\pi_0$ and $G_1$ (or, equivalently, $\pi_0$ and $G$), where $G_1$ is itself entirely characterized by $F_0$ and $F_1$, by Proposition 1. The mixture distribution $G$ is concave if and only if Assumption (2) holds. More generally, we note that making a regularity assumption on $G_1$ (or $g_1$) is equivalent to making the same regularity assumption on $G$ (or $g$):

**Remark 4 (Differentiability assumptions)** *Throughout the paper, differentiability assumptions on the distribution of the p-values near 1 are expressed in terms of g, the (mixture) p-value density. As $g = \pi_0 + (1-\pi_0)g_1$, we note that they could equally be written in terms of $g_1$, the p-value density under the alternative hypothesis.*

### 2.2 Type I and II Error Rate Control in Multiple Testing

We define a multiple testing procedure $\mathcal{P}$ as a collection of functions $(\mathcal{P}_\alpha)_{\alpha\in[0,1]}$ such that for any $\alpha \in [0,1]$, $\mathcal{P}_\alpha$ takes as input a vector of $m$ $p$-values, and returns a subset of $\{1,\dots m\}$ corresponding to the indices of hypotheses to be rejected. For a given procedure $\mathcal{P}$ and a given $\alpha \in [0,1]$, the function $\mathcal{P}_\alpha$ will be called "Procedure $\mathcal{P}$ at (target) level $\alpha$". In this paper, we focus on *thresholding-based* multiple testing procedures, for which the rejected hypotheses are those with $p$-values less than a threshold. Each possible value for the threshold corresponds to a trade-off between false positives (type I errors) and false negatives (type II errors). Most risk measures developed for multiple testing procedures are based on type I errors. We focus on one such measure, the False Discovery Rate (FDR), which is one of the most widely used error rate in multiple testing. Denoting by $R_m$ be the total number of rejections of $\mathcal{P}_\alpha$ among $m$ hypotheses tested, and by $V_m$ the number of false rejections, the corresponding False Discovery Proportion is defined as $\mathrm{FDP}_m = V_m/(R_m \vee 1)$, and the False Discovery Rate is the expected FDP, that is:

$$\mathrm{FDR}_m = \mathbb{E}\left[\frac{V_m}{R_m \vee 1}\right].$$

A trivial way to control the FDR, or any risk measure only based on type I errors, is to make no rejection with high probability. Obviously, this is not the best strategy, as it may lead to a high number of type II errors. The performance of multiple testing procedures may be evaluated through their power, which is a function of the number of type II errors. Specifically, the power of a multiple testing procedure at level $\alpha$ is generally defined as the (random) proportion of correct rejections (true positives) among true alternative hypotheses (see, for example, Chi, 2007a):

$$\Pi_m = \frac{R_m - V_m}{(m - m_0(m)) \vee 1}.$$

**Remark 5** *All of the quantities defined in this section implicitly depend on the multiple testing procedure considered, $\mathcal{P} = (\mathcal{P}_\alpha)_{\alpha\in[0,1]}$. However, for simplicity, we will write $R_m, V_m, \mathrm{FDR}_m$, and $\Pi_m$, instead of $R_m^{\mathcal{P}_\alpha}, V_m^{\mathcal{P}_\alpha}, \mathrm{FDR}_m^{\mathcal{P}_\alpha}$, and $\Pi_m^{\mathcal{P}_\alpha}$ whenever not ambiguous.*

**Remark 6** *By definition, the power of a thresholding-based procedure is a non-decreasing function of its threshold. Therefore, among thresholding-based procedures that yield FDR less than a prescribed level, maximizing power is equivalent to maximizing the threshold of the procedure.*

### 2.3 The Benjamini-Hochberg Procedure

Suppose we wish to control the FDR at level $\alpha$. Let $p_{(1)} \leq \ldots \leq p_{(m)}$ be the ordered $p$-values, and denote by $H_{(i)}$ the null hypothesis corresponding to $p_{(i)}$. Define $\widehat{I}_m(\alpha)$ as the largest index $k \geq 0$ such that $p_{(k)} \leq \alpha k/m$. The Benjamini-Hochberg procedure at level $\alpha$ rejects all $H_{(i)}$ such that $i \leq \widehat{I}_m(\alpha)$ (if $\widehat{I}_m(\alpha) = 0$, then no rejection is made). This procedure has been proposed by Benjamini and Hochberg (1995) in the context of FDR control; Seeger (1968) reported that it had previously been used by Eklund (1961–1963) in another multiple testing context. When all true null hypotheses are independent, the BH procedure at level $\alpha$ yields *strong* FDR control, that is, it entails FDR $\leq \alpha$ regardless of the number of true null hypotheses (Benjamini and Hochberg, 1995). The BH procedure also controls the FDR when the $p$-values satisfy specific forms of positive dependence, see Benjamini and Yekutieli (2001). Figure 1 illustrates the application of the BH procedure with $\alpha = 0.2$ to $m = 100$ simulated hypotheses, among which 20 are true alternatives. The left panel illustrates the above definition of the BH procedure. An equivalent definition is that the procedure rejects all hypotheses with associated $p$-value is less than $\widehat{\tau}_m(\alpha) = \alpha \widehat{I}_m(\alpha)/m$. The

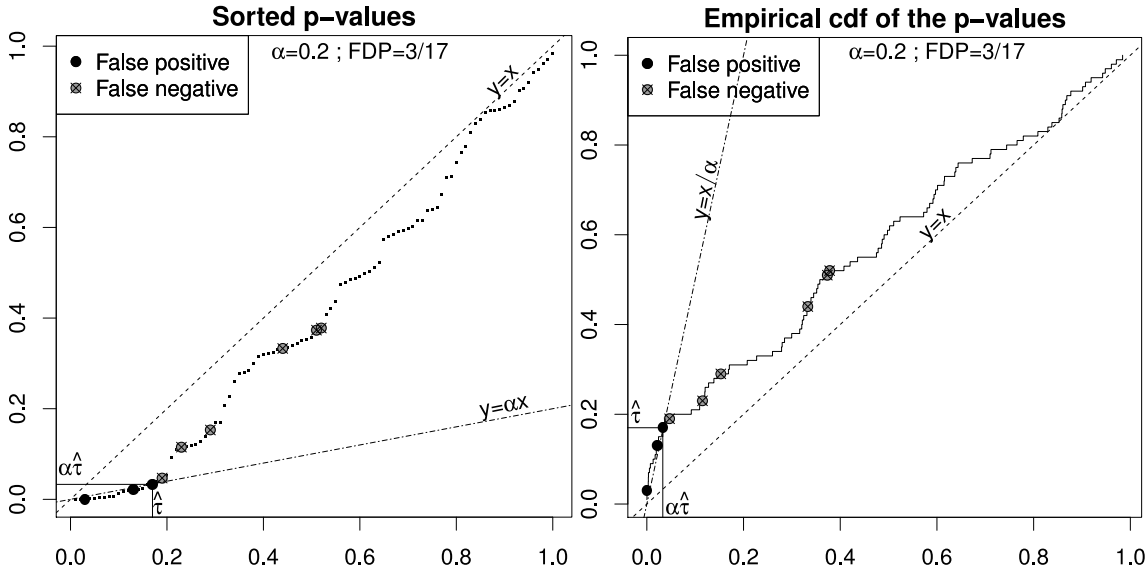



Figure 1: Illustrations of the BH procedure on a simulated example with $m = 100$. Left: sorted $p$-values: $i/m \mapsto p_{(i)}$. Right: empirical distribution function: $t \mapsto \widehat{\mathbb{G}}_m(t)$.

right panel provides a dual representation of the same information, where the $x$ and $y$ axes have been swapped. It gives a geometrical interpretation of $\widehat{\tau}_m(\alpha)$ as the largest crossing point between the line $y = x/\alpha$ and the empirical distribution function of the $p$-values, defined for $t \in [0,1]$ by $\widehat{\mathbb{G}}_m(t) = \sum_{i=1}^m \mathbf{1}_{P_i \leq t}$:

$$\widehat{\tau}_m(\alpha) = \sup\{t \in [0,1], \widehat{\mathbb{G}}_m(t) \geq t/\alpha\}.$$

### 2.4 Plug-in Procedures

In our setting where all of the hypotheses tested are independent, the BH procedure at target level $\alpha$ (henceforth denoted by BH($\alpha$) for short) in fact yields FDR control at level $\pi_0\alpha$ exactly (Benjamini

and Hochberg, 1995; Benjamini and Yekutieli, 2001). This entails that the BH($\alpha'$) procedure yields FDR $\leq \alpha$ if and only if $\alpha' \leq \alpha/\pi_0$. Therefore, as the threshold of the BH($\alpha$) procedure is a non-decreasing function of $\alpha$ and by Remark 6, the BH($\alpha/\pi_0$) procedure is optimal in our setting, in the sense that it yields maximum power among procedures of the form BH($\alpha'$) that control the FDR at level $\alpha$. As $\pi_0$ is unknown, this procedure cannot be implemented; it is generally referred to as the Oracle BH procedure.

**Remark 7** *If $\alpha \geq \pi_0$, then rejecting all null hypotheses is optimal, as it corresponds to the largest possible threshold while still maintaining* FDR $= \pi_0 \leq \alpha$. *Therefore, we will assume that $\alpha < \pi_0$ throughout the paper.*

In order to mimic the Oracle procedure, it is natural to apply the BH procedure at level $\alpha/\hat{\pi}_{0,m}$, where $\hat{\pi}_{0,m} \leq 1$ is an estimator of $\pi_0$ (Benjamini and Hochberg, 2000). Such plug-in procedures (also known as two-stage adaptive procedures) have the same geometric interpretation as the BH procedure (see Figure 1) in terms of the largest crossing point, with $\alpha/\hat{\pi}_{0,m}$ instead of $\alpha$. Their rejection threshold can be written as $\widehat{\tau}^0_m(\alpha) = \widehat{\tau}_m(\alpha/\hat{\pi}_{0,m})$, that is:

$$\widehat{\tau}^0_m(\alpha) = \sup\{t \in [0,1], \widehat{\mathbb{G}}_m(t) \geq \hat{\pi}_{0,m} t/\alpha\}\,.$$

Note that $\widehat{\tau}^0_m$ depends on the observations through both $\widehat{\mathbb{G}}_m$ and $\hat{\pi}_{0,m}$. By construction, a plug-in procedure based on an estimator $\hat{\pi}_{0,m}$ that converges in probability to $\pi_{0,\infty} \in [\pi_0, 1)$ as $m \to +\infty$ is asymptotically more powerful that the original BH procedure.

Adapting a method originally proposed by Schweder and Spjøtvoll (1982), Storey (2002) defined $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda) = \#\{i/P_i \geq \lambda\}/\#\{i \geq \lambda\}$ for $\lambda \in (0,1)$. This estimator is generally referred to as the Storey-$\lambda$ estimator. It may also be written as a function of the empirical distribution of the $p$-values:

$$\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda) = \frac{1 - \widehat{\mathbb{G}}_m(\lambda)}{1-\lambda}\,. \tag{3}$$

The rationale for $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$ is that under Assumption (2), larger $p$-values are more likely to correspond to true null hypotheses than smaller ones. Moreover, $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$ converges in probability to $(1-G(\lambda))/(1-\lambda)$, where the limit is greater than $\pi_0$ as $G$ stochastically dominates the uniform distribution. Several choices of $\lambda$ have been proposed, including $\lambda = 1/2$ (Storey and Tibshirani, 2003), a data-driven choice based on the bootstrap (Storey et al., 2004), and $\lambda = \alpha$ (Blanchard and Roquain, 2009). In our setting, a slightly modified version of the corresponding plug-in BH($\alpha/\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$) procedure where $1/m$ is added to the numerator in (3) achieves strong FDR control at level $\alpha$ (Storey et al., 2004). We note that the Storey-$\lambda$ estimator $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$ can be viewed as a kernel estimator of the density $g$ at 1.

**Definition 8 (Kernel of order $\ell$ and kernel estimator of a density at a point)**

1. *A kernel of order $\ell \in \mathbb{N}$ is a function $K : \mathbb{R} \to \mathbb{R}$ such that the functions $u \mapsto u^j K(u)$ are integrable for any $j = 0 \ldots \ell$, and satisfy $\int_{\mathbb{R}} K = 1$, and $\int_{\mathbb{R}} u^j K(u) du = 0$ for $j = 1 \ldots \ell$.*

2. *The kernel estimator of a density $g$ at $x_0$ based on $m$ independent, identically distributed observations $x_1, \ldots x_m$ from $g$ is defined by*

$$\hat{g}_m(x_0) = \frac{1}{mh} \sum_{i=1}^{m} K\left(\frac{x_i - x_0}{h}\right),$$

*where $h > 0$ is called the bandwidth of the estimator, and $K$ is a kernel.*

By Definition 8, $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$ is a kernel estimator of the density $g$ at 1 with kernel $K^{\mathrm{Sto}}(t) = \mathbf{1}_{[-1,0]}(t)$ and bandwidth $h = 1-\lambda$. $K^{\mathrm{Sto}}$ is an asymmetric, rectangular kernel of order 0.

### 2.5 Criticality and Asymptotic Properties of FDR Controlling Procedures

Upper bounds on the asymptotic number of rejections of FDR controlling procedures have been identified and characterized by Chi (2007a) and Chi and Tan (2008), who introduced the notion of *critical value of a multiple testing problem* and that of *critical value of a multiple testing procedure*. Both notions are defined formally below. They are tightly connected, with the important difference that the former only depends on the multiple testing problem, while the latter depends on both the multiple testing problem and a specific multiple testing procedure.

**Definition 9 (Critical value of a multiple testing problem)** *The critical value of the multiple testing problem parametrized by $\pi_0$ and $G$ is defined by*

$$\underline{\alpha}^\star = \inf_{t\in(0,1]} \frac{\pi_0 t}{G(t)}. \tag{4}$$

Chi and Tan (2008, proof of Proposition 3.2) proved that for any multiple testing procedure, for $\alpha < \underline{\alpha}^\star$, there exists a positive constant $c(\alpha)$ such that almost surely, for $m$ large enough, the events $\{V_m/R_m \le \alpha\}$ and $\{R_m \ge c(\alpha)\log m\}$ are incompatible. This restriction is intrinsic to the multiple testing problem, in the sense that it holds regardless of the considered multiple testing procedure. Obviously, this is not a limitation when $\underline{\alpha}^\star = 0$. We introduce the following Condition:

$$\underline{\alpha}^\star > 0. \tag{5}$$

Whether Condition (5) is satisfied or not only depends on $G$. However, the value of $\underline{\alpha}^\star$ as defined in (4) depends on both $\pi_0$ and $G$. Under Assumption (2) we have $\underline{\alpha}^\star = \lim_{t\to 0}\pi_0 t/G(t) = \pi_0/(\pi_0 + (1-\pi_0)g_1(0))$, where $g_1(0) \in [0,+\infty]$ is defined by $g_1(0) = \lim_{t\to 0} g_1(t)$. By Proposition 1, $g_1(0)$ only depends on the behavior of the test statistics distribution. In particular, under Assumption (2), Condition (5) is satisfied if and only if the likelihood ratio $f_1/f_0$ is bounded near $+\infty$.

We now introduce the notion of *critical value of a multiple testing procedure*. Chi (2007a) defined the critical value of the BH procedure as $\alpha^\star_{BH} = \inf_{t\in(0,1]} t/G(t)$. Let us denote by

$$\tau_\infty(\alpha) = \sup\{t \in [0,1], G(t) \ge t/\alpha\} \tag{6}$$

the rightmost crossing point between $G$ and the line $y = x/\alpha$. Chi (2007a) has proved the following result:

**Proposition 10 (Asymptotic properties of the BH procedure)** *For $\alpha \in [0,1]$, let $\widehat{\tau}_m(\alpha)$ be the threshold of the BH($\alpha$) procedure, and let $\tau_\infty(\alpha)$ be defined by* (6)*. Let $\alpha^\star_{BH} = \inf_{t\in(0,1]} t/G(t)$. As $m \to +\infty$,*

1. *If $\alpha < \alpha^\star_{BH}$, then $\widehat{\tau}_m(\alpha) \xrightarrow{a.s.} 0$;*
2. *If $\alpha > \alpha^\star_{BH}$, then $\widehat{\tau}_m(\alpha) \xrightarrow{a.s.} \tau_\infty(\alpha)$, where the limit is positive.*

A straightforward consequence of Proposition 10 is that the BH($\alpha$) procedure has asymptotically null power when $\alpha < \alpha^\star_{BH}$ and positive power when $\alpha > \alpha^\star_{BH}$. The following Definition generalizes the notion of critical value of to a generic multiple testing procedure:

**Definition 11 (Critical value of a multiple testing procedure)** *Let $\mathcal{P}$ denote a multiple testing procedure. The critical value of $\mathcal{P}$ is defined by*

$$\alpha^\star_{\mathcal{P}} = \sup \left\{ \alpha \in [0,1], \Pi_m^{\mathcal{P}_\alpha} \xrightarrow[m\to+\infty]{a.s.} 0 \right\}.$$

The critical value $\alpha^\star_{\mathcal{P}}$ depends on both the procedure $\mathcal{P}$, and the multiple setting. For the BH procedure, criticality ($\alpha < \alpha^\star_{BH}$) corresponds to situations where the target FDR level $\alpha$ is so small that there is no positive crossing point between $G$ and the line $y = x/\alpha$. Conversely, when $\alpha > \alpha^\star_{BH}$, there is a positive crossing point between $G$ and the line $y = x/\alpha$, as illustrated by Figure 1 (right). The almost sure convergence results of Proposition 10 in the case $\alpha > \alpha^\star_{BH}$ were extended by Neuvial (2008), in the conditional setting. Specifically, the threshold $\widehat{\tau}_m(\alpha)$ of the BH procedure was shown to converge in distribution to $\tau_\infty(\alpha)$ at rate $m^{-1/2}$ as soon as $\alpha > \alpha^\star_{BH}$. Neuvial (2008) also proved that similar central limit theorems hold for a class of thresholding-based FDR controlling procedures that covers some plug-in procedures, including the Storey-$\lambda$ procedure: the threshold of a procedure $\mathcal{P}$ of this class converges in distribution to a procedure-specific, positive value at rate $m^{-1/2}$ as soon as $\alpha > \alpha^\star_{\mathcal{P}}$.

**Remark 12 (Criticality of a multiple testing problem versus criticality of a procedure)** *Whether Condition* (5) *hols or not only depends on the behavior of the test statistics distribution. However, this condition is tightly connected to the critical value of FDR controlling procedures. In order to shed some light on this connection, we note that $\underline{\alpha}^\star = \pi_0 \alpha^\star_{BH}$ may be interpreted as the critical value of the Oracle BH procedure BH$(\alpha/\pi_0)$. Therefore, as the Oracle BH procedure at level $\alpha$ is the most powerful procedure among thresholding-based procedures that control FDR at level $\alpha$, $\underline{\alpha}^\star$ is a lower bound on the critical values of these procedures. Specifically, multiple problems for which Condition* (5) *is satisfied or not differ in that:*

- *when Condition* (5) *is satisfied, all thresholding-based procedures that control FDR have null asymptotic power in a range of levels containing $[0, \underline{\alpha}^\star)$;*
- *when Condition* (5) *is not satisfied, some procedures (including BH) have positive asymptotic power for any positive level $\alpha$.*

This paper extends the asymptotic results of Chi (2007a) and Neuvial (2008) to the case of plug-in procedures of the form BH$(\alpha/\hat{\pi}_{0,m})$, where $\hat{\pi}_{0,m}$ is a kernel estimator of the $p$-value distribution $g$ at 1. Specifically, we consider a class of kernel estimators of $\pi_0$, which includes a modification of the Storey-$\lambda$ estimator, where the parameter $\lambda$ tends to 1 as $m \to \infty$. In Section 3, we prove that this class of estimators of $\pi_0$ achieves non-parametric convergence rates of the form $m^{-k/(2k+1)}/\eta_m$, where $\eta_m$ goes to 0 slowly enough as $m \to +\infty$, and $k$ controls the regularity of $g$ at 1. In Section 4, we characterize the critical value $\alpha^\star_0$ of plug-in procedures based on such estimators, and prove that when the target FDR level $\alpha$ is greater than $\alpha^\star_0$, the convergence rate of these plug-in procedures is $m^{-k/(2k+1)}/\eta_m$, which is slower than the parametric rate achieved by the BH procedure and by the plug-in procedures studied in Neuvial (2008). In Section 5, these results are applied to one and two-sided tests in location and Student models. Practical consequences and possible extensions of this work are discussed in Section 6.

## 3. Asymptotic Properties of Non-Parametric Estimators of $\pi_0$

Let $\lambda \in (0,1)$. The expectation $\overline{\pi_0}(\lambda)$ of the Storey-$\lambda$ estimator is given by

$$\overline{\pi_0}(\lambda) = \pi_0 + (1-\pi_0)\frac{1-G_1(\lambda)}{1-\lambda}. \tag{7}$$

Moreover, as a regular function of the empirical distribution of the $p$-values, $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$ has the following asymptotic distribution for $\lambda \in (0,1)$ (Genovese and Wasserman, 2004):

$$\sqrt{m}\left(\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda) - \overline{\pi_0}(\lambda)\right) \rightsquigarrow \mathcal{N}\left(0, \frac{G(\lambda)(1-G(\lambda))}{(1-\lambda)^2}\right).$$

In our setting, $g_1$ is positive, as noted in Section 2.1. Therefore, we have $G_1(\lambda) < 1$ for any $\lambda \in (0,1)$, and the bias $\overline{\pi_0}(\lambda) - \pi_0$ is positive: the Storey-$\lambda$ estimator achieves a parametric convergence rate, but it is not a consistent estimator of $\pi_0$. Under Assumption (2), this bias decreases as $\lambda$ increases (by Equation (7)). In order to mimic the Oracle BH$(\alpha/\pi_0)$ procedure, it is therefore natural to choose $\lambda$ close to 1. We consider plug-in procedures where $\pi_0$ is estimated by $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$, with $h_m \to 0$ as $m \to +\infty$. As the limit in probability of this estimator is $g(1) = \pi_0 + (1-\pi_0)g_1(1)$, it is consistent if and only if the following "purity" condition, which has been introduced by Genovese and Wasserman (2004), is met:

$$g_1(1) = 0 \tag{8}$$

We note that the Storey-$\lambda$ estimator is not a consistent estimator of $\pi_0$ even in when Condition (8) is met. Moreover, Condition (8) is entirely determined by the shape of the test statistics under the alternative hypothesis. The asymptotic bias and variance of $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$ are characterized by Proposition 13:

**Proposition 13 (Asymptotic bias and variance of $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$)** *Let $(h_m)_{m\in\mathbb{N}}$ be a positive sequence such that $h_m \to 0$.*

1. *If $mh_m \to +\infty$ as $m \to +\infty$, then*

$$\sqrt{mh_m}\left(\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m) - \mathbb{E}\left[\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)\right]\right) \rightsquigarrow \mathcal{N}(0, g(1)).$$

2. *Assume that for $k \geq 1$, $g$ is $k$ times differentiable at 1, with $g^{(l)}(1) = 0$ for $1 \leq l < k$. Then*

$$\mathbb{E}\left[\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)\right] - g(1) \underset{m\to+\infty}{=} \frac{(-1)^k g^{(k)}(1)}{(k+1)!}h_m^k + \mathrm{o}\left(h_m^k\right).$$

Only the bias term in Proposition 13 depends on the regularity $k$ of the distribution near 1: the asymptotic bias is of order $h_m^k$, while the asymptotic variance of $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$ is of order $(mh_m)^{-1}$, regardless of the regularity of the distribution. The bandwidth $h_m$ in Proposition 13 realizes a trade-off between the asymptotic bias and variance of $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$. When the regularity of the distribution is known, a natural way to resolve this bias/variance trade-off is to calibrate $h_m$ such that the Mean Squared Error (MSE) of the corresponding estimator is asymptotically minimum. This gives rise to an optimal choice of the bandwidth, which is characterized by the following proposition:

**Proposition 14 (Asymptotic properties of $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$)** *Assume that $g$ is $k$ times differentiable at 1 for $k \geq 1$, with $g^{(l)}(1) = 0$ for $1 \leq l < k$.*

1. *If $g^{(k)}(1) \neq 0$, then the asymptotically optimal bandwidth for $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$ in terms of MSE is of order $m^{-1/(2k+1)}$, and the corresponding MSE is of order $m^{-2k/(2k+1)}$.*

2. *Let $\eta_m$ be any sequence such that $\eta_m \to 0$ and $m^{k/(2k+1)}\eta_m \to +\infty$ as $m \to +\infty$. Then, letting $h_m(k) = m^{-1/(2k+1)}\eta_m^2$, we have, as $m \to +\infty$:*

$$m^{k/(2k+1)}\eta_m \left(\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m(k)) - g(1)\right) \rightsquigarrow \mathcal{N}(0, g(1)). \tag{9}$$

Proposition 14 is proved in Appendix B. The convergence rate in (9) is a typical convergence rate for non-parametric estimators of a density at a point. However, Proposition 14 cannot be derived from classical results on kernel estimators such as those obtained in Tsybakov (2009). Indeed, such results typically require that the order of the kernel matches the regularity $k$ of the density, whereas the kernel of Storey's estimator, $K^{\mathrm{Sto}}(t) = \mathbf{1}_{[-1,0]}(t)$, is of order 0. The results that can be obtained with kernels of order $k$ are summarized by Proposition 15; we refer to Tsybakov (2009) for a proof of this result.

**Proposition 15 ($k^{\text{th}}$ order kernel estimator)** *Assume that for $k \geq 1$, $g$ is $k$ times differentiable at 1. Let $\hat{g}^k_m(1)$ be a kernel estimator of $g(1)$ with bandwidth $h_m$, associated with a $k^{\text{th}}$ order kernel.*

1. *The optimal bandwidth for $\hat{g}^k_m(1)$ in terms of MSE is of order $m^{-1/(2k+1)}$, and the corresponding MSE is of order $m^{-2k/(2k+1)}$;*

2. *Let $\eta_m$ be any sequence such that $\eta_m \to 0$ and $m^{k/(2k+1)}\eta_m \to +\infty$ as $m \to +\infty$. Then letting $h_m(k) = m^{-1/(2k+1)}\eta_m^2$, we have, as $m \to +\infty$:*

$$m^{k/(2k+1)}\eta_m \left(\hat{g}^k_m(1) - g(1)\right) \rightsquigarrow \mathcal{N}(0, g(1)).$$

Propositions 14 and 15 show that the convergence rate of kernel estimators of $g(1)$ with asymptotically optimal bandwidth directly depends on the regularity $k$ of $g$ at 1. The only difference between the two propositions is that the assumption that the first $k-1$ derivatives of $g$ are null at 1 for $\hat{\pi}_{0,m}(1-h_m)$ is not needed for $k^{\text{th}}$ order kernel estimators. Importantly, these convergence rates cannot be improved in our setting, in the sense that $m^{-k/(2k+1)}$ is the minimax rate for the estimation of a density at a point where its regularity is of order $k$ (Tsybakov, 2009, Chapter 2).

To the best of our knowledge, the only non-parametric estimators of $\pi_0$ for which convergence rates have been established in our setting are those proposed by Storey (2002), Swanepoel (1999) and Hengartner and Stark (1995). We now briefly review asymptotic properties of these estimators in the context of multiple testing, as stated in Genovese and Wasserman (2004), and show that their convergence rates can essentially be recovered by Propositions 14 and 15.

**Confidence envelopes for the density:** Hengartner and Stark (1995) derived a finite sample confidence envelope for a monotone density. Assuming that $G$ is concave and that $g$ is Lipschitz in a neighborhood of 1, Genovese and Wasserman (2004) obtained an estimator which converged to $g(1)$ at rate $(\ln m)^{1/3}m^{-1/3}$. The same rate of convergence can be achieved by Proposition 14 or 15 (for $\eta_m = (\ln m)^{-1/3}$) if we assume that $g$ is differentiable at 1. This is a slightly stronger assumption than the ones made by Hengartner and Stark (1995), but it still corresponds to a regularity of order 1.

**Spacings-based estimator:** Swanepoel (1999) proposed a two-step estimator of the minimum of an unknown density based on the distribution of the spacings between observations: first, the location of the minimum is estimated, and then the density at this point is itself estimated. Assuming that at the value at which the density $g$ achieves its minimum, $g$ and $g^{(1)}$ are null, and $g^{(2)}$ is bounded away from 0 and $+\infty$ and Lipschitz, then for any $\delta > 0$, there exists an estimator converging at rate $(\ln m)^{\delta} m^{-2/5}$ to the true minimum. The same rate of convergence can be achieved by Proposition 14 or 15 (for $\eta_m = (\ln m)^{-\delta}$) if one assumes that $g$ is twice differentiable at 1 (and additionally that $g^{(1)}(1) = 0$ for Proposition 14). In our setting, the Lipschitz condition for the second derivative is unnecessary: the minimum of $g$ is necessarily achieved at 1 because $g$ is non-increasing (under Assumption (2)), so the first step of the estimation in Swanepoel (1999) may be omitted.

As both estimators are estimators of $g(1)$, the differences in their asymptotic properties are driven by the differences in the regularity assumptions made for $g$ (or $g_1$) near 1, rather than by their specific form.

## 4. Consistency, Criticality and Convergence Rates of Plug-in Procedures

The aim of this section is to derive convergence rates for plug-in procedures based on the estimators $\hat{\pi}_{0,m}$ of $\pi_0$ studied in Section 3. Specifically, our goal is to establish central limit theorems for the threshold $\widehat{\tau}^0_m(\alpha)$ of the plug-in procedure $\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ and the associated False Discovery Proportion, which we denote by $\mathrm{FDP}_m(\widehat{\tau}^0_m(\alpha))$. The convergence results obtained by Neuvial (2008) cover a broad class of FDR controlling procedures, including the BH procedure and plug-in procedures based on estimators of $\pi_0$ that depend on the observations only through the empirical distribution function $\widehat{\mathbb{G}}_m$ of the $p$-values (Storey, 2002; Storey et al., 2004; Benjamini et al., 2006). Although these results were obtained in the conditional setting of Benjamini and Hochberg (1995), extending them to the unconditional setting considered here is relatively straightforward, because the proof techniques developed in Neuvial (2008) can be adapted to this setting. For completeness, the asymptotic properties of the BH procedure and the plug-in procedure based on the Storey-$\lambda$ estimator are derived in Appendix C. The problem considered in this section is more challenging, as the kernel estimators introduced in Section 3 depend on $m$ not only through $\widehat{\mathbb{G}}_m$, but also through the bandwidth of the kernel (for example, $h_m$ for $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$).

Let $\hat{\pi}_{0,m}$ denote a generic estimator of $\pi_0$. We assume that $\hat{\pi}_{0,m}$ converges in probability to $\pi_{0,\infty} \leq 1$ as $m \to +\infty$. We do not assume that $\pi_{0,\infty} = \pi_0$. Therefore, $\hat{\pi}_{0,m}$ may or may not be a consistent estimator of $\pi_0$. We recall that the $\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ procedure rejects all hypotheses with $p$-values smaller than

$$\widehat{\tau}^0_m(\alpha) = \sup\left\{ t \in [0,1], \widehat{\mathbb{G}}_m(t) \geq \hat{\pi}_{0,m} t/\alpha \right\}.$$

We now study the behavior of the $\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ procedure when $\hat{\pi}_{0,m}$ converges at a rate $r_m$ slower than the parametric rate $m^{-1/2}$, that is, $m^{-1/2} = \mathrm{o}(r_m)$. We define the asymptotic threshold $\tau^0_\infty(\alpha)$ corresponding to $\widehat{\tau}^0_m(\alpha)$ as

$$\tau^0_\infty(\alpha) = \sup\{ t \in [0,1], G(t) \geq \pi_{0,\infty} t/\alpha \}.$$

We have $\tau^0_\infty(\alpha) = \tau_\infty(\alpha/\pi_{0,\infty})$, that is, the asymptotic threshold of the BH procedure defined in Equation (6) at level $\alpha/\pi_{0,\infty}$.

**Theorem 16 (Asymptotic properties of plug-in procedures)** *Let $\hat{\pi}_{0,m}$ be an estimator of $\pi_0$ such that $\hat{\pi}_{0,m} \to \pi_{0,\infty}$ in probability as $m \to +\infty$. Let $\alpha_0^\star = \pi_{0,\infty}\alpha_{BH}^\star$. Then:*

1. *$\alpha_0^\star$ is the critical value of the BH$(\alpha/\hat{\pi}_{0,m})$ procedure;*
2. *Further assume that the asymptotic distribution of $\hat{\pi}_{0,m}$ is given by*

$$\sqrt{mh_m}\,(\hat{\pi}_{0,m} - \pi_{0,\infty}) \rightsquigarrow \mathcal{N}(0, s_0^2)$$

*for some $s_0$, with $h_m = o(1/\ln\ln m)$ and $mh_m \to +\infty$ as $m \to +\infty$. Then, under Assumption (2), for any $\alpha > \alpha_0^\star$,*

   (a) *The asymptotic distribution of the threshold $\widehat{\tau}_m^0(\alpha)$ is given by*

$$\sqrt{mh_m}\left(\widehat{\tau}_m^0(\alpha) - \tau_\infty^0(\alpha)\right) \rightsquigarrow \mathcal{N}\left(0, \left(\frac{s_0\tau_\infty^0(\alpha)/\alpha}{\pi_{0,\infty}/\alpha - g(\tau_\infty^0(\alpha))}\right)^2\right);$$

   (b) *The asymptotic distribution of the FDP achieved by the BH$(\alpha/\hat{\pi}_{0,m})$ procedure is given by*

$$\sqrt{mh_m}\left(\mathrm{FDP}_m(\widehat{\tau}_m^0(\alpha)) - \frac{\pi_0\alpha}{\pi_{0,\infty}}\right) \rightsquigarrow \mathcal{N}\left(0, \left(\frac{\pi_0\alpha s_0}{\pi_{0,\infty}^2}\right)^2\right).$$

Theorem 16 states that for $\alpha > \alpha_0^\star$, for any estimator $\hat{\pi}_{0,m}$ that converges in distribution at a rate $r_m$ slower than the parametric rate $m^{-1/2}$, the plug-in procedure BH$(\alpha/\hat{\pi}_{0,m})$ converges at rate $r_m$ as well. This is a consequence of the fact that $r_m$ dominates the fluctuations of $\widehat{\mathbb{G}}_m$, which are of parametric order.

We now state the main result of the paper (Corollary 17), that is, the asymptotic properties of plug-in procedures associated with the estimators of $\pi_0$ studied in Section 3, for which $s_0^2 = g(1)$. This result can be derived by combining the results of Theorem 16 with those of Propositions 14 and 15.

**Corollary 17** *Assume that (2) holds, and that $g$ is $k$ times differentiable at 1 for $k \geq 1$. Define $h_m(k) = m^{-1/(2k+1)}\eta_m^2$, where $\eta_m \to 0$ and $m^{k/(2k+1)}\eta_m \to +\infty$ as $m \to +\infty$. Denote by $\hat{\pi}_{0,m}^k$ one of the following two estimators of $\pi_0$:*

- *Storey's estimator $\hat{\pi}_{0,m}^{\mathrm{Sto}}(1 - h_m(k))$; in this case, it is further assumed that $g^{(l)}(1) = 0$ for $1 \leq l < k$;*
- *A kernel estimator of $g(1)$ associated with a $k^{\text{th}}$ order kernel with bandwidth $h_m(k)$.*

*Then*

1. *$\alpha_0^\star = g(1)\alpha_{BH}^\star$ is the critical value of the BH$(\alpha/\hat{\pi}_{0,m}^k)$ procedure;*
2. *For any $\alpha > \alpha_0^\star$,*

   (a) *The asymptotic distribution of the threshold $\widehat{\tau}_m^0(\alpha)$ is given by*

$$m^{k/(2k+1)}\eta_m\left(\widehat{\tau}_m^0(\alpha) - \tau_\infty^0(\alpha)\right) \rightsquigarrow \mathcal{N}\left(0, \left(\frac{\tau_\infty^0(\alpha)/\alpha}{g(1)/\alpha - g(\tau_\infty^0(\alpha))}\right)^2 g(1)\right);$$

*(b) The asymptotic distribution of the FDP achieved by the BH$(\alpha/\hat{\pi}_{0,m})$ procedure is given by*

$$m^{k/(2k+1)}\eta_m\left(\mathrm{FDP}_m(\widehat{\tau}^0_m(\alpha)) - \frac{\pi_0\alpha}{g(1)}\right) \rightsquigarrow \mathcal{N}\left(0, \frac{\pi_0^2\alpha^2}{g(1)^3}\right).$$

We note that unlike the modification of the Storey-$\lambda$ estimator studied here, the estimators of $\pi_0$ based on kernels of order $k$ do not require the first $k-1$ derivatives of $g$ at 1 to be null. Therefore, the latter are generally preferable to the former. Corollary 17 has the following consequences, which are also summarized in Table 1:

- Assume that Condition (8) is met. Then the asymptotic threshold of the BH$(\alpha/\hat{\pi}_{0,m})$ procedure is $\tau_\infty(\alpha/\pi_0)$, that is, the asymptotic threshold of the Oracle procedure BH$(\alpha/\pi_0)$. In particular, the asymptotic FDP achieved by the estimators in Corollary 17 is then *exactly* $\alpha$ (and its asymptotic variance is $\alpha^2/\pi_0$), whereas the asymptotic FDP of the original BH procedure is $\pi_0\alpha$.

- We have
$$\underline{\alpha}^\star \leq \alpha_0^\star \leq \alpha^\star_{\mathrm{Sto}(\lambda)} \leq \alpha^\star_{BH}.$$
In models where Condition (5) is not satisfied, all the critical values in (4) are null, implying that all the corresponding procedures have positive power for any target FDR level. In models where Condition (5) is satisfied, all the critical values in (4) are positive, and (4) implies that the range of target FDR values $\alpha$ that yield asymptotically positive power is larger for the plug-in procedures studied in this paper than for the BH procedure or the Storey-$\lambda$ procedure.

- We have $\tau^0_\infty(\alpha) \geq \tau^{0,\lambda}_\infty(\alpha) \geq \tau_\infty(\alpha)$, where $\tau^{0,\lambda}_\infty(\alpha)$ denotes the asymptotic threshold of the Storey-$\lambda$ procedure, which is formally defined and characterized in Appendix C. Therefore, as the power of a thresholding-based FDR controlling procedure is a non-decreasing function of its threshold (Remark 6), the asymptotic power of the BH$(\alpha/\hat{\pi}_{0,m})$ procedure is greater than that of both the Storey-$\lambda$ and the original BH procedures, even in the range $\alpha > \alpha^\star_{BH}$ where all of them have positive asymptotic power.

| Name | $\hat{\pi}_{0,m}$ | FDR $/\alpha$ | Rate | (Asy. var. of FDP)/ FDR |
|---|---|---|---|---|
| BH | 1 | $\pi_0$ | $m^{-1/2}$ | $(\pi_0\tau_\infty(\alpha))^{-1} - 1$ |
| Oracle BH | $\pi_0$ | 1 | $m^{-1/2}$ | $(\tau_\infty(\alpha/\pi_0))^{-1} - 1$ |
| Storey-$\lambda$ | $\hat{\pi}^{\mathrm{Sto}}_{0,m}(\lambda)$ | $\pi_0/\overline{\pi_0}(\lambda)$ | $m^{-1/2}$ | $(\pi_0\tau^{0,\lambda}_\infty(\alpha))^{-1} + (1-G(\lambda))^{-1}$ |
| Kernel($h_m(k)$) | $\hat{\pi}^k_{0,m}$ | $\pi_0/g(1)$ | $m^{-k/(2k+1)}$ | $g(1)^{-1}$ |

Table 1: Summary of the asymptotic properties of the FDR controlling procedures considered in this paper, for a target FDR level $\alpha$ greater than the (procedure-specific) critical value. Note that "Storey-$\lambda$" denotes the original procedure with a fixed $\lambda$, while our extension with $\lambda = 1 - h_m(k)$ is categorized in the table as a particular case of kernel estimator (last row). For Storey-$\lambda$, we also assume that $\lambda > \tau^{0,\lambda}_\infty(\alpha)$.

These results characterize the increase in asymptotic power achieved by plug-in procedures based on kernel estimators of $\pi_0$. However, this increased asymptotic power comes at the price of a slower

convergence rate. Specifically, the convergence rate of plug-in procedures is the non-parametric rate $m^{-k/(2k+1)}/\eta_m$ (where $k$ controls the regularity of $g$) for the $\mathrm{BH}(\alpha/\hat{\pi}^k_{0,m})$ procedure, while the parametric rate $m^{-1/2}$ was achieved by the original BH procedure, the Oracle BH procedure, and the Storey-$\lambda$ procedure (as proved in Appendix C).

## 5. Application to Location and Student Models

In Section 4 we proved that the asymptotic behavior of plug-in procedures depends on whether the target FDR level $\alpha$ is above or below the critical value $\alpha^\star_0$ characterized by Theorem 16, and by establishing convergence rates for these procedures when $\alpha > \alpha^\star_0$. Both the critical value $\alpha^\star_0$ and the obtained convergence rates depend on the test statistics distribution. In the present section, these results are applied to Gaussian and Laplace location models, and to the Student model. We begin by defining these models (Section 5.1) and studying criticality in each of them (Section 5.2). Then, we derive convergence rates for plug-in procedures based on the kernel estimators of $\pi_0$ considered in Sections 3 and 4, both for two-sided tests (Section 5.3) and one-sided tests (Section 5.4).

### 5.1 Models for the Test Statistics

In this subsection, we define the location and Student models used throughout Section 5.

#### 5.1.1 Location Models

In location models the distribution of the test statistic under $\mathcal{H}_1$ is a shift from that of the test statistic under $\mathcal{H}_0$: $F_1 = F_0(\cdot - \theta)$ for some location parameter $\theta > 0$. The most widely studied location models are the Gaussian and Laplace (double exponential) location models. Both the Gaussian and the Laplace distribution can be viewed as instances of a more general class of distributions introduced by Subbotin (1923) and given for $\gamma \geq 1$ by

$$f^\gamma_0(x) = \frac{1}{C_\gamma} e^{-|x|^\gamma/\gamma}, \text{ with } C_\gamma = \int_{-\infty}^{+\infty} e^{-|x|^\gamma/\gamma} dx = 2\Gamma(1/\gamma)\gamma^{1/\gamma - 1}.$$

Therefore, the likelihood ratio in the $\gamma$-Subbotin location model may be written as

$$\frac{f^\gamma_1}{f^\gamma_0}(x) = \exp\left(\frac{|x|^\gamma}{\gamma} - \frac{|x-\theta|^\gamma}{\gamma}\right). \tag{10}$$

The Gaussian case corresponds to $\gamma = 2$ and the Laplace case to $\gamma = 1$. In the Laplace case, the distribution of the $p$-values under the alternative can be derived explicitly, see Lemma 22 in Appendix. We focus on $1 \leq \gamma \leq 2$ as this corresponds to situations in which Assumption (2) is satisfied. Specifically, for one-sided tests, Assumption (2) holds as soon as $\gamma \geq 1$, because then $f^\gamma_1/f^\gamma_0$ is non-decreasing; for two-sided tests, if additionally $\gamma \leq 2$, then Assumption (2) holds (as proved in Appendix A, Proposition 23).

#### 5.1.2 Student Model

Student's $t$ distribution is widely used in applications, as it naturally arises when testing equality of means of Gaussian random variables with unknown variance. In the Student model with parameter $\nu > 0$, $F_0$ is the (central) $t$ distribution with $\nu$ degrees of freedom, and $F_1$ is the non-central $t$ distribution with $\nu$ degrees of freedom and non-centrality parameter $\theta > 0$. The Student model is not

a location model, as $F_1$ cannot be written as a translation of $F_0$. Following Chi (2007a, Equation (3.5)), we note that the likelihood ratio of the Student model may be written as

$$\frac{f_1}{f_0}(t) = \sum_{j=0}^{+\infty} a_j(\nu,\theta)\psi_{(j,\nu)}(t)\,, \tag{11}$$

where $\psi_{(j,\nu)}(t) = (t/\sqrt{t^2+\nu})^j = \mathrm{sgn}(t)^j\left(1+\nu/t^2\right)^{-j/2}$ for $t \in \mathbb{R}$ and

$$a_j(\nu,\theta) = e^{-\theta^2/2}\frac{\Gamma((\nu+j+1)/2)}{\Gamma((\nu+1)/2)}\frac{(\sqrt{2}\theta)^j}{j!}\,.$$

**Remark 18** *The sequence $a_j(\nu,\theta)$ is positive, and it is not hard to see that $(\sum_j a_j(\nu,\theta))$ is a convergent series using Stirling's formula. Therefore, as $\psi_{(j,\nu)}(t) \in [-1,1]$, the dominated convergence theorem ensures that the right-hand side of Equation* (11) *is well-defined for any $t \in \mathbb{R}$.*

Another useful expression for the Student likelihood ratio may be derived from the integral expression of the density of a non-central $t$ distribution given by Johnson and Welch (1940):

$$\frac{f_1}{f_0}(t) = \exp\left[-\frac{\theta^2}{2}\,\frac{1}{1+\frac{t^2}{\nu}}\right]\frac{Hh_\nu\left(-\frac{\theta t}{\sqrt{\nu+t^2}}\right)}{Hh_\nu(0)}\,, \tag{12}$$

where $Hh_\nu(z) = \int_0^{+\infty}\frac{u^\nu}{\nu!}e^{-\frac{1}{2}(u+z)^2}dx$. As noted by Chi (2007a, Section 3.1), the likelihood ratio of Student test statistics is non-decreasing, which implies that Assumption (2) holds for one-sided tests. It also holds for two-sided tests, as proved in Appendix A, Proposition 26.

The location models and the Student model considered here are parametrized by two parameters: (i) a non-centrality parameter $\theta$, which encodes a notion of distance between $\mathcal{H}_0$ and $\mathcal{H}_1$; (ii) a parameter which controls the (common) tails of the distribution under $\mathcal{H}_0$ and $\mathcal{H}_1$: $\gamma$ for the $\gamma$-Subbotin model, and $\nu$ for the Student model with $\nu$ degrees of freedom.

### 5.2 Criticality

As the asymptotic behavior of plug-in procedures crucially depends on whether the target FDR level is above or below the critical value $\alpha_0^\star$ characterized by Theorem 16, it is of primary importance to study criticality in the models we are interested in. Noting that $\alpha_0^\star = \pi_{0,\infty}\alpha_{BH}^\star = \pi_{0,\infty}\underline{\alpha}^\star/\pi_0$, we have $\alpha_0^\star > 0$ if and only if Condition (5) is satisfied, that is, if and only if the likelihood ratio $f_1/f_0$ is bounded near $+\infty$. In this section, we study Condition (5) in location and Student models.

#### 5.2.1 LOCATION MODELS

In location models, where $f_1 = f_0(\cdot-\theta)$ with $\theta > 0$, the behavior of the likelihood ratio is closely related to the tail behavior of the distribution of the test statistics: for a given non-centrality parameter $\theta$, the heavier the tails, the smaller the difference between $f_1$ and $f_0$. In a $\gamma$-Subbotin location model, Equation (10) yields $|1-\theta/x|^\gamma \sim 1-\gamma\theta/x$ as $x \to +\infty$. Thus $|x|^\gamma\left(1-|1-\theta/x|^\gamma\right) \sim \gamma\theta x^{\gamma-1}$, and the behavior of the likelihood ratio $f_1^\gamma/f_0^\gamma$ is driven by the value of $\gamma$, as illustrated by Figure 2 for the Gaussian and Laplace location models with location parameter $\theta \in \{1,2\}$.

If $\gamma > 1$, then $\lim_{+\infty} f_1^\gamma/f_0^\gamma = +\infty$. Therefore, the slope of the cumulative distribution function of the $p$-values is infinite at 0, and Condition (5) is not satisfied for the Subbotin model: $\underline{\alpha}^\star = 0$ for

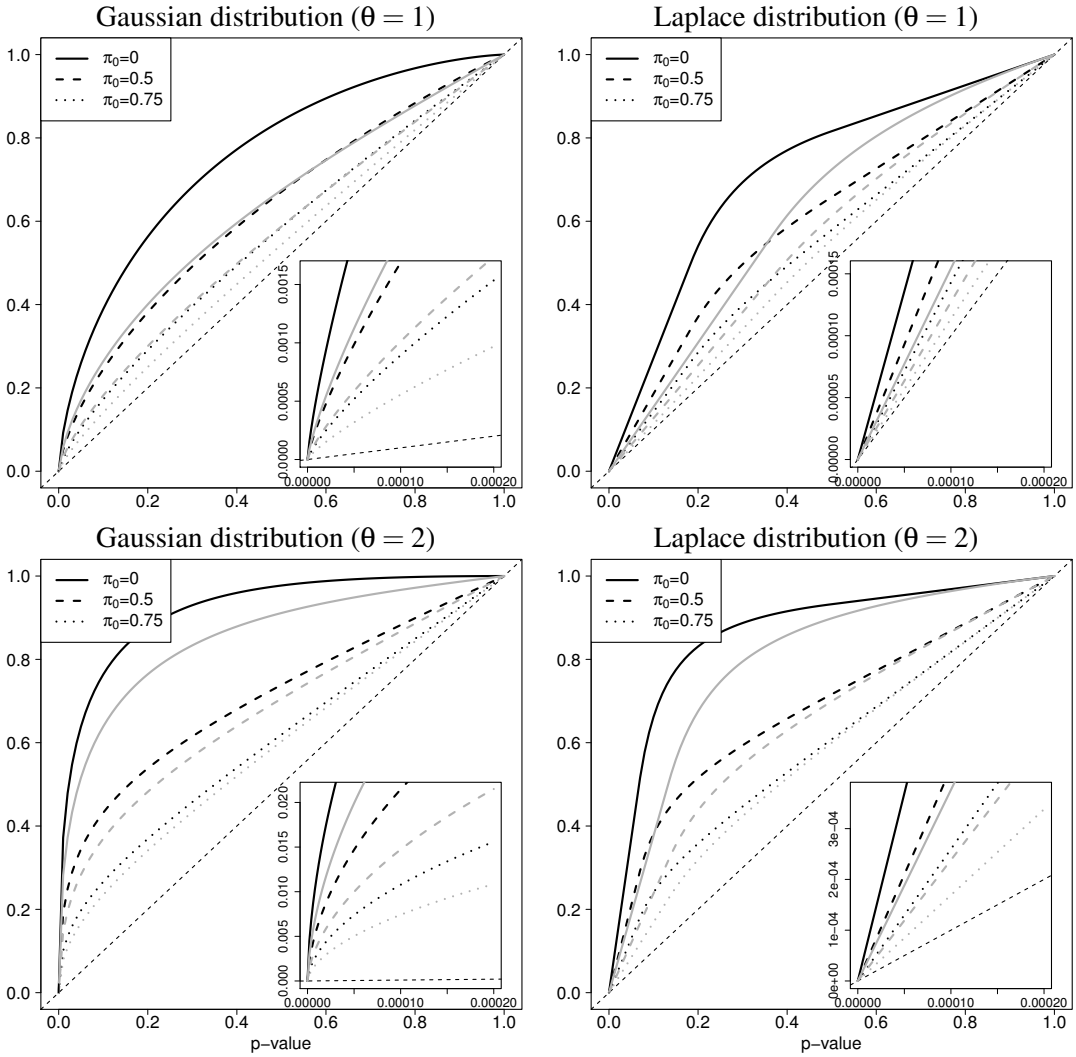



Figure 2: Distribution functions $G$ for one-sided $p$-values (black) and two-sided $p$-values (gray), in Gaussian location models (left: Condition (5) is not satisfied), and Laplace location models (right: Condition (5) is satisfied) for $\pi_0$ =0, 0.5 and 0.75. The location parameter $\theta$ is set to 1 in top panels and 2 in bottom panels. Inserted plot: zoom in the region $p < 2.10^{-4}$.

any $\theta$ and $\pi_0$. This situation is illustrated by Figure 2 (left panels) for the Gaussian model ($\gamma = 2$). In such a situation, for any target FDR level $\alpha$, the asymptotic fraction of rejections by the BH($\alpha$) procedure or by a plug-in procedure of the form BH($\alpha/\hat{\pi}_{0,m}$), where $\hat{\pi}_{0,m} \to \pi_{0,\infty}$ in probability as $m \to +\infty$, is positive by Lemma 27.

If $\gamma = 1$ (Laplace model, as illustrated by Figure 2, right panels), then the likelihood ratio of the model is $f_1^\gamma / f_0^\gamma(x) = \exp(|x| - |x-\theta|)$. It is bounded as $x \to +\infty$, with $\lim_{x\to+\infty} f_1^\gamma / f_0^\gamma(x) = e^\theta$. Therefore, Condition (5) is satisfied for the Laplace location model. Specifically, we have $\underline{\alpha}^\star =$

$\pi_0/(\pi_0+(1-\pi_0)g_1(0))$, with $g_1(0)=e^{\theta}$ for one-sided $p$-values, and $g_1(0)=\cosh\theta$ for two-sided $p$-values. Laplace-distributed test statistics appear as a limit situation in terms of criticality: within the family of $\gamma$-Subbotin location models with $\gamma\in[1,2]$, the Laplace model ($\gamma=1$) is the only one for which Condition (5) is satisfied.

#### 5.2.2 STUDENT MODEL

For the Student model, Equation (12) yields that $(f_1/f_0)(t)$ converges to $s_\nu(\theta)$ as $t\to+\infty$ and $s_\nu(-\theta)$ as $t\to-\infty$, where $s_\nu(\theta)=Hh_\nu(-\theta)/Hh_\nu(0)$ is positive for any $\theta$. Therefore, Condition (5) is satisfied for one-sided and two-sided tests in the Student model (this had already been noted by Chi (2007a) for one-sided tests). Figure 3 gives the distribution function of one- and two-sided $p$-values in the Student model with parameters $\theta\in\{1,2\}$ and $\nu\in\{10,50\}$, for $\pi_0\in\{0,0.5,0.75\}$. Although criticality is much less obvious than for the Laplace model, the inserted plots which zoom into a region where the $p$-values are very small ($p<2.10^{-4}$) do suggest for $\nu=10$ that the slope of the distribution function has a finite limit at 0 for the Student model. As an illustration, we calculated that the critical values for one-sided tests in the Student model for $\pi_0=0.75$ for $\theta\in\{1,2\}$ are respectively 0.173 and 0.015 for $\nu=10$, and $4.10^{-3}$ and $7.10^{-6}$ for $\nu=50$.

### 5.3 Two-Sided Tests

In this section we study consistency and convergence rates for two-sided tests.

#### 5.3.1 CONSISTENCY

Let us first recall that by Proposition 1.2, we have for two-sided tests under a model satisfying Assumption (1):

$$g_1(t)=\frac{1}{2}\left(\frac{f_1}{f_0}\left(q_0(t/2)\right)+\frac{f_1}{f_0}\left(-q_0(t/2)\right)\right), \tag{13}$$

where $q_0:t\mapsto F_0^{-1}(1-t)$ tends to 0 as $t\to 1/2$. A straightforward consequence of (13) is that $g_1(1)=(f_1/f_0)(0)$. As $f_1>0$, we have $g(1)=\pi_0+(1-\pi_0)g_1(1)>\pi_0$. Therefore, Condition (8) is not met, and the kernel estimators of $\pi_0$ studied in Section 3 are not consistent for the estimation of $\pi_0$. Specifically, we have $g_1(1)=e^{-\theta^2/2}$ for Gaussian and Student test statistics, and $g_1(1)=e^{-\theta}$ for Laplace test statistics.

#### 5.3.2 CONVERGENCE RATES

Another consequence of (13) is that if for $k\geq 1$ the likelihood ratio $f_1/f_0$ is $k$ times semi-differentiable at 0, then $g$ is $k$ times (left-)differentiable at 1. In particular, this holds *for any* $k$ in the $\gamma$-Subbotin location model with $\gamma\in[1,2]$, which covers the Gaussian and Laplace cases. It also holds for the Student model (as proved in Proposition 25). For these models, Corollary 17 entails that for any $k>0$, if $\hat{\pi}_{0,m}$ is a kernel estimator of $g$ associated with a $k^{\text{th}}$ order kernel with bandwidth $h_m(k)=m^{-1/(2k+1)}\eta_m^2$ (where $\eta_m\to 0$ and $m\eta_m\to+\infty$ as $m\to+\infty$), then the corresponding plug-in procedure $\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ converges in distribution at rate $m^{-k/(2k+1)}/\eta_m$ for any $\alpha$ greater than $\alpha_0^\star=g(1)\alpha_{BH}^\star$. These results are summarized in the last column of Table 2.

Let us now consider the modification of the Storey-$\lambda$ estimator introduced in Section 3: $\hat{\pi}_{0,m}=\hat{\pi}_{0,m}^{\mathrm{Sto}}(1-h_m)$, with $h_m\to 0$ as $m\to+\infty$. By Corollary 17, the optimal convergence rate of the

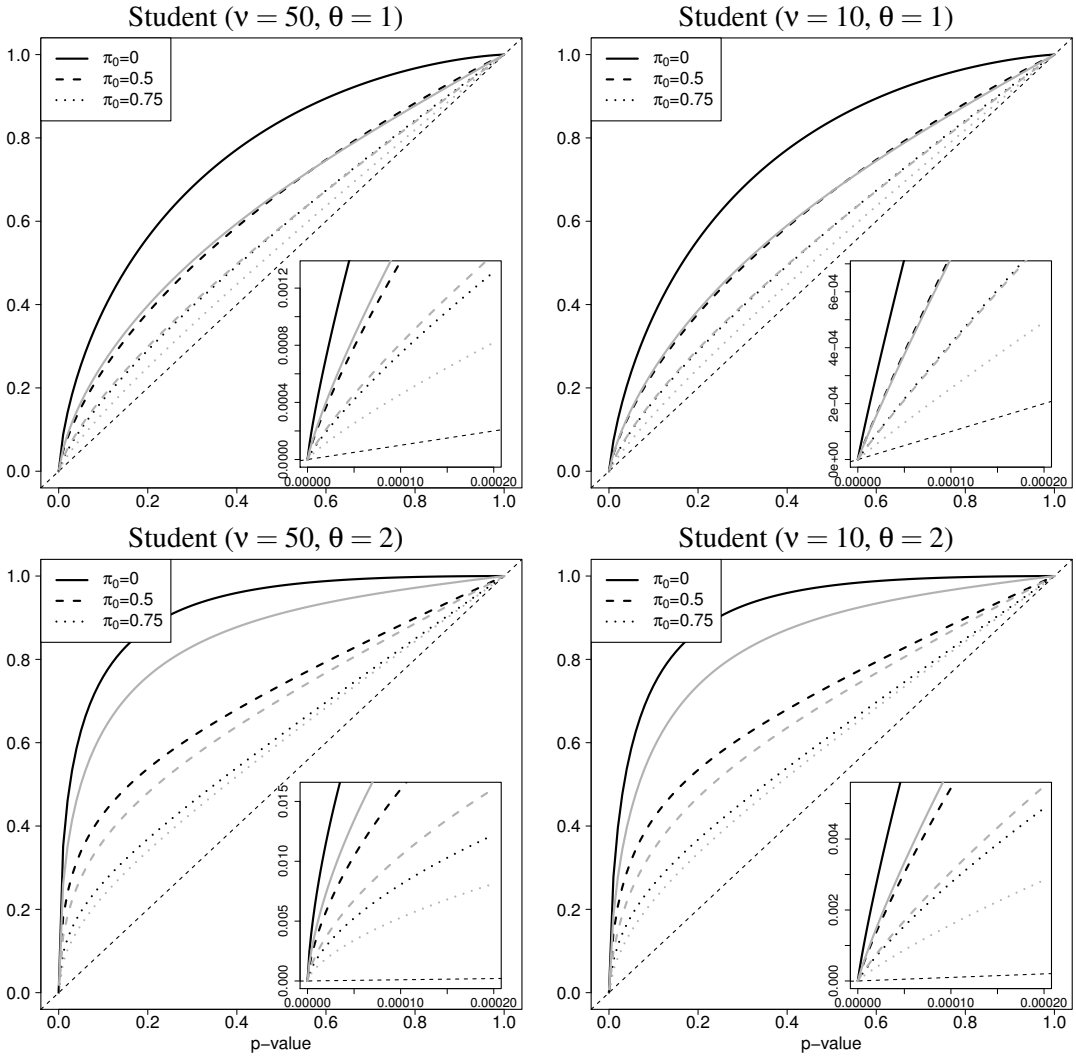



Figure 3: Distribution functions $G$ for one-sided $p$-values (black) and two-sided $p$-values (gray) in Student models with $\nu = 50$ degrees of freedom (left) and $\nu = 10$ (right). The location parameter $\theta$ is set to 1 in top panels and 2 in bottom panels. Any Student model satisfies Condition (5). Inserted plots: zoom in the region $p < 2.10^{-4}$.

$\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ procedure is then determined by the order of the first non null derivative of $g$ at 1. In order to calculate this order, we use the following lemma:

**Lemma 19** *Under Assumption* (1)*, the density function $g_1$ of two-sided p-values under the alternative hypothesis satisfies:*

1. *If $f_1/f_0$ is semi-differentiable at 0, with left-derivative $\ell_-$ and right-derivative $\ell_+$, then $g_1^{(1)}$ is semi-differentiable at 1 and we have:*

$$g_1^{(1)}(1) = -\frac{\ell_+ - \ell_-}{4f_0(0)}. \tag{14}$$

   *In particular, $g_1^{(1)}(1) = 0$ if and only if $f_1/f_0$ is differentiable at 0.*

2. *If $f_1/f_0$ is twice differentiable at 0, then $g_1^{(1)}$ is twice differentiable at 1 and we have:*

$$g_1^{(2)}(1) = \frac{1}{4f_0(0)^2}\left(\frac{f_1}{f_0}\right)^{(2)}(0).$$

Lemma 19 may be applied to two-sided tests for $\gamma$-Subbotin location models, and for the Student model. For the two-sided Gaussian model, $f_1/f_0$ is $C^\infty$ near 0 and $(f_1/f_0)^{(2)}(0) \neq 0$. The same holds for the two-sided Student model, as shown in Appendix A.2 (Proposition 25). For both models, Lemma 19 entails that $g^{(1)}(1) = 0$ and $g^{(2)}(1) > 0$. For two-sided Laplace test statistics, the likelihood ratio $f_1/f_0 : t \mapsto \exp(|t-\theta| - |t|)$ has a singularity at $t = 0$ but it is semi-differentiable at 0 (and differentiable on $(-\infty, \theta) \setminus \{0\}$), with left and right derivatives at 0 given by $\ell_- = 0$ and $\ell_+ = e^{-\theta}$. Lemma 19 yields that $g^{(1)}(1) = -(1-\pi_0)e^{-\theta}/2$. In particular, letting $k = 1$ for the Laplace model and $k = 2$ for the Gaussian and Student models, Corollary 17 yields that if $\hat{\pi}_{0,m} = \hat{\pi}^{\mathrm{Sto}}_{0,m}(1 - m^{-1/(2k+1)}\eta_m^2)$, where $\eta_m \to 0$, then for any $\alpha > \alpha_0^\star = g(1)\alpha_{BH}^\star$, the FDP of the $\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ procedure converges in distribution at rate $m^{-k/(2k+1)}/\eta_m$ toward $\pi_0\alpha/g(1)$, where $g(1) = \pi_0 + (1-\pi_0)e^{-\theta^2/2}$ in the Gaussian and Student models, and $g(1) = \pi_0 + (1-\pi_0)e^{-\theta}/2$ in the Laplace model. These rates are slower than those obtained at the beginning of this section for $k^{\text{th}}$ order kernels because the latter do not require the derivatives of $g$ of order $l < k$ to be null at 1, which implied that any $k > 0$ could be chosen (see Table 2 for a comparison).

### 5.4 One-Sided Tests

In this section we study consistency and convergence rates for one-sided tests.

#### 5.4.1 CONSISTENCY

For one-sided tests, we have $g_1(t) = (f_1/f_0)(q_0(t))$. As $\lim_{t\to 1} q_0(t) = -\infty$, Condition (8) is met if and only if the likelihood ratio $(f_1/f_0)(t)$ tends to 0 as $t \to -\infty$. For the Student model, $f_1/f_0$ tends to $s_\nu(-\theta) > 0$ as $t \to -\infty$. This implies that Condition (8) is not satisfied in that model: $\pi_0$ cannot be consistently estimated using a consistent estimator of $g(1)$, because $g(1) = \pi_0 + (1-\pi_0)e^{-\theta} > \pi_0$. For location models, we begin by establishing a connection between purity and criticality (Proposition 21), which is a consequence of the following symmetry property:

**Lemma 20 (Likelihood ratios in symmetric location models)** *Consider a location model in which the test statistics have densities $f_0$ under $\mathcal{H}_0$, and $f_1 = f_0(\cdot - \theta)$ under $\mathcal{H}_1$ for some $\theta \neq 0$. Under Assumption* (1)*, we have*

$$\lim_{-\infty} \frac{f_0}{f_1} = \lim_{+\infty} \frac{f_1}{f_0}.$$

For one-sided tests in symmetric location models, Lemma 20 implies the following result:

**Proposition 21 (Purity and criticality)** *Let $g_1$ be the density of* one-sided *p-values under the alternative hypothesis, and $\underline{\alpha}^\star$ the critical value of the multiple testing problem. Under Assumption* (1) *and Assumption* (2)*,*

1. *Condition* (5) *and Condition* (8) *are complementary events, in the sense that $\underline{\alpha}^\star = 0$ if and only if $g_1(1) = 0$;*
2. *If $\lim_{+\infty} f_1/f_0$ is finite, then $\underline{\alpha}^\star = \pi_0/(\pi_0 + (1-\pi_0)g_1(0))$ and $g(1) = \pi_0 + (1-\pi_0)g_1(1)$ are connected by $g_1(0)g_1(1) = 1$.*

Proposition 21 implies that contrary to two-sided location models, in which we always have $g_1(1) > 0$, consistency *may* be achieved in one-sided location models using kernel estimators such as those considered here, depending on model parameters. In particular, there is no criticality in the one-sided Gaussian model, implying that Condition (8) is satisfied in that model: we have $g(1) = \pi_0$, and $\pi_0$ can be consistently estimated using the kernel estimators of $g(1)$ introduced in Section 3. In the one-sided Laplace model, Condition (5) is satisfied, implying that Condition (8) is not satisfied in that model: $\pi_0$ cannot be consistently estimated using these kernel estimators of $g(1)$.

### 5.4.2 CONVERGENCE RATES

For the one-sided Student model, Proposition 25 entails that $g_1$ is $C^\infty$, and all its derivatives or order greater than 1 are null at 1. Therefore, any $k > 0$, if $\hat{\pi}^k_{0,m}$ denotes any of the two estimators studied in Corollary 17 for a $k^{\text{th}}$ order kernel with bandwidth $h_m(k) = m^{-1/(2k+1)}\eta_m^2$ (where $\eta_m \to 0$ and $m\eta_m \to +\infty$ as $m \to +\infty$), then the corresponding plug-in procedure $\mathrm{BH}(\alpha/\hat{\pi}^k_{0,m})$ converges in distribution at rate $m^{-k/(2k+1)}/\eta_m$ for any $\alpha$ greater than $\alpha_0^\star = g(1)\alpha^\star_{BH}$. These results are summarized in the first row of Table 2.

For the one-sided Laplace model, the distribution of the $p$-values satisfies $G_1(t) = 1-(1-t)e^{-\theta}$ for $t \geq 1/2$, see Lemma 22 in Appendix A. Therefore, for $t \geq 1/2$, $(1-G(t))/(1-t)$ is constant, equal to $g(1) = \pi_0 + (1-\pi_0)e^{-\theta}$, as illustrated by the solid curves in the right panels of Figure 2. Therefore, for any fixed $\lambda \geq 1/2$, the Storey-$\lambda$ estimator is an unbiased estimator of $g(1)$, which converges to $g(1)$ at rate $m^{-1/2}$. The same property holds for any kernel estimator of $g(1)$ with a fixed bandwidth. These results are summarized in the third row of Table 2.

In the Gaussian model however, the regularity of $g_1$ near 1 is poor for one-sided tests: we have

$$g_1(t) = \exp\left(-\frac{\theta^2}{2} - \theta\Phi^{-1}(t)\right),$$

where $\Phi(= F_0)$ denotes the standard Gaussian distribution function. As $h \to 0$, $\Phi^{-1}(1-h) \leq \sqrt{2\ln(1/h)}$, implying that

$$g_1(1-h) \geq \exp\left(-\frac{\theta^2}{2} - \theta\sqrt{2\ln(1/h)}\right).$$

Therefore, $g_1$ is not differentiable at 1, and the convergence rates of the kernel estimators of $\pi_0$ studied in Section 3 are slower than $m^{-1/3}$ in our setting. These results are summarized in the second row of Table 2.

The difference between one- and two-sided tests in the Gaussian location model is illustrated by Figure 4 for $\theta = 1$, that is when testing $\mathcal{N}(0,1)$ against $\mathcal{N}(1,1)$. The density of two-sided $p$-values has a positive limit at 1, and its derivative at 1 is 0, making it possible to estimate $g(1) =$

| Model | $\lim_0 1/g_1$ | $g_1(1)$ | Convergence rates $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m(k))$ | Convergence rates $\hat{g}^k_m(1)/\eta_m$ |
|---|---|---|---|---|
| 1-sided Student | $s_\nu(\theta)$ | $s_\nu(-\theta)$ | $\ll m^{-k/(2k+1)}/\eta_m$ | $\ll m^{-k/(2k+1)}/\eta_m$ |
| 1-sided Gaussian | 0 | 0 | $\gg m^{-1/3}$ | $\gg m^{-1/3}$ |
| 1-sided Laplace | $e^{-\theta}$ | $e^{-\theta}$ | $m^{-1/2}$ | $m^{-1/2}$ |
| 2-sided Student | $(s_\nu(\theta)+s_\nu(-\theta))/2$ | $e^{-\theta^2/2}$ | $m^{-2/5}/\eta_m$ | $\ll m^{-k/(2k+1)}/\eta_m$ |
| 2-sided Gaussian | 0 | $e^{-\theta^2/2}$ | $m^{-2/5}/\eta_m$ | $\ll m^{-k/(2k+1)}/\eta_m$ |
| 2-sided Laplace | $\cosh\theta$ | $e^{-\theta}$ | $m^{-1/3}/\eta_m$ | $\ll m^{-k/(2k+1)}/\eta_m$ |

Table 2: Properties of one- and two-sided test statistics distributions in Student, Gaussian, and Laplace models, and convergence rates of the kernel estimators studied. When the rate depends on $k$, the value of $k$ may be chosen arbitrarily large. $\eta_m$ is a sequence such that $\eta_m \to 0$ and $m\eta_m \to +\infty$ as $m \to +\infty$.

$\pi_0 + (1-\pi_0)e^{-\theta^2/2}$ at rate $m^{-2/5}$, by Corollary 17. Conversely, the density of one-sided $p$-values tends to 0 at 1, but is not differentiable: the true $\pi_0$ can be estimated consistently, but the convergence rate is slower.

## 6. Concluding Remarks

This paper studies asymptotic properties of a family of plug-in procedures based on the BH procedure. When compared to the BH procedure or to the Storey-$\lambda$ procedure, the results for general models obtained in Section 4 show that incorporating the proposed estimators of $\pi_0$ into the BH procedure asymptotically yields (i) tighter FDR control (or, equivalently, greater power) and (ii) smaller critical values, thereby increasing the range of situations in which the resulting procedure has positive asymptotic power. These improvements come at the price of a reduction in the convergence rate from the parametric rate $m^{-1/2}$ to a non-parametric rate $m^{-k/(2k+1)}$, where $k$ is connected to the order of differentiability of the test statistics distribution. As the results obtained for the proposed modification of the Storey-$\lambda$ estimator $\hat{\pi}^{\mathrm{Sto}}_{0,m}(1-h_m)$ require stronger conditions (null derivatives of $g_1$) than for kernel estimators with a kernel of order $k$, we conclude that it is generally better to use the latter class of estimators.

Our application of these results to specific models for the test statistics sheds some light on the influence of the test statistics distribution on convergence rates of plug-in procedures:

- When the test statistics distribution is $C^\infty$ (which is the case for two-sided Gaussian test statistics, and for Laplace and Student tests statistics), the obtained convergence rates are slower than the parametric rate, but may be arbitrarily close to it by choosing a kernel of sufficiently high order. The resulting estimators are not consistent estimators of $\pi_0$, although the bias decreases as the non-centrality parameter $\theta$ increases.

- When the regularity of the test statistics distribution is poor (such as in the one-sided Gaussian model), the convergence rate of the FDP achieved by the plug-in procedures studied in this paper is slower. The plug-in procedures studied are still asymptotically more powerful than the BH procedure or the Storey-$\lambda$ procedure, but the FDP actually achieved by the plug-in procedures studied for a given $m$ may be far from the target FDR level.

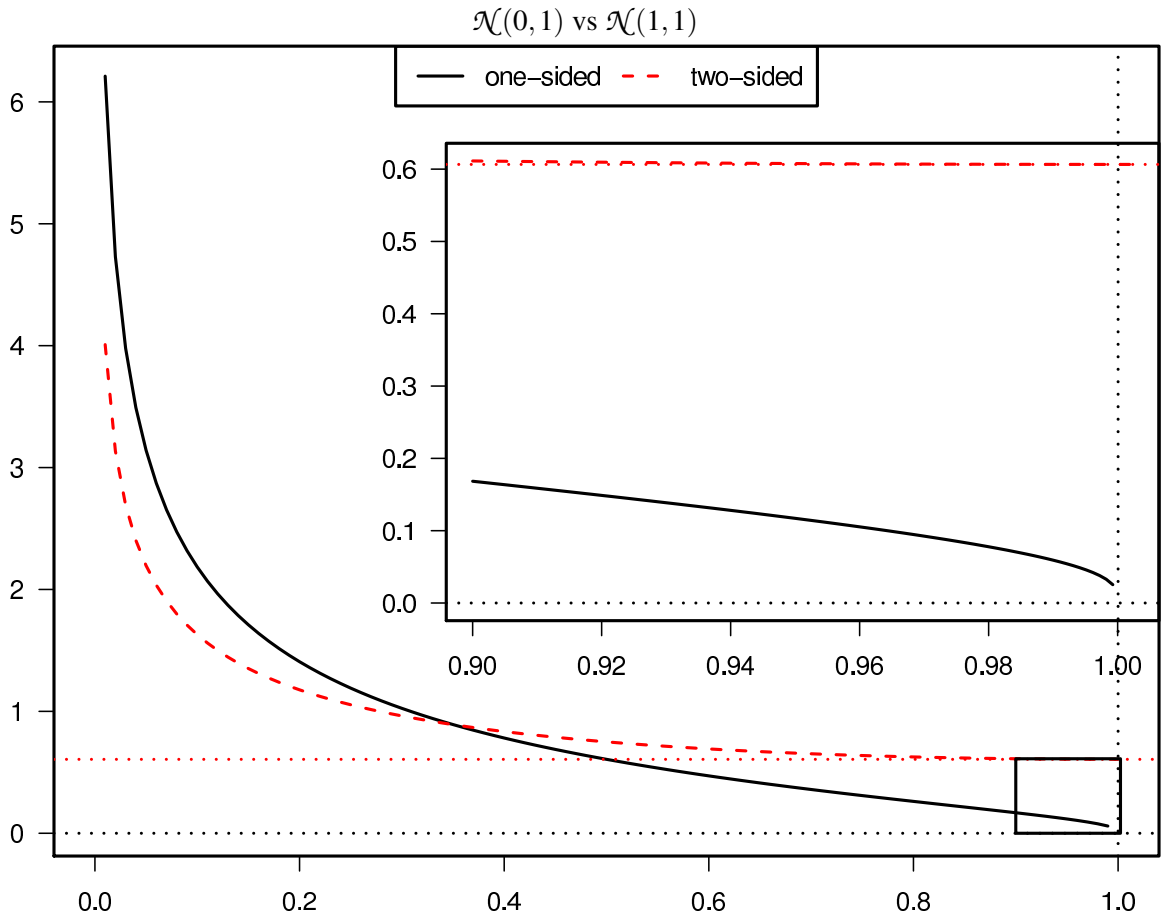



Figure 4: Density of one- and two-sided $p$-values under the alternative hypothesis for the location model $\mathcal{N}(0,1)$ versus $\mathcal{N}(1,1)$. Inserted plot: zoom in the region $[0.9,1]$, which is highlighted by a black box in the main plot.

Obtaining more precise conclusions in the context of a specific data set or application exceeds the scope of the present paper, as it would require extending the obtained results to more realistic settings such as the ones that are now described.

### 6.1 Extensions of the Multiple Testing Setting Considered

An interesting research direction would be to extend the multiple testing setting considered here to more realistic assumptions. A typical example of application is the case of differential expression analyses in genomics, which aim at identifying those genes whose expression level differs between two known populations of samples. First, we have assumed that all null hypotheses are independent, and that all true alternative hypotheses follow the same distribution. The independence assumption is not realistic, as genes are known to interact with each other, in particular through transcriptional regulation networks. Moreover, the degree of differential expression needs not be identical for all genes under $\mathcal{H}_1$. As for the results on criticality that have been used in this paper, the proof given

in Chi (2007a) essentially relies on the assumption that the $p$-values are independently and identically distributed. Therefore, it seems that these results could be extended to composite distributions under $\mathcal{H}_1$, provided that the corresponding marginal distributions are still independently and identically distributed. Extending these results to settings where the independence assumption is relaxed seems a more challenging question. As for the convergence results established in Section 4, their proofs rely on the formalism laid down by Neuvial (2008). Therefore, these results could be extended to other dependency assumptions, or to composite distributions under $\mathcal{H}_1$ provided that the convergence in distribution of the empirical distribution functions $(\widehat{\mathbb{G}}_{0,m}, \widehat{\mathbb{G}}_{1,m})$ holds under these assumptions. In that spirit, the results of Neuvial (2008) have been extended to an equi-correlated Gaussian model (Delattre and Roquain, 2011) and to a more general Gaussian model where the covariance matrix is supposed to be close enough to the identity as the number of tests grows to infinity (Delattre and Roquain, 2012).

Second, we have shown that the asymptotic properties of FDR controlling procedures are driven by the shape and regularity of the test statistics distribution. In practice, the test statistics distribution depends on the size of the sample used to generate them. In differential expression analyses, a natural test statistic is Student's $t$, whose distribution depends on sample size through both the number of degrees of freedom $\nu$ and a non-centrality parameter $\theta$. In the spirit of the results of Chi (2007b) on the influence of sample size on criticality, it would be interesting to study the convergence rates of plug-in procedures when both the sample size and the number of hypotheses tested grow to infinity.

### 6.2 Alternative Strategies to Estimate $\pi_0$

The estimators of $\pi_0$ considered in this paper are kernel estimators of the density $g$ at 1. Therefore, they achieve non-parametric convergence rates of the form $m^{-k/(2k+1)}/\eta_m$, where $k$ controls the regularity of $g$ near 1 and $\eta_m \to 0$ slowly enough. An interesting open question is whether these non-parametric rates may be improved. Other strategies for estimating $\pi_0$ may be considered to achieve faster convergence rates, including the following two:

- One-stage adaptive procedures as proposed by Blanchard and Roquain (2009) and Finner et al. (2009) allow more powerful FDR control than the standard BH procedure without explicitly incorporating an estimate of $\pi_0$: they are not plug-in procedures.

- Jin (2008) proposed an estimator of $\pi_0$ based on the Fourier transform of the empirical characteristic function of the $Z$-scores associated to the $p$-values. This estimator does not focus on the behavior of the density near 1, and might not suffer from the same limitations as the estimators studied here. This estimator was shown to be consistent for the estimation of $\pi_0$ when the $Z$-scores follow a Gaussian location mixture, but no convergence rates were established.

In a general semi-parametric framework where $g_1$ is not necessarily decreasing, and its regularity is not specified, Nguyen and Matias (2013) have recently proved that if the Lebesgue measure of the set on which $g_1$ achieves its minimum is null, then no consistent estimator of $\min_t g(t)$ with a finite asymptotic variance can reach the parametric convergence rate $m^{-1/2}$. In our setting where $g_1$ is decreasing, the measure of the set on which $g_1$ is minimum is indeed null, except if $g_1$ is constant on an interval of the form $[t_0, 1]$. For one-sided tests where $g_1(t) = (f_1/f_0)(F_0^{-1}(1-t))$, this extreme situation arises if and only if the likelihood ratio is constant on an interval of the form $[x_0, +\infty)$. Among all models studied in Section 5, the only case in which this occurs is the one-sided

Laplace model, where $(f_1/f_0)(x) = \exp(|x| - |x-\theta|) = e^{\theta}$ for $x \geq \theta > 0$. The kernel estimators that we have studied here do reach the rate $m^{-1/2}$ in this case.

In the more common situation in which the measure of the set on which $g_1$ vanishes (or achieves its minimum) is null, the above negative result of Nguyen and Matias (2013) suggests that there is little room for improving on the non-parametric convergence rates obtained in Propositions 14 and 15. We conjecture that it is not possible for consistent estimators of $g(1)$ to reach a parametric convergence rate in this setting.

## Acknowledgments

The author would like to thank Stéphane Boucheron for insightful advice and comments, and Catherine Matias and Etienne Roquain for many useful discussions. He is also grateful to anonymous referees for very constructive comments and suggestions that greatly helped improve the paper.

This work was partly supported by the association "Courir pour la vie, courir pour Curie", and by the French ANR project TAMIS.

## Appendix A. Calculations in Specific Models

In this section we perform calculations in location and Student models.

### A.1 Location Models

Lemma 22 gives the distribution of the $p$-value under the alternative hypothesis for one-sided tests in the Laplace model. The proof is straightforward, so it is omitted.

**Lemma 22 (One-sided Laplace location model)** *Assume that the probability distribution function of the test statistics is $f_0 : x \mapsto \frac{1}{2}e^{-|x|}$ under the null hypothesis, and $f_1 : x \mapsto \frac{1}{2}e^{-|x-\theta|}$ under the alternative, with $\theta > 0$ (one-sided test). Then*

1. *The one-sided* p*-value function is*

$$1 - F_0(x) = \begin{cases} \frac{1}{2}e^{(-|x|)} & \text{if } x \geq 0 \\ 1 - \frac{1}{2}e^{(-|x|)} & \text{if } x < 0 \end{cases};$$

2. *The inverse one-sided* p*-value function is*

$$(1 - F_0)^{-1}(t) = \begin{cases} \ln\left(\frac{1}{2t}\right) & \text{if } 0 \leq t \leq \frac{1}{2} \\ \ln(2(1-t)) & \text{if } \frac{1}{2} < t < 1 \end{cases};$$

3. *The cdf of one-sided* p*-values under $\mathcal{H}_1$ is*

$$G_1(t) = \begin{cases} te^{\theta} & \text{if } 0 \leq t \leq \frac{e^{-\theta}}{2} \\ 1 - \frac{1}{4t}e^{-\theta} & \text{if } \frac{e^{-\theta}}{2} \leq t \leq \frac{1}{2} \\ 1 - (1-t)e^{-\theta} & \text{if } t \geq \frac{1}{2} \end{cases};$$

*4. The probability distribution function of one-sided* p*-values under $\mathcal{H}_1$ is*

$$g_1(t) = \begin{cases} e^{\theta} & \text{if } 0 \le t \le \frac{e^{-\theta}}{2} \\ \frac{1}{4t^2} e^{-\theta} & \text{if } \frac{e^{-\theta}}{2} \le t \le \frac{1}{2} \\ e^{-\theta} & \text{if } t \ge \frac{1}{2} \end{cases}.$$

**Proposition 23 (Concavity in two-sided γ-Subbotin models)** *If the test statistics follow a γ-Subbotin distribution with $\gamma \in [1,2]$, then the distribution function of the two-sided p-values under the alternative $G_1$ is concave.*

**Proof** [Proof of Proposition 23] Assumption (1) holds for Subbotin models. By Lemma 2, we need to prove that the likelihood ratio $f_1^\gamma/f_0^\gamma$ of the γ-Subbotin model with γ is such that $h : x \mapsto (f_1^\gamma/f_0^\gamma)(x) + (f_1^\gamma/f_0^\gamma)(-x)$ is non-decreasing on $\mathbb{R}_+$. The function $h$ is differentiable on $(0,+\infty) \setminus \{\theta\}$, and its derivative is given by

$$h'(x) = \left(\frac{f_1^\gamma}{f_0^\gamma}\right)'(x) - \left(\frac{f_1^\gamma}{f_0^\gamma}\right)'(-x),$$

where

$$\left(\frac{f_1^\gamma}{f_0^\gamma}\right)'(y) = \left(\operatorname{sgn}(y)|y|^{\gamma-1} - \operatorname{sgn}(y-\theta)|y-\theta|^{\gamma-1}\right)\frac{f_1^\gamma}{f_0^\gamma}(y) \tag{15}$$

for any $y \in \mathbb{R} \setminus \{0,\theta\}$. Let $x > 0$ such that $x \neq \theta$, we are going to prove that $h'(x) \ge 0$. As $f_1^\gamma/f_0^\gamma$ is non-decreasing, both $\left(f_1^\gamma/f_0^\gamma\right)'(x)$ and $\left(f_1^\gamma/f_0^\gamma\right)'(-x)$ are non-negative. If $\left(f_1^\gamma/f_0^\gamma\right)'(-x) = 0$, then $h'(x) \ge 0$ as desired. From now on, we assume that $\left(f_1^\gamma/f_0^\gamma\right)'(-x)$ is positive. As $\theta > 0$, (15) entails that

$$\frac{\left(f_1^\gamma/f_0^\gamma\right)'(x)}{\left(f_1^\gamma/f_0^\gamma\right)'(-x)} = \frac{x^{\gamma-1} - \operatorname{sgn}(x-\theta)|x-\theta|^{\gamma-1}}{(x+\theta)^{\gamma-1} - x^{\gamma-1}} \frac{f_1(x)^\gamma}{f_1(-x)^\gamma},$$

where $f_1(x)^\gamma > f_1(-x)^\gamma$ because $-|x-\theta| + |x+\theta| > 0$. As $\left(f_1^\gamma/f_0^\gamma\right)'(-x) > 0$, it is enough to show that

$$x^{\gamma-1} - \operatorname{sgn}(x-\theta)|x-\theta|^{\gamma-1} \ge (x+\theta)^{\gamma-1} - x^{\gamma-1} \tag{16}$$

in order to prove that $h'(x) \ge 0$. By the concavity of $x \mapsto x^{\gamma-1}$ on $\mathbb{R}_+$ for $1 \le \gamma \le 2$, $\phi : x \mapsto \theta^{-1}(x^{\gamma-1} - (x-\theta)^{\gamma-1})$ is non-increasing on $[\theta,+\infty]$. Therefore, if $x > \theta$ we have $\phi(x) \ge \phi(x+\theta)$ and (16) holds. If $x < \theta$, then noting that for any $a,b > 0$ and $\zeta \in [0,1]$, $a^\zeta + b^\zeta \ge (a+b)^\zeta$, we have, for $1 \le \gamma \le 2$, $x^{\gamma-1} + (\theta-x)^{\gamma-1} \ge \theta^{\gamma-1} \ge (x+\theta)^{\gamma-1} - x^{\gamma-1}$, and (16) holds as well. ■

### A.2 Student Model

In this section, we use the function series expansion of the Student likelihood ratio given in (11) to study the regularity of this likelihood ratio.

**Lemma 24 (Derivative of the Student likelihood ratio)** *Let $\nu \in \mathbb{N}^*$ and $\theta > 0$. The likelihood ratio $f_1/f_0$ of the Student model with $\nu$ degrees of freedom and non-centrality parameter $\theta$ is $C^1$ on $\mathbb{R}$, and for any $t \in \mathbb{R}$,*

$$\left(\frac{f_1}{f_0}\right)'(t) = \nu(\nu+t^2)^{-3/2} \sum_{j=0}^{+\infty} a_j^1(\nu,\theta)\psi_{(j,\nu)}(t)\,, \tag{17}$$

*where $a_j^1(\nu,\theta) = (j+1)a_{j+1}(\nu,\theta)$ is such that $(\sum_j a_j^1(\nu,\theta))$ converges absolutely.*

**Proof** [Proof of Lemma 24] As $(\sum_j a_j(\nu,\theta))$ converges absolutely and as $\psi_{(j,\nu)}$ is differentiable on $\mathbb{R}$ for any $j \geq 0$ and bounded (by [-1,1]), the dominated convergence theorem ensures that $f_1/f_0$ is differentiable on $\mathbb{R}$ and that its derivative is given by:

$$\left(\frac{f_1}{f_0}\right)'(t) = \sum_{j=1}^{+\infty} a_j(\nu,\theta)\psi'_{(j,\nu)}(t)\,.$$

For $t \neq 0$, we have $\log\left(\mathrm{sgn}(t)^j\psi_{(j,\nu)}(t)\right) = -j/2\left(\log(1+\nu/t^2)\right)$, whose derivative is given by $j\nu/(\nu t+t^3)$, so that

$$\psi'_{(j,\nu)}(t) = \psi_{(j,\nu)}(t)\frac{j\nu}{t(\nu+t^2)}\,.$$

As $\psi_{(j,\nu)}(t) \underset{t\to 0}{\sim} (t/\sqrt{\nu})^j$, we have $\psi_{(j,\nu)}(0) = 0$, $\psi'_{(j,\nu)}(0) = 0$, and $\psi'_{(j,\nu)}$ is continuous at 0. Equation (17) follows by noting that $\psi_{(j+1,\nu)}(t)/\psi_{(j,\nu)}(t) = t/\sqrt{t^2+\nu}$, and that $(\sum_j a_j^1(\nu,\theta))$ converges absolutely by Stirling's formula. ■

Lemma 24 entails the following result:

**Proposition 25 (Regularity of the Student likelihood ratio)** *Let $\nu \in \mathbb{N}^*$ and $\theta > 0$. The likelihood ratio $f_1/f_0$ of the Student model with $\nu$ degrees of freedom and non-centrality parameter $\theta$ is has the following properties:*

1. *$f_1/f_0$ is $C^\infty$ on $\mathbb{R}$;*
2. *For any $k \in \mathbb{N}^*$, we have $(f_1/f_0)^{(k)}(t) \to 0$ as $|t| \to +\infty$;*
3. *$(f_1/f_0)^{(2)}(0) \neq 0$.*

**Proof** [Proof of Proposition 25]

1. By (24), the function series in $(f_1/f_0)'$ has the same form as $f_1/f_0$; therefore, the result easily follows by induction.
2. By (24), Leibniz formula entails that the successive derivatives of $f_1/f_0$ are linear combinations of products of function series of the same form as $f_1/f_0$ by derivatives of $t \mapsto (\nu+t^2)^{-3/2}$. The result follows by the dominated convergence theorem, as all the derivatives of $t \mapsto (\nu+t^2)^{-3/2}$ tend to 0 as $|t| \to +\infty$;
3. The result follows by differentiating (17) at 0.

■

**Proposition 26 (Concavity in the two-sided Student model)** *The distribution function $G_1$ of two-sided p-values in the Student model satisfies (2).*

**Proof** [Proof of Proposition 26] By Lemma 2, we need to prove that the likelihood ratio $f_1/f_0$ of the Student model is such that $t \mapsto (f_1/f_0)(t) + (f_1/f_0)(-t)$ is non-decreasing. Equation (17) yields for $t \in \mathbb{R}$

$$\left(\frac{f_1}{f_0}\right)'(t) + \left(\frac{f_1}{f_0}\right)'(-t) = \nu(\nu+t^2)^{-3/2} \sum_{j=0}^{+\infty} a_j^1(\nu,\theta)\left(\psi_{(j,\nu)}(t) - \psi_{(j,\nu)}(-t)\right), \tag{18}$$

with $\psi_{(j,\nu)}(t) - \psi_{(j,\nu)}(-t) = (1-(-1)^j)(t/\sqrt{\nu+t^2})^{-j}$. Therefore, as $a_j^1(\nu,\theta) > 0$, (18) yields $(f_1/f_0)'(t) + (f_1/f_0)'(-t) \geq 0$, which concludes the proof. ■

## Appendix B. Asymptotics of a Modification of the Storey-λ Estimator

This section gathers the proofs of the asymptotic properties of the estimator $\hat{\pi}_{0,m}^{\mathrm{Sto}}(1-h_m)$ stated in Section 3.

**Proof** [Proof of Proposition 13]

1. We demonstrate that $\hat{\pi}_{0,m}^{\mathrm{Sto}}(1-h_m)$ may be written as a sum of $m$ independent random variables that satisfy the Lindeberg-Feller conditions for the Central Limit Theorem (Pollard, 1984). Let $Z_i^m = \mathbf{1}_{P_i \geq 1-h_m}$, where the $P_i$ are the $p$-values. $Z_i^m$ follows a Bernoulli distribution with parameter $1-G(1-h_m)$. Letting

$$Y_i^m = \frac{Z_i^m - \mathbb{E}[Z_i^m]}{\sqrt{mh_m}},$$

we have $\sum_{i=1}^m Y_i^m = \sqrt{mh_m}\left(\hat{\pi}_{0,m}^{\mathrm{Sto}}(1-h_m) - \mathbb{E}\left[\hat{\pi}_{0,m}^{\mathrm{Sto}}(1-h_m)\right]\right)$. The $(Y_i^m)_{1\leq i\leq m}$ are centered, independent random variables, with $\mathrm{Var}\, Y_i^m = \mathrm{Var}\, Z_i^m/(mh_m) = G(1-h_m)(1-G(1-h_m))/(mh_m)$, which is equivalent to $g(1)/m$ as $m \to +\infty$. Therefore,

$$\lim_{m\to+\infty} \sum_{i=1}^m \mathbb{E}\left[(Y_i^m)^2\right] = g(1).$$

Finally we prove that for any $\varepsilon > 0$,

$$\lim_{m\to+\infty} \sum_{i=1}^m \mathbb{E}\left[(Y_i^m)^2 \mathbf{1}_{|Y_i^m|>\varepsilon}\right] = 0.$$

As $Z_i^m \in \{0,1\}$ and $\mathbb{E}[Z_i^m] \in [0,1]$, we have $(Y_i^m)^2 \leq 1/(mh_m)$, and

$$\begin{aligned}\sum_{i=1}^m \mathbb{E}\left[(Y_i^m)^2 \mathbf{1}_{|Y_i^m|>\varepsilon}\right] &\leq \frac{1}{h_m}\mathbb{E}\left[\mathbf{1}_{|Y_1^m|>\varepsilon}\right] \\ &= \frac{1}{h_m}\mathbb{P}(|Y_1^m|>\varepsilon) \\ &\leq \frac{1}{h_m}\frac{\operatorname{Var} Y_1^m}{\varepsilon^2}\end{aligned}$$

by Chebycheff's inequality. As $mh_m \to +\infty$ and $\operatorname{Var} Y_1^m \sim g(1)/m$ as $m \to +\infty$, the above sum therefore goes to 0 as $mh_m \to +\infty$. The Lindeberg-Feller conditions for the Central Limit Theorem are thus fulfilled, and we have

$$\sum_{i=1}^m Y_i^m \rightsquigarrow \mathcal{N}(0, g(1)),$$

which concludes the proof.

2. As $G(\lambda) = \pi_0\lambda + (1-\pi_0)G_1(\lambda)$, we have, for any $\lambda < 1$,

$$\frac{1-G(\lambda)}{1-\lambda} = \pi_0 + (1-\pi_0)\frac{1-G_1(\lambda)}{1-\lambda}.$$

Therefore, the bias is given by

$$\mathbb{E}\left[\hat{\pi}_{0,m}^{\mathrm{Sto}}(\lambda)\right] - \pi_0 = (1-\pi_0)\frac{1-G_1(\lambda)}{1-\lambda}.$$

A Taylor expansion as $\lambda \to 1$ yields

$$\begin{aligned}1 - G_1(\lambda) &= \sum_{l=0}^k \frac{(-1)^l g_1^{(l)}(1)}{(l+1)!}(1-\lambda)^{l+1} + \mathrm{o}\left((1-\lambda)^{l+1}\right) \\ &= (1-\lambda)g_1(1) + \frac{(-1)^k g_1^{(k)}(1)}{(k+1)!}(1-\lambda)^{k+1} + \mathrm{o}\left((1-\lambda)^{k+1}\right)\end{aligned}$$

as $g_1^{(l)}(1) = (1-\pi_0)^{-1}g^{(l)}(1) = 0$ for $1 \leq l < k$. Therefore, if $h_m \to 0$ as $m \to +\infty$, we have

$$\mathbb{E}\left[\hat{\pi}_{0,m}^{\mathrm{Sto}}(1-h_m)\right] - g(1) = (1-\pi_0)\frac{(-1)^k g_1^{(k)}(1)}{(k+1)!}h_m^k + \mathrm{o}\left(h_m^k\right),$$

which concludes the proof, as $(1-\pi_0)g_1^{(k)}(1) = g^{(k)}(1)$. ■

**Proof** [Proof of Proposition 14] By Proposition 13, the asymptotic variance of $\hat{\pi}_{0,m}(1-h_m)$ is equivalent to $g(1)/(mh_m)$, and the bias is of order $h_m^k$. The optimal bandwidth is obtained for $h_m$ proportional to $m^{-1/(2k+1)}$, because this choice balances variance and squared bias. The proportionality constant is an explicit function of $k$, $\pi_0$, $g_1(1)$, and $g_1^{(k)}(1)$. By definition, the MSE that corresponds

to this optimal choice is twice the corresponding squared bias, that is, of order $m^{-2k/(2k+1)}$, which completes the proof of (1). To prove (2), we note that

$$\sqrt{mh_m}\,(\hat{\pi}_{0,m} - g(1)) = \sqrt{mh_m}\,(\hat{\pi}_{0,m} - \mathbb{E}[\hat{\pi}_{0,m}]) + \sqrt{mh_m}\,(\mathbb{E}[\hat{\pi}_{0,m}] - g(1))\,,$$

where $\hat{\pi}_{0,m}$ denotes $\hat{\pi}_{0,m}(1-h_m)$ to alleviate notation. The first term (variance) converges in distribution to $\mathcal{N}(0, g(1))$ by Proposition 13 (1) as soon as $\sqrt{mh_m} \to +\infty$. The second term (bias) is of the order of $\sqrt{mh_m}h_m^k = \sqrt{mh_m^{2k+1}}$ by Proposition 13 (2). Taking $h_m(k) = h_m^\star(k)\eta_m^2$, where $\eta_m \to 0$, we have $mh_m^{2k+1} \to 0$, which ensures that the bias term converges in probability to 0. ■

## Appendix C. Extension of Neuvial (2008) to the Unconditional Setting

In this section, we show that the results obtained by Neuvial (2008) in the original (conditional) setting of Benjamini and Hochberg (1995) also hold in the unconditional setting considered here, at the price of an additional term in the asymptotic variance due to the fluctuations of the random variable $\pi_{0,m}$. We start by stating a lemma which provides a lower bound on the critical value of plug-in procedures. It is is a consequence of Proposition 10(1).

**Lemma 27** *Let $\alpha_m$ be a sequence of (possibly data-dependent) levels that converges in probability to $\alpha_\infty \in (0,1)$ as $m \to +\infty$. If $\alpha_\infty < \alpha_{BH}^\star$, then the threshold $\widehat{\tau}_m(\alpha_m)$ of the BH$(\alpha_m)$ procedure converges in probability to 0 as $m \to +\infty$. If the convergence of $\alpha_m$ to $\alpha_\infty$ holds almost surely, then the convergence of $\widehat{\tau}_m(\alpha_m)$ to 0 holds almost surely as well.*

**Proof** [Proof of Lemma 27] Assume that $\alpha_m$ converges to $\alpha_\infty$ in probability, with $\alpha_\infty < \alpha_{BH}^\star$. Let $\varepsilon > 0$, we are going to show that there exists an integer $N > 0$ such that for a large enough $m$, the number of rejections of the BH$(\alpha_m)$ procedure is less than $N$ with probability greater than $1-\varepsilon$. Let $\bar{\alpha} = (\alpha_\infty + \alpha_{BH}^\star)/2$. As $\alpha_m \xrightarrow{P} \alpha_\infty < \bar{\alpha}$, there exists an integer $M$ such that for any $m \geq M$, $\alpha_m \leq \bar{\alpha}$ with probability greater than $1-\varepsilon/2$. As $\bar{\alpha} < \alpha_{BH}^\star$, Proposition 10(1) entails that the number of rejections by the BH$(\bar{\alpha})$ procedure is bounded in probability as $m \to +\infty$; that is, there exist two integers $N$ and $M'$ such that for $m \geq M'$, the number of rejections of the BH$(\bar{\alpha})$ procedure is less than $N$ with probability greater that $1-\varepsilon/2$. Thus, for any $m \geq \max(M, M')$, the number of rejections of the BH$(\alpha_m)$ procedure is less than $N$ with probability greater that $1-\varepsilon$. The proof for the almost sure convergence in the case when $\alpha_m$ converges to $\alpha_\infty$ almost surely is similar. ■

We follow the proof technique introduced by Neuvial (2008), by writing the empirical threshold of a given FDR controlling procedure (and its associated FDP) as the result of the application of a *threshold function* of the empirical distribution of the observed $p$-values. As the regularity of the threshold functions involved has already been established by Neuvial (2008), the result is a consequence of the fact that the $p$-value distributions under the null and the alternative hypotheses (as defined below) satisfy Donsker's theorem in the current unconditional setting. This Donsker's theorem has been established by Genovese and Wasserman (2004, Theorem 4.1). For $a \in \{0,1\}$ and $t \in [0,1]$, we let $\widehat{\Gamma}_{a,m}(t) = m^{-1}\sum_{i=1}^m \mathbf{1}_{H_a \text{ true and } P_i \leq t}$.

**Proposition 28 (Genovese and Wasserman 2004)** *As $m \to +\infty$, we have:*

*1.*

$$\sqrt{m}\left(\begin{pmatrix}\widehat{\Gamma}_{0,m}(t)\\ \widehat{\Gamma}_{1,m}(t)\end{pmatrix}-\begin{pmatrix}\pi_0 t\\ (1-\pi_0)g_1(t)\end{pmatrix}\right)\rightsquigarrow\begin{pmatrix}\mathbb{W}_0\\ \mathbb{W}_1\end{pmatrix},$$

*where* $(\mathbb{W}_0,\mathbb{W}_1)$ *is a two-dimensional, centered Gaussian process with covariance function* $\gamma(s,t)$ *defined for any* $(s,t)\in[0,1]^2$ *by*

$$\gamma(s,t)=\begin{pmatrix}\pi_0 s\wedge t-\pi_0^2 st & -\pi_0 s(1-\pi_0)G_1(t)\\ -\pi_0 t(1-\pi_0)G_1(s) & (1-\pi_0)G_1(s\wedge t)-(1-\pi_0)^2G_1(s)G_1(t)\end{pmatrix};$$

*2.*

$$\sqrt{m}\left(\widehat{\mathbb{G}}_m-G\right)\rightsquigarrow\mathbb{W},$$

*where* $\mathbb{W}\overset{(d)}{=}\mathbb{W}_0+\mathbb{W}_1$ *is a one-dimensional, centered Gaussian process with covariance function* $(s,t)\mapsto G(s\wedge t)-G(s)G(t)$.

Note that $\widehat{\Gamma}_{0,m}=\pi_{0,m}\widehat{\mathbb{G}}_{0,m}$ and $\widehat{\Gamma}_{1,m}=(1-\pi_{0,m})\widehat{\mathbb{G}}_{1,m}$, where $(\widehat{\mathbb{G}}_{0,m},\widehat{\mathbb{G}}_{1,m})$ are the empirical distribution functions of the $p$-values under $\mathcal{H}_0$ and $\mathcal{H}_1$, respectively. The results of Neuvial (2008) have been obtained by directly considering the convergence of the process $(\widehat{\mathbb{G}}_{0,m},\widehat{\mathbb{G}}_{1,m})$ instead of $(\widehat{\Gamma}_{0,m},\widehat{\Gamma}_{1,m})$, because $\pi_{0,m}$ was deterministic in the conditional setting (see Neuvial 2009, Theorem 3.1). The results established in Neuvial (2008) (in particular Theorem 3.2) can be translated to the unconditional setting just by replacing the processes $\pi_0\mathbb{Z}_0$ and $\pi_1\mathbb{Z}_1$ in Neuvial (2008) by the processes $\mathbb{W}_0$ and $\mathbb{W}_1$ defined in Proposition 28, and consequently, the process $\mathbb{Z}=\pi_0\mathbb{Z}_0+\pi_1\mathbb{Z}_1$ by $\mathbb{W}=\mathbb{W}_0+\mathbb{W}_1$.

Therefore, the asymptotic properties of the BH procedure and Storey's procedure (that is, $\mathrm{BH}(\cdot/\hat{\pi}_{0,m}^{\mathrm{Sto}}(\lambda))$ in the unconditional setting can be obtained by adapting the proof of the corresponding theorems (Theorems 4.2 and 4.15) in Neuvial (2008):

**Corollary 29 (BH procedure, unconditional setting)** *For any* $\alpha\geq\alpha^\star_{BH}$, *we have*

1. *The asymptotic distribution of the threshold* $\widehat{\tau}_m(\alpha)$ *is given by*

$$\sqrt{m}\left(\widehat{\tau}_m(\alpha)-\tau_\infty(\alpha)\right)\rightsquigarrow\mathcal{N}\left(0,\frac{G(\tau_\infty(\alpha))(1-G(\tau_\infty(\alpha)))}{(1/\alpha-g(\tau_\infty(\alpha)))^2}\right);$$

2. *The asymptotic distribution of the associated FDPs is given by*

$$\sqrt{m}\left(\mathrm{FDP}_m(\widehat{\tau}_m(\alpha))-\pi_0\alpha\right)\rightsquigarrow\mathcal{N}\left(0,(\pi_0\alpha)^2\left(\frac{1}{\pi_0\tau_\infty(\alpha)}-1\right)\right).$$

The asymptotic properties of the BH Oracle procedure are simply obtained by applying Corollary 29 at level $\alpha/\pi_0$.

**Corollary 30 (Storey-$\lambda$ procedure, unconditional model)** *For any* $\lambda\in[0,1)$, *and* $\alpha\in[0,1]$, *let* $\widehat{\tau}_m^{0,\lambda}(\alpha)=\mathcal{T}^{\mathrm{Sto}(\lambda)}(\widehat{\mathbb{G}}_m)$ *be the empirical threshold* $\widehat{\tau}_m^{0,\lambda}(\alpha)$ *of Storey's procedure at level* $\alpha$, *and* $\tau_\infty^{0,\lambda}(\alpha)=\mathcal{T}^{\mathrm{Sto}(\lambda)}(G)$ *be the corresponding asymptotic threshold. Then,*

1. $\alpha^\star_{\mathrm{Sto}(\lambda)}=\overline{\pi_0}(\lambda)\,\alpha^\star_{BH}$ *is the critical value of Storey's procedure;*

2. *For any* $\alpha > \alpha^\star_{\mathrm{Sto}(\lambda)}$*:*

   (a) *The asymptotic distribution of the threshold* $\widehat{\tau}_m^{0,\lambda}(\alpha)$ *is given by*

$$\sqrt{m}\left(\widehat{\tau}_m^{0,\lambda}(\alpha) - \tau_\infty^{0,\lambda}(\alpha)\right) \rightsquigarrow \frac{\tau_\infty^{0,\lambda}(\alpha)}{\overline{\pi_0}(\lambda)/\alpha - g(\tau_\infty^{0,\lambda}(\alpha))}\left\{\frac{\mathbb{W}(\tau_\infty^{0,\lambda}(\alpha))}{\tau_\infty^{0,\lambda}(\alpha)} + \frac{1}{\alpha}\frac{\mathbb{W}(\lambda)}{1-\lambda}\right\},$$

   *where* $\mathbb{W}$ *is a centered Gaussian process with covariance function* $(s,t) \mapsto G(s\wedge t) - G(s)G(t)$*;*

   (b) *The asymptotic distribution of the associated FDPs is given by*

$$\sqrt{m}\left(\mathrm{FDP}_m(\widehat{\tau}_m^{0,\lambda}(\alpha)) - \pi_0\alpha/\overline{\pi_0}(\lambda)\right) \rightsquigarrow \mathcal{N}\left(0,\sigma_\lambda^2\right),$$

   *where*

$$\sigma_\lambda^2 = \left(\frac{\pi_0\alpha}{\overline{\pi_0}(\lambda)}\right)^2\left\{\frac{1}{\pi_0\tau_\infty^{0,\lambda}(\alpha)} + 2\frac{\tau_\infty^{0,\lambda}(\alpha)\wedge\lambda}{\tau_\infty^{0,\lambda}(\alpha)(1-G(\lambda))} - \frac{1}{1-G(\lambda)}\right\}.$$

Note that Corollary 30 with $\lambda = 0$ recovers Corollary 29.

## Appendix D. Asymptotic Properties of Plug-in Procedures

This section gathers the proofs of Section 4.

### D.1 Proof of Theorem 16

We denote by $\widehat{\rho}_m^0(\alpha)$ the proportion of rejections, and by $\widehat{v}_m^0(\alpha)$ the proportion of incorrect rejections by the plug-in procedure $\mathrm{BH}(\alpha/\hat{\pi}_{0,m})$ (among all $m$ hypotheses tested). They may be written as $\widehat{\rho}_m^0(\alpha) = \widehat{\mathbb{G}}_m(\widehat{\tau}_m^0(\alpha)) = \widehat{\tau}_m^0(\alpha)\hat{\pi}_{0,m}/\alpha$ and $\widehat{v}_m^0(\alpha) = \pi_{0,m}\widehat{\mathbb{G}}_{0,m}(\widehat{\tau}_m^0(\alpha))$, respectively. The following Lemma shows that the convergence rate of $(\widehat{\tau}_m^0(\alpha), \widehat{v}_m^0(\alpha), \widehat{\rho}_m^0(\alpha))$ for a large enough $\alpha$ is driven by the convergence rate of $\hat{\pi}_{0,m}$. In order to alleviate notation, we omit the "$(\alpha)$" in $\widehat{\tau}_m^0$, $\widehat{\rho}_m^0$, $\widehat{v}_m^0$, $\tau_\infty^0$, $\rho_\infty^0$, $v_\infty^0$ in the remainder of this section. Moreover, $\mathrm{FDP}_m(\widehat{\tau}_m^0(\alpha))$ will simply be denoted by $\widehat{\mathrm{FDP}}_m^0$.

**Lemma 31** *Let* $\hat{\pi}_{0,m}$ *be an estimator of* $\pi_0$ *such that* $\hat{\pi}_{0,m} \to \pi_{0,\infty}$ *in probability as* $m \to +\infty$*. Define* $\alpha_0^\star = \pi_{0,\infty}\alpha_{BH}^\star$*, and let* $\alpha > \alpha_0^\star$*. Then, under Assumption* (2)*, we have, as* $m \to +\infty$*:*

1. $\widehat{\tau}_m^0$ *converges in probability to* $\tau_\infty^0$ *as* $m \to +\infty$*, with* $g(\tau_\infty^0) < \pi_{0,\infty}/\alpha$*. If the convergence of* $\hat{\pi}_{0,m}$ *to* $\pi_{0,\infty}$ *holds almost surely, then that of* $\widehat{\tau}_m^0$ *to* $\tau_\infty^0$ *holds almost surely as well;*

2. *Further assume that* $\sqrt{mh_m}(\hat{\pi}_{0,m} - \pi_{0,\infty})$ *converges in distribution for some* $h_m$ *such that* $h_m = \mathrm{o}(1/\ln\ln m)$ *and* $mh_m \to +\infty$ *as* $m \to +\infty$*. Then* $(\widehat{\tau}_m^0, \widehat{v}_m^0, \widehat{\rho}_m^0)$ *converges at in distribution at rate* $1/\sqrt{mh_m}$*, with*

$$\begin{pmatrix}\widehat{\tau}_m^0\\ \widehat{v}_m^0\\ \widehat{\rho}_m^0\end{pmatrix} - \begin{pmatrix}\tau_\infty^0\\ v_\infty^0\\ \rho_\infty^0\end{pmatrix} = \frac{\tau_\infty^0/\alpha}{\pi_{0,\infty}/\alpha - g(\tau_\infty^0)}\begin{pmatrix}1\\ \pi_0\\ g(\tau_\infty^0)\end{pmatrix}(\pi_{0,\infty} - \hat{\pi}_{0,m})(1 + \mathrm{o_P}(1)),$$

   *where* $v_\infty^0 = \pi_0\tau_\infty^0$ *and* $\rho_\infty^0 = G(\tau_\infty^0) = \pi_{0,\infty}\tau_\infty^0/\alpha$*.*

**Proof** [Proof of Lemma 31] For 1., we assume that the convergence of $\hat{\pi}_{0,m}$ to $\pi_{0,\infty}$ holds in probability. If it also holds almost surely, then the convergence of $\widehat{\tau}^0_m$ to $\tau^0_\infty$ is almost sure as well. The sketch of the proof is inspired by van der Vaart (1998, Lemma 21.3). Let $\psi_{F,\zeta} : t \mapsto t/\zeta - F(t)$ for any distribution function $F$ and any $\zeta \in (0,1]$. As $\widehat{\mathbb{G}}_m(\widehat{\tau}^0_m) = \hat{\pi}_{0,m}\widehat{\tau}^0_m/\alpha$ and $G(\tau^0_\infty) = \pi_{0,\infty}\tau^0_\infty/\alpha$, we have $\psi_{G,\alpha/\pi_{0,\infty}}(\tau^0_\infty) = 0$ and $\psi_{\widehat{\mathbb{G}}_m,\alpha/\hat{\pi}_{0,m}}(\widehat{\tau}^0_m) = 0$. The proof relies on the following property:

**(a)** $\psi_{G,\alpha/\pi_{0,\infty}}(\widehat{\tau}^0_m)$ converges in probability to $0 = \psi_{G,\alpha/\pi_{0,\infty}}(\tau^0_\infty)$;

**(b)** $\psi_{G,\alpha/\pi_{0,\infty}}$ is locally invertible in a neighborhood of $\tau^0_\infty$, with $\dot{\psi}_{G,\alpha/\pi_{0,\infty}}(\tau^0_\infty) > 0$.

To prove $(a)$, we note that

$$\begin{aligned}-\psi_{G,\alpha/\pi_{0,\infty}}(\widehat{\tau}^0_m) &= G(\widehat{\tau}^0_m) - \pi_{0,\infty}\widehat{\tau}^0_m/\alpha \\ &= (G - \widehat{\mathbb{G}}_m)(\widehat{\tau}^0_m) + (\widehat{\mathbb{G}}_m(\widehat{\tau}^0_m) - \hat{\pi}_{0,m}\widehat{\tau}^0_m/\alpha) + (\hat{\pi}_{0,m} - \pi_{0,\infty})\widehat{\tau}^0_m/\alpha\,.\end{aligned}$$

The first term converges to 0 almost surely, the second one is identically null, and the third one converges in probability to 0 as $\hat{\pi}_{0,m}$ converges in probability to $\pi_{0,\infty}$, and $\widehat{\tau}^0_m \in [0,1]$. Item (b) holds as $G$ in concave (by Assumption (2)) and $\alpha/\pi_{0,\infty} > \alpha^\star_{BH}$, where $\alpha^\star_{BH} = \lim_{u\to 0} u/G(u)$ is the critical value of the BH procedure (see Neuvial 2008, Lemma 7.6 page 1097 for a proof of the invertibility).

Combining $(a)$ and $(b)$, $\widehat{\tau}^0_m$ converges in probability to $\tau^0_\infty$, and $\dot{\psi}_{G,\alpha/\pi_{0,\infty}}(\tau^0_\infty) = \pi_{0,\infty}/\alpha - g(\tau^0_\infty)$ is positive, which proves the first item. For the second item, we only give the proof for $\widehat{\tau}^0_m$, as the proofs for $\widehat{\nu}^0_m$ and $\widehat{\rho}^0_m$ are similar. The idea of the proof is that the fluctuations of $\bar{\mathbb{G}}_m = \widehat{\mathbb{G}}_m - G$, the centered empirical process associated with $G$, are of order $1/\sqrt{m}$ by Donsker's theorem (Donsker, 1951); thus, these fluctuations are negligible with respect to the fluctuations of $\hat{\pi}_{0,m} - \pi_{0,\infty}$, which are assumed to be of order $1/\sqrt{mh_m}$ with $h_m \to 0$. We have

$$\begin{aligned}G(\widehat{\tau}^0_m) - G(\tau^0_\infty) &= (G(\widehat{\tau}^0_m) - \widehat{\mathbb{G}}_m(\widehat{\tau}^0_m)) + (\widehat{\mathbb{G}}_m(\widehat{\tau}^0_m) - G(\tau^0_\infty)) \\ &= -\bar{\mathbb{G}}_m(\widehat{\tau}^0_m) + (\hat{\pi}_{0,m}\widehat{\tau}^0_m/\alpha - \pi_{0,\infty}\tau^0_\infty/\alpha)\end{aligned}$$

because $\widehat{\mathbb{G}}_m(\widehat{\tau}^0_m) = \hat{\pi}_{0,m}\widehat{\tau}^0_m/\alpha$ and $G(\tau^0_\infty) = \pi_{0,\infty}\tau^0_\infty/\alpha$. Therefore,

$$G(\widehat{\tau}^0_m) - G(\tau^0_\infty) = -\bar{\mathbb{G}}_m(\widehat{\tau}^0_m) + \frac{\hat{\pi}_{0,m}}{\alpha}(\widehat{\tau}^0_m - \tau^0_\infty) + \frac{\hat{\pi}_{0,m} - \pi_{0,\infty}}{\alpha}\tau^0_\infty\,.$$

As $\widehat{\tau}^0_m \xrightarrow{P} \tau^0_\infty$ as $m \to +\infty$, we also have $G(\widehat{\tau}^0_m) - G(\tau^0_\infty) = (\widehat{\tau}^0_m - \tau^0_\infty)(g(\tau^0_\infty) + o_P(1))$ by Taylor's formula. Hence we have

$$\left(g(\tau^0_\infty) - \hat{\pi}_{0,m}/\alpha + o_P(1)\right)(\widehat{\tau}^0_m - \tau^0_\infty) = -\bar{\mathbb{G}}_m(\widehat{\tau}^0_m) + (\hat{\pi}_{0,m} - \pi_{0,\infty})\tau^0_\infty/\alpha\,.$$

Now because $\hat{\pi}_{0,m}$ converges in probability to $\pi_{0,\infty}$, we have

$$g(\tau^0_\infty) - \hat{\pi}_{0,m}/\alpha = (g(\tau^0_\infty) - \pi_{0,\infty}/\alpha)(1 + o_P(1)).$$

By 1, we have $\pi_{0,\infty}/\alpha > g(\tau^0_\infty)$, so that for sufficiently large $m$:

$$\widehat{\tau}^0_m - \tau^0_\infty = \frac{\bar{\mathbb{G}}_m(\widehat{\tau}^0_m)}{g(\tau^0_\infty) - \pi_{0,\infty}/\alpha}(1 + o_P(1)) + \frac{\tau^0_\infty/\alpha}{g(\tau^0_\infty) - \pi_{0,\infty}/\alpha}(\hat{\pi}_{0,m} - \pi_{0,\infty})\,.$$

Finally, we note that as $\|\bar{\mathbb{G}}_m\|_\infty \sim c\sqrt{\ln\ln m/m}$ (by the Law of the Iterated Logarithm) and $h_m = \mathrm{o}(1/\ln\ln m)$, we have $\bar{\mathbb{G}}_m(\widehat{\tau}^0_m) = \mathrm{o}_{\mathrm{P}}\left(1/\sqrt{mh_m}\right)$. Moreover, $\sqrt{mh_m}\,(\hat{\pi}_{0,m} - \pi_{0,\infty})$ converges in distribution, so that the term $(\hat{\pi}_{0,m} - \pi_{0,\infty})\tau^0_\infty/\alpha$ dominates the right-hand side. Finally, we have

$$\widehat{\tau}^0_m - \tau^0_\infty = \frac{\tau^0_\infty/\alpha}{g(\tau^0_\infty) - \pi_{0,\infty}/\alpha}(\hat{\pi}_{0,m} - \pi_{0,\infty})(1 + \mathrm{o}_{\mathrm{P}}(1)),$$

which concludes the proof for $\widehat{\tau}^0_m$. ■

**Proof** [Proof of Theorem 16] 1. is a consequence of Lemma 27 combined with Lemma 31(1); 2.(a) is a consequence of Lemma 27(2). Let us prove 2.(b). By Lemma 31, we have

$$\sqrt{mh_m}\left(\begin{pmatrix}\widehat{\nu}^0_m\\ \widehat{\rho}^0_m\end{pmatrix} - \begin{pmatrix}\nu^0_\infty\\ \rho^0_\infty\end{pmatrix}\right) \rightsquigarrow \xi_\infty \begin{pmatrix}\pi_0\\ g(\tau^0_\infty)\end{pmatrix} X,$$

where $X \sim \mathcal{N}(0, s_0^2)$ and

$$\xi_\infty = \frac{\tau^0_\infty/\alpha}{\pi_{0,\infty}/\alpha - g(\tau^0_\infty)}.$$

Recall that $\widehat{\mathrm{FDP}}^0_m = \widehat{\nu}^0_m/(\widehat{\rho}^0_m \vee m^{-1})$. We begin by noting that for a large enough $m$, we have $\widehat{\rho}^0_m > 1/m$ almost surely. This is a consequence of the fact that (i) $\widehat{\rho}^0_m = \widehat{\mathbb{G}}_m(\widehat{\tau}^0_m) = \hat{\pi}_{0,m}\widehat{\tau}^0_m/\alpha$, with $\widehat{\tau}^0_m$ bounded away from 0 (by 1.), and (ii) $\hat{\pi}_{0,m}$ converges to $\pi_{0,\infty} \geq \pi_0 > \alpha$. As a consequence, the factor $m^{-1}$ may be omitted in $\widehat{\mathrm{FDP}}^0_m$ for a large enough $m$; the FDP may then be written as $\widehat{\mathrm{FDP}}^0_m = \gamma(\widehat{\nu}^0_m, \widehat{\rho}^0_m)$, where $\gamma\colon (u,v) \mapsto u/v$ for any $u \geq 0$ and $v > 0$. $\gamma$ is differentiable for any such $(u,v)$, with derivative $\dot{\gamma}_{u,v} = (1/v, -u/v^2) = 1/v(1, -u/v)$. In particular, recalling that $\nu^0_\infty = \pi_0\tau^0_\infty$ and $\rho^0_\infty = G(\tau^0_\infty) = \pi_{0,\infty}\tau^0_\infty/\alpha$, we have

$$\dot{\gamma}_{\nu^0_\infty,\rho^0_\infty} = \frac{\alpha}{\tau^0_\infty\pi_{0,\infty}}\left(1, -\frac{\pi_0\alpha}{\pi_{0,\infty}}\right). \tag{19}$$

As $\gamma(\nu^0_\infty, \rho^0_\infty) = \pi_0\alpha/\pi_{0,\infty}$, the Delta method yields

$$\sqrt{mh_m}\left(\widehat{\mathrm{FDP}}^0_m - \frac{\pi_0\alpha}{\pi_{0,\infty}}\right) \rightsquigarrow \mathcal{N}\left(0, w^2\right),$$

with $w = s_0\xi_\infty\, \dot{\gamma}_{\nu^0_\infty,\rho^0_\infty}\begin{pmatrix}\pi_0\\ g(\tau^0_\infty)\end{pmatrix}$.

By (19), we have $\dot{\gamma}_{\nu^0_\infty,\rho^0_\infty}\begin{pmatrix}\pi_0\\ g(\tau^0_\infty)\end{pmatrix} = \frac{\alpha^2\pi_0}{\tau^0_\infty\pi^2_{0,\infty}}(\pi_{0,\infty}/\alpha - g(\tau^0_\infty))$, which implies that $w = s_0\pi_0\alpha/\pi^2_{0,\infty}$.

■

### D.2 Consistency, Purity and Criticality

**Proof** [Proof of Lemma 20] We note that

$$\begin{aligned}\frac{f_1(x)}{f_0(x)} &= \frac{f_0(x-\theta)}{f_0(x)} \text{ by definition of a location model} \\ &= \frac{f_0(-x+\theta)}{f_0(-x)} \text{ by Assumption (1)} \\ &= \frac{f_0(-x+\theta)}{f_1(-x+\theta)},\end{aligned}$$

which concludes the proof, as $\theta$ is a fixed scalar. ■

**Proof** [Proof of Proposition 21] We have $\alpha^\star_{BH} = \lim_{t\to 0} 1/g(t)$, where $g = \pi_0 + (1-\pi_0)g_1$ and

$$g_1(t) = \frac{f_1}{f_0}\left(-F_0^{-1}(t)\right).$$

Therefore, as $\lim_{t\to 0} F_0^{-1}(t) = +\infty$, the result is a consequence of Lemma 20. ■

### D.3 Regularity of $g_1$ for Two-Sided Tests in Symmetric Models

**Proof** [Proof of Lemma 19]

1. We make the additional assumption that there exists $\eta > 0$ such that $f_1/f_0$ is differentiable on $V_\eta = [-\eta,\eta]\setminus\{0\}$, and that its derivative tends to $\ell_-$ as $u \to 0^-$ and $\ell_+$ as $u \to 0^+$. This assumption makes the proof simpler, and it holds in the models considered in this paper. However, the result still holds (and is simpler to state) without this extra assumption. By Proposition 1, we have under Assumption (1)

$$g_1(t) = \frac{1}{2}\left(\frac{f_1}{f_0}(q_0(t/2)) + \frac{f_1}{f_0}(-q_0(t/2))\right),$$

where $q_0(t/2) = F_0^{-1}(1-t/2)$ maps $Q_\eta = [2(1-F_0(\eta)),1)$ onto $(0,\eta]$. Therefore, $g_1$ is differentiable on $Q_\eta$ and satisfies, for any $t$ in $Q_\eta$:

$$\begin{aligned}g_1^{(1)}(t) &= \frac{1}{2}\left\{\left(\frac{f_1}{f_0}\right)'(q_0(t/2)) - \left(\frac{f_1}{f_0}\right)'(-q_0(t/2))\right\} \times \frac{1}{2}q_0'(t/2) \\ &= -\frac{1}{4f_0(q_0(t/2))}\left(\left(\frac{f_1}{f_0}\right)'(q_0(t/2)) - \left(\frac{f_1}{f_0}\right)'(-q_0(t/2))\right). \qquad (20)\end{aligned}$$

As $t \to 1$, $q_0(t/2) \to 0^+$, (20) implies that $g_1$ is differentiable at 1 with derivative given by (14).

2. Similarly, we prove the result with the extra assumption that $f_1/f_0$ is twice differentiable in a neighborhood of 0. Then (20) entails that $g_1$ is itself twice differentiable in a neighborhood

of 1. Writing $g_1^{(1)}(t) = a(t)b(t)$, with

$$\begin{cases} a(t) & = 1/(4f_0(q_0(t/2))) \\ b(t) & = -(f_1/f_0)'(q_0(t/2)) + (f_1/f_0)'(-q_0(t/2)) \end{cases},$$

we have $g_1^{(2)}(t) = a'(t)b(t) + a(t)b'(t)$. As $q_0(1/2) = F_0^{-1}(1/2) = 0$ , we have $b(1) = 0$, so that $g_1^{(2)}(1) = a(1)b'(1)$, where $a(1) = 1/(4f_0(0))$ and

$$b'(t) = \frac{1}{2f_0(q_0(t/2))} \left( \left( \frac{f_1}{f_0} \right)^{(2)} (q_0(t/2)) + \left( \frac{f_1}{f_0} \right)^{(2)} (-q_0(t/2)) \right).$$

Thus $b'(1) = 1/(2f_0(0)) \times 2(f_1/f_0)^{(2)}(0)$, which concludes the proof.

■

## References


Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 57(1):289–300, 1995.

Y. Benjamini and Y. Hochberg. On the adaptive control of the false discovery rate in multiple testing with independent statistics. *Journal of Educational and Behavioral Statistics*, 25(1):60–83, 2000.

Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics*, 29(4):1165–1188, 2001.

Y. Benjamini, A. M. Krieger, and D. Yekutieli. Adaptive linear step-up procedures that control the false discovery rate. *Biometrika*, 93(3):491–507, 2006.

G. Blanchard and E. Roquain. Adaptive FDR control under independence and dependence. *Journal of Machine Learning Research*, 10:2837–2871, 2009.

Z. Chi. On the performance of FDR control: constraints and a partial solution. *The Annals of Statistics*, 35(4):1409–1431, 2007a.

Z. Chi. Sample size and positive false discovery rate control for multiple testing. *Electronic Journal of Statistics*, 1:77–118, 2007b.

Z. Chi and Z. Tan. Positive false discovery proportions: intrinsic bounds and adaptive control. *Statistica Sinica*, 18(3):837–860, 2008.

S. Delattre and E. Roquain. On the false discovery proportion convergence under Gaussian equicorrelation. *Statistics & Probability Letters*, 81(1):111–115, January 2011.

S. Delattre and E. Roquain. Asymptotics of empirical distribution function for Gaussian subordinated arrays with an application to multiple testing. HAL preprint, 2012. URL `http://hal.archives-ouvertes.fr/hal-00739749`.

M. D. Donsker. An invariance principle for certain probability limit theorems. *Memoirs of the American Mathematical Society*, 6:12, 1951.

B. Efron, R. J. Tibshirani, J. D. Storey, and V. G. Tusher. Empirical bayes analysis of a microarray experiment. *Journal of the American Statistical Association*, 96(456):1151–1160, December 2001.

G. Eklund. Massignifikansproblemet. Unpublished seminar papers, Uppsala University Institute of Statistics, 1961–1963.

H. Finner, T. Dickhaus, and M. Roters. On the false discovery rate and an asymptotically optimal rejection curve. *The Annals of Statistics*, 37(2):596–618, April 2009.

C. R. Genovese and L. Wasserman. A stochastic process approach to false discovery control. *The Annals of Statistics*, 32(3):1035–1061, 2004.

N. W. Hengartner and P. B. Stark. Finite-sample confidence envelopes for shape-restricted densities. *The Annals of Statistics*, 23(2):525–550, 1995.

J. Jin. Proportion of non-zero normal means: universal oracle equivalences and uniformly consistent estimators. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(3): 461–493, 2008.

N. L. Johnson and B. L. Welch. Applications of the non-central $t$-distribution. *Biometrika*, 31(3): 362–389, 1940.

P. Neuvial. Asymptotic properties of false discovery rate controlling procedures under independence. *Electronic Journal of Statistics*, 2:1065–1110, 2008.

P. Neuvial. Corrigendum to "Asymptotic properties of false discovery rate controlling procedures under independence". *Electronic Journal of Statistics*, 3:1083–1083, 2009.

V. H. Nguyen and C. Matias. On efficient estimators of the proportion of true null hypotheses in a multiple testing setup. Hal preprint, 2013. URL `http://hal.archives-ouvertes.fr/hal-00647082`.

D. B. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, 1984.

E. Roquain and F. Villers. Exact calculations for false discovery proportion with application to least favorable configurations. *The Annals of Statistics*, 39(1):584–612, 2011.

T. Schweder and E. Spjøtvoll. Plots of $p$-values to evaluate many tests simultaneously. *Biometrika*, 69(3):493–502, 1982.

P. Seeger. A note on a method for the analysis of significances en masse. *Technometrics*, pages 586–593, 1968.

J. D. Storey. A direct approach to false discovery rates. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(3):479–498, 2002.

J. D. Storey. The positive false discovery rate: A bayesian interpretation and the $q$-value. *The Annals of Statistics*, 31(6):2013–2035, 2003.

J. D. Storey and R. J. Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences of the United States of America*, 100(16):9440–5, August 2003.

J. D. Storey, J. E. Taylor, and D. O. Siegmund. Strong control, conservative point estimation and simultaneous conservative consistency of false discovery rates: a unified approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(1):187–205, 2004.

M. T. Subbotin. On the law of frequency of errors. *Matematicheskii Sbornik*, 31:296–301, 1923.

J. W. H. Swanepoel. The limiting behavior of a modified maximal symmetric $2s$-spacing with applications. *The Annals of Statistics*, 27(1):24–35, 1999.

A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

A. W. van der Vaart. *Asymptotic Statistics*, volume 3 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, Cambridge, 1998.

Journal of Machine Learning Research 14 (2013) 1461-1504 Submitted 4/12; Revised 1/13; Published 6/13

# Conjugate Relation between Loss Functions and Uncertainty Sets in Classification Problems

**Takafumi Kanamori** KANAMORI@IS.NAGOYA-U.AC.JP
*Department of Computer Science and Mathematical Informatics*
*Nagoya University*
*Nagoya 464-8603 Japan*

**Akiko Takeda** TAKEDA@MIST.I.U-TOKYO.AC.JP
**Taiji Suzuki** S-TAIJI@STAT.T.U-TOKYO.AC.JP
*Department of Mathematical Informatics*
*The University of Tokyo*
*Tokyo 113-8656, Japan*

**Editor:** John Shawe-Taylor

## Abstract

There are two main approaches to binary classification problems: the loss function approach and the uncertainty set approach. The loss function approach is widely used in real-world data analysis. Statistical decision theory has been used to elucidate its properties such as statistical consistency. Conditional probabilities can also be estimated by using the minimum solution of the loss function. In the uncertainty set approach, an uncertainty set is defined for each binary label from training samples. The best separating hyperplane between the two uncertainty sets is used as the decision function. Although the uncertainty set approach provides an intuitive understanding of learning algorithms, its statistical properties have not been sufficiently studied. In this paper, we show that the uncertainty set is deeply connected with the convex conjugate of a loss function. On the basis of the conjugate relation, we propose a way of revising the uncertainty set approach so that it will have good statistical properties such as statistical consistency. We also introduce statistical models corresponding to uncertainty sets in order to estimate conditional probabilities. Finally, we present numerical experiments, verifying that the learning with revised uncertainty sets improves the prediction accuracy.

**Keywords:** loss function, uncertainty set, convex conjugate, consistency

## 1. Introduction

In classification problems, the goal is to predict output labels for given input vectors. For this purpose, a decision function defined on the input space is estimated from training samples. The output value of the decision function is used for predicting the labels. In binary classification problems, the sign of the decision function is expected to provide an accurate prediction of the labels. Many learning algorithms use loss functions as a penalty of misclassifications. A decision function minimizing the empirical mean of the loss function over the training samples is employed as an estimator (Cortes and Vapnik, 1995; Schölkopf et al., 2000; Freund and Schapire, 1997; Hastie et al., 2001). For example, the hinge loss, exponential loss and logistic loss are used for support vector machine (SVM), Adaboost and logistic regression, respectively. In regards to binary classification tasks, recent studies have elucidated the statistical properties of learning algorithms using loss functions

©2013 Takafumi Kanamori, Akiko Takeda and Taiji Suzuki.

(see Bartlett et al. 2006; Steinwart 2005, 2003; Schapire et al. 1998; Zhang 2004; Vapnik 1998 for details).

The loss function approach provides not only an estimator of the decision function, but also an estimator of the conditional probability of binary labels for a given input. The sign of the estimated decision function is used for the label prediction, and the magnitude of the decision function is connected to the conditional probability via the loss function. This connection has been studied by many researchers (Friedman et al., 1998; Bartlett and Tewari, 2007). For example, the logistic loss and exponential loss produce logistic models, whereas the hinge loss cannot be used to estimate the conditional probability except the probability 0.5 (Bartlett and Tewari, 2007).

Another approach to binary classification problems, the maximum-margin criterion, is taken in statistical learning. Under the maximum-margin criterion, the best separating hyperplane between the two output labels is used as the decision function. Hard-margin SVM (Vapnik, 1998) defines a convex-hull of input vectors for each binary label, and takes into account the maximum-margin between the two convex-hulls. For the non-separable case, $\nu$-SVM gives us a similar picture (Schölkopf et al., 2000; Bennett and Bredensteiner, 2000). Ellipsoidal sets as well as polyhedral sets such as the convex-hull of finite input points can be used to solve classification problems (Lanckriet et al., 2003; Nath and Bhattacharyya, 2007). In this paper, the set used in the maximum-margin criterion is referred to as an *uncertainty set*. This term comes from the field of robust optimization in mathematical programming (Ben-Tal et al., 2009).

There have been studies on the statistical properties of learning with uncertainty sets. For example, Lanckriet et al. (2003) proposed minimax probability machine (MPM) using ellipsoidal uncertainty sets and studied its statistical properties in the worst-case setting. In statistical learning using uncertainty sets, the main concern is to develop optimization algorithms under the maximum margin criterion (Mavroforakis and Theodoridis, 2006). So far, however, the statistical properties of learning with uncertainty sets have not been studied as much as those of learning with loss functions.

The main purpose of this paper is to study the relation between the loss function approach and uncertainty set approach, and to use the relation to transform learning with uncertainty sets into loss-based learning in order to clarify the statistical properties of learning algorithms. As mentioned above, loss functions naturally involve statistical models of conditional probabilities. As a result, we can establish a correspondence between uncertainty sets and statistical models of conditional probabilities. Note that some of the existing learning methods using uncertainty sets do not necessarily have good statistical properties, such as the statistical consistency. We propose a way of revising uncertainty sets to establish statistical consistency.

Figure 1 shows how uncertainty sets, loss functions and statistical models are related. Starting from a learning algorithm with uncertainty sets, we obtain the corresponding loss function and statistical model via the convex conjugate. Usually, uncertainty sets are designed on the basis of an intuitive understandings of real-world data. By revising uncertainty sets, we can obtain the corresponding loss functions and statistical models. We also derive sufficient conditions under which the corresponding loss function produces a statistically consistent estimator. We think that our method of revising uncertainty sets can bridge the gap between intuitive statistical modeling and the nice statistical properties of learning algorithms.

The paper is organized as follows. Section 2 reviews the existing learning methods using loss functions and uncertainty sets. We describe the relation between the loss function and uncertainty set in $\nu$-SVM. Section 3 is an investigation of the general relation between loss functions and uncertainty sets. In addition, we describe statistical models derived from loss functions. Section 4 shows

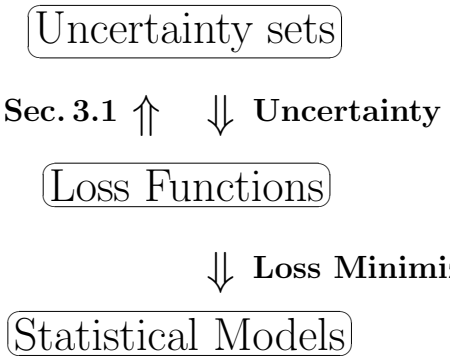



Figure 1: Relations among uncertainty sets, loss functions and statistical models. In Section 3.1, we derive uncertainty sets from loss functions by using the convex conjugate of loss functions. In Section 3.2, we derive statistical models from loss functions. Section 4.2 shows how to revise uncertainty sets in order to obtain loss functions from them. By applying the relations in the diagram, we can transform learning with uncertainty sets into loss-based learning so that we can benefit from good statistical properties such as statistical consistency.

how to revising the uncertainty set so that it will have good statistical properties. Section 5 describes a kernel-based learning algorithm derived from uncertainty sets. Section 6 proves that the kernel-based algorithm has statistical consistency. The results of numerical experiments are described in Section 7. We conclude in section 8. The details of the proofs are shown in the Appendix.

Let us summarize the notations to be used throughout the paper. The indicator function is denoted as $[\![A]\!]$; that is, $[\![A]\!]$ equals 1 if $A$ is true, and 0 otherwise. The column vector $\boldsymbol{x}$ in Euclidean space is written in boldface. The transposition of $\boldsymbol{x}$ is denoted as $\boldsymbol{x}^T$. The Euclidean norm of the vector $\boldsymbol{x}$ is expressed as $\|\boldsymbol{x}\|$. For a set $S$ in a linear space, the convex hull of $S$ is denoted as conv$S$ or conv$(S)$. The number of elements in $S$ is denoted as $|S|$. The expectation of the random variable $Z$ w.r.t. the probability distribution $P$ is described as $\mathbb{E}_P[Z]$. We will drop the subscript $P$ when it is clear from the context. The set of all measurable functions on the set $\mathcal{X}$ relative to the measure $P$ is denoted by $L_0$. The supremum norm of $f \in L_0$ is denoted as $\|f\|_\infty$. Elements in $\mathcal{X}$ are written in Roman alphabets such as $x \in \mathcal{X}$ if $\mathcal{X}$ is not necessarily a subset of the Euclidean space. For the reproducing kernel Hilbert space $\mathcal{H}$, $\|f\|_{\mathcal{H}}$ is the norm of $f \in \mathcal{H}$ defined from the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ on $\mathcal{H}$.

## 2. Preliminaries and Previous Studies

We define $\mathcal{X}$ as the input space and $\{+1,-1\}$ as the set of binary labels. Suppose that the training samples $(x_1,y_1),\ldots,(x_m,y_m) \in \mathcal{X} \times \{+1,-1\}$ are drawn i.i.d. according to a probability distribution $P$ on $\mathcal{X} \times \{+1,-1\}$. The goal is to estimate a decision function $f : \mathcal{X} \to \mathbb{R}$ such that the sign of $f(x)$ provides an accurate prediction of the unknown binary label associated with the input $x$ under the probability distribution $P$. In other words, the probability of $\mathrm{sign}(f(x)) \neq y$ for the estimated

decision function $f$ is expected to be as small as possible.[1] In this article, the composite function of the sign function and the decision function, $\mathrm{sign}(f(x))$, is referred to as classifier.

### 2.1 Learning with Loss Functions

In binary classification problems, the prediction accuracy of the decision function $f$ is measured by the 0-1 loss $[\![\mathrm{sign}(f(x)) \neq y]\!]$, which equals 1 when the sign of $f(x)$ is different from $y$ and 0 otherwise.

The average prediction performance of the decision function $f$ is evaluated by the expected 0-1 loss, that is,

$$\mathcal{E}(f) = \mathbb{E}[\,[\![\mathrm{sign}(f(x)) \neq y]\!]\,].$$

The Bayes risk $\mathcal{E}^*$ is defined as the minimum value of the expected 0-1 loss over all the measurable functions on $\mathcal{X}$,

$$\mathcal{E}^* = \inf\{\mathcal{E}(f) : f \in L_0\}. \tag{1}$$

The Bayes risk is the lowest achievable error rate given the probability $P$. Given a set of training samples, $T = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, the empirical 0-1 loss is expressed as

$$\widehat{\mathcal{E}}_T(f) = \frac{1}{m}\sum_{i=1}^{m} [\![\mathrm{sign}(f(x_i)) \neq y_i]\!].$$

In what follows, the subscript $T$ in $\widehat{\mathcal{E}}_T(f)$ will be dropped if it is clear from the context.

In general, minimization of $\widehat{\mathcal{E}}_T(f)$ is a hard problem (Arora et al., 1997). The main difficulty comes from the non-convexity of the 0-1 loss $[\![\mathrm{sign}(f(x)) \neq y]\!]$ as a function of $f$. Hence, many learning algorithms use a surrogate loss in order to make the computation tractable. For example, SVM uses the hinge loss, $\max\{1 - yf(x), 0\}$, and Adaboost uses the exponential loss, $\exp\{-yf(x)\}$. Both the hinge loss and the exponential loss are convex in $f$, and they provide an upper bound of the 0-1 loss. Thus, the minimizer under the surrogate loss is also expected to minimize the 0-1 loss. The quantitative relation between the 0-1 loss and the surrogate loss was studied by Bartlett et al. (2006) and Zhang (2004).

Regularization is used to avoid overfitting of the estimated decision function to the training samples. The complexity of the estimated classifier is limited by adding a regularization term such as the squared norm of the decision function to an empirical surrogate loss. The balance between the regularization term and the surrogate loss is adjusted by using a regularization parameter (Evgeniou et al., 1999; Steinwart, 2005). Accordingly, regularization controls the deviation of the empirical loss from the expected loss. The optimization is computationally tractable when both the regularization term and the surrogate loss are convex.

Besides computational tractability, surrogate loss functions have another benefit. As discussed by Friedman et al. (1998) and Bartlett and Tewari (2007), surrogate loss functions provide statistical models for the conditional probability of a label $y$ for a given $x$, that is, $P(y|x)$. A brief introduction to this idea is given below.

1. As Bartlett et al. (2006) pointed out, the particular choice of the value of $\mathrm{sign}(0)$ is not important, but we need to choose some value in $\{+1, -1\}$.

Let us consider a minimization problem of the expected loss, $\min_{f \in L_0} \mathbb{E}[\ell(-yf(x))]$, where $\ell(-yf(x))$ is a surrogate loss of a decision function $f(x)$. The function $\ell : \mathbb{R} \to \mathbb{R}$ is assumed to be differentiable. In a similar way to Lemma 1 of Friedman et al. (1998), it is sufficient to minimize the loss function conditional on $x$:

$$\mathbb{E}[\ell(-yf(x))|x] = P(y = +1|x)\ell(-f(x)) + P(y = -1|x)\ell(f(x)).$$

At the optimal solution, the derivative is equal to zero, that is,

$$\frac{\partial}{\partial f(x)} \mathbb{E}[\ell(-yf(x))|x] = -P(y = +1|x)\ell'(-f(x)) + P(y = -1|x)\ell'(f(x)),$$

where $\ell'$ is the derivative of $\ell$. Therefore, we have

$$P(y = +1|x) = \frac{\ell'(f(x))}{\ell'(f(x)) + \ell'(-f(x))}$$

for the optimal solution $f$. An estimator of the conditional probability can be obtained by substituting an estimated decision function into the above expression. For example, the exponential loss $\exp\{-yf(x)\}$ yields the logistic model

$$P(y = +1|x) = \frac{e^{f(x)}}{e^{f(x)} + e^{-f(x)}}.$$

The relation between surrogate losses and statistical models was extensively studied by Bartlett and Tewari (2007).

### 2.2 Learning with Uncertainty Sets

Besides statistical learning using loss functions, there is another approach to binary classification problems, that is, statistical learning based on the *uncertainty set*. What follows is a brief introduction to the basic idea of the uncertainty set. We assume that $\mathcal{X}$ is a subset of Euclidean space.

Uncertainty sets describe uncertainties or ambiguities present in robust optimization problems (Ben-Tal et al., 2009). The parameter in the optimization problem may not be precisely determined. For example, in portfolio optimization, the objective function may depend on a future stock price. Instead of precise information, we have an uncertainty set which probably includes the true parameter of the optimization problem. Typically, the worst case is the setting in which the robust optimization problem with uncertainty sets is solved.

Statistical learning with uncertainty sets is an application of robust optimization to classification problems. An uncertainty set is prepared for each binary label. Each uncertainty set is assumed to include the mean vector of the distribution of input point $\boldsymbol{x}$ conditioned on each label (Takeda et al., 2013). For example, $\mathcal{U}_p$ and $\mathcal{U}_n$ are confidence regions such that the conditional probabilities, $P(\boldsymbol{x} \in \mathcal{U}_p|y = +1)$ and $P(\boldsymbol{x} \in \mathcal{U}_n|y = -1)$, are both equal to 0.95. Another example is one in which the uncertainty set $\mathcal{U}_p$ (resp. $\mathcal{U}_n$) consists of the convex hull of input vectors in training samples having the positive (resp. negative) label. The convex hull of data points is used in hard margin SVM (Bennett and Bredensteiner, 2000). An ellipsoidal uncertainty set is also used for the robust classification in the worst-case setting (Lanckriet et al., 2003; Nath and Bhattacharyya, 2007).

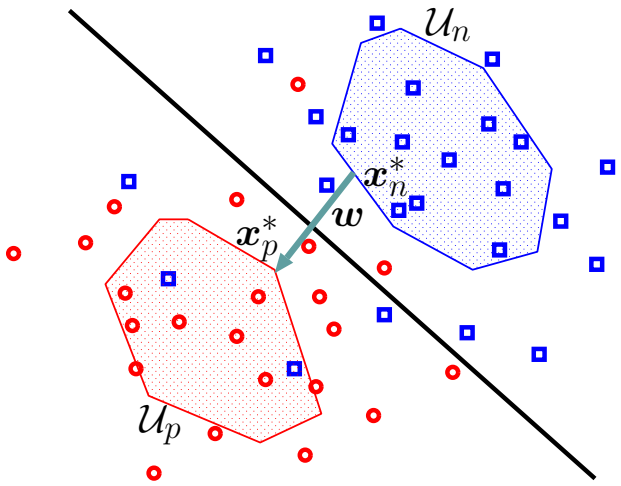



Figure 2: Decision boundary estimated by solving the minimum distance problem with the uncertainty sets $\mathcal{U}_p$ and $\mathcal{U}_n$.

We use the uncertainty set to estimate the linear decision function $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$. Here, let us consider the *minimum distance problem*

$$\min_{\boldsymbol{x}_p, \boldsymbol{x}_n} \|\boldsymbol{x}_p - \boldsymbol{x}_n\| \quad \text{subject to } \boldsymbol{x}_p \in \mathcal{U}_p, \boldsymbol{x}_n \in \mathcal{U}_n. \tag{2}$$

Let $\boldsymbol{x}_p^*$ and $\boldsymbol{x}_n^*$ be optimal solutions of (2). Then, the normal vector of the decision function, $\boldsymbol{w}$, can be estimated with $c(\boldsymbol{x}_p^* - \boldsymbol{x}_n^*)$, where $c$ is a positive real number. Figure 2 illustrates the estimated decision boundary. When both $\mathcal{U}_p$ and $\mathcal{U}_n$ are compact subsets satisfying $\mathcal{U}_p \cap \mathcal{U}_n = \emptyset$, the estimated normal vector cannot be the null vector. The minimum distance problem appears in the hard margin SVM (Vapnik, 1998; Bennett and Bredensteiner, 2000), ν-SVM (Schölkopf et al., 2000; Crisp and Burges, 2000) and the learning algorithms proposed by Nath and Bhattacharyya (2007) and Mavroforakis and Theodoridis (2006). Section 2.3 briefly describes the relation between ν-SVM and the minimum distance problem. Another criterion is used to estimate the linear decision function in minimax probability machine (MPM) proposed by Lanckriet et al. (2003), but the ellipsoidal uncertainty set also plays an important role in MPM.

The minimum distance problem is equivalent to the maximum margin principle (Vapnik, 1998; Bennett and Bredensteiner, 2000). When the bias term $b$ in the linear decision function is estimated such that the decision boundary bisects the line segment connecting $\boldsymbol{x}_p^*$ and $\boldsymbol{x}_n^*$, the estimated decision boundary will have the maximum margin between the uncertainty sets, $\mathcal{U}_p$ and $\mathcal{U}_n$. Takeda et al. (2013) studied the relation between the minimum distance problem and the maximum margin principle.

### 2.3 Loss Functions and Uncertainty Sets in ν-SVM

Here, we will describe how the loss function approach and uncertainty set approach are related to each other in ν-SVM (Schölkopf et al., 2000). We will follow the presentation laid out in Crisp and

Burges (2000) and Bennett and Bredensteiner (2000). We will extend this relation to more general learning algorithms in Section 3.

Suppose that the input space $\mathcal{X}$ is a subset of Euclidean space $\mathbb{R}^d$, and we have the linear decision function, $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$, where the normal vector $\boldsymbol{w} \in \mathbb{R}^d$ and the bias term $b \in \mathbb{R}$ are parameters to be estimated based on training samples. By applying the kernel trick (Berlinet and Thomas-Agnan, 2004; Schölkopf and Smola, 2002), we can obtain rich statistical models for the decision function, while maintaining computational tractability.

The decision function used in $\nu$-SVM is estimated as the optimal solution of

$$\min_{w,b,\rho} \frac{1}{2}\|\boldsymbol{w}\|^2 - \nu\rho + \frac{1}{m}\sum_{i=1}^{m} \max\{\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b), 0\}, \quad \boldsymbol{w} \in \mathbb{R}^d,\ b \in \mathbb{R},\ \rho \in \mathbb{R}, \tag{3}$$

where $\nu \in (0,1)$ is a prespecified constant that acts as the regularization parameter. $\nu$-SVM uses a variant of the hinge loss, $\max\{\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b), 0\}$, as a surrogate loss. As Schölkopf et al. (2000) pointed out, the parameter $\nu$ controls the margin errors and number of support vectors. Roughly speaking, the derivative of the objective function with respect to $\rho$ yields

$$\frac{1}{m}\sum_{i=1}^{m} [\![ y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < \rho ]\!] = \nu, \tag{4}$$

where the subdifferential at $\rho = y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)$ has been ignored for simplicity. The left side of (4) is called the margin error. The quantity $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)$ is referred to as the margin, and the equality above implies that an optimal $\rho$ is the $\nu$-quantile of the empirical distribution of margins $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)$, $i = 1, \ldots, m$. The empirical loss in $\nu$-SVM is minimized over training samples such that $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < \rho$, and training samples having a large margin, that is, $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq \rho$, do not contribute to the loss function $\max\{\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b), 0\}$. As a result, the sum of the second and third terms in $\nu$-SVM (3) imply the mean of the negative margin $-y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)$ such that $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < \rho$, that is,

$$-\nu\rho + \frac{1}{m}\sum_{i=1}^{m} \max\{\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b), 0\} = \nu \cdot \frac{1}{m\nu} \sum_{i:y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < \rho} (-y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b))$$

at the optimal solution. The above loss function is known as the conditional value-at-risk in the field of mathematical finance (Rockafellar and Uryasev, 2002). The relation between $\nu$-SVM and the conditional value-at-risk was studied by Takeda and Sugiyama (2008).

The original formulation of $\nu$-SVM uses a non-negativity constraint, $\rho \geq 0$. As shown by Crisp and Burges (2000), the non-negativity constraint is redundant. Indeed, for an optimal solution $\widehat{\boldsymbol{w}}, \widehat{b}, \widehat{\rho}$, we have

$$-\nu\widehat{\rho} \leq \frac{1}{2}\|\widehat{\boldsymbol{w}}\|^2 - \nu\widehat{\rho} + \frac{1}{m}\sum_{i=1}^{m} \max\{\widehat{\rho} - y_i(\widehat{\boldsymbol{w}}^T\boldsymbol{x}_i + \widehat{b}), 0\} \leq 0,$$

where the last inequality comes from the fact that the parameter, $\boldsymbol{w} = \boldsymbol{0}$, $b = 0$, $\rho = 0$, is a feasible solution of (3). As a result, we have $\widehat{\rho} \geq 0$ for $\nu > 0$.

Now let us briefly show that the dual problem of (3) yields a minimum distance problem in which the reduced convex-hulls of training samples are used as uncertainty sets (See Bennett and

Bredensteiner 2000 for details). Problem (3) is equivalent to

$$\min_{\boldsymbol{w},b,\rho,\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 - \nu\rho + \frac{1}{m}\sum_{i=1}^{m}\xi_i, \quad \text{subject to } \xi_i \geq 0,\ \xi_i \geq \rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b),\ i = 1,\ldots,m.$$

The Lagrangian function is defined as

$$L(\boldsymbol{w},b,\rho,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \nu\rho + \frac{1}{m}\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i(\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - \xi_i) - \sum_{i=1}^{m}\beta_i\xi_i,$$

where $\alpha_i$, $\beta_i$, $i = 1,\ldots,m$ are non-negative Lagrange multipliers. For the training samples, we define $M_p$ and $M_n$ as the set of sample indices for each label, that is,

$$M_p = \{i \mid y_i = +1\}, \quad M_n = \{i \mid y_i = -1\}. \tag{5}$$

The min-max theorem (Bertsekas et al., 2003, Proposition 6.4.3) provides

$$\begin{aligned}
&\inf_{\boldsymbol{w},b,\rho,\boldsymbol{\xi}} \sup_{\boldsymbol{\alpha}\geq\mathbf{0},\boldsymbol{\beta}\geq\mathbf{0}} L(\boldsymbol{w},b,\rho,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) \\
&= \sup_{\boldsymbol{\alpha}\geq\mathbf{0},\boldsymbol{\beta}\geq\mathbf{0}} \inf_{\boldsymbol{w},b,\rho,\boldsymbol{\xi}} L(\boldsymbol{w},b,\rho,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) \\
&= \sup_{\boldsymbol{\alpha}\geq\mathbf{0},\boldsymbol{\beta}\geq\mathbf{0}} \inf_{b,\rho,\boldsymbol{\xi}} -\frac{1}{2}\Big\|\sum_{i=1}^{m}\alpha_i y_i \boldsymbol{x}_i\Big\|^2 + \sum_{i=1}^{m}\xi_i\left(\frac{1}{m} - \alpha_i - \beta_i\right) + \rho\left(\sum_{i=1}^{m}\alpha_i - \nu\right) - b\sum_{i=1}^{m}\alpha_i y_i \qquad (6) \\
&= \sup_{\boldsymbol{\alpha}}\left\{ -\frac{1}{2}\Big\|\sum_{i=1}^{m}\alpha_i y_i \boldsymbol{x}_i\Big\|^2 \ :\ \sum_{i=1}^{m}\alpha_i = \nu,\ \sum_{i=1}^{m}\alpha_i y_i = 0,\ 0 \leq \alpha_i \leq \frac{1}{m}\right\} \\
&= -\frac{\nu^2}{8}\inf_{\boldsymbol{\alpha}}\left\{ \Big\|\sum_{i\in M_p}\gamma_i\boldsymbol{x}_i - \sum_{j\in M_n}\gamma_j\boldsymbol{x}_j\Big\|^2 \ :\ \sum_{i\in M_p}\gamma_i = \sum_{i\in M_n}\gamma_i = 1,\ 0 \leq \gamma_i \leq \frac{2}{m\nu}\right\}. \qquad (7)
\end{aligned}$$

The following equalities should hold in (6) above

$$\frac{1}{m} - \alpha_i - \beta_i = 0,\ (i = 1,\ldots,m), \quad \sum_{i=1}^{m}\alpha_i - \nu = 0, \quad \sum_{i=1}^{m}\alpha_i y_i = 0.$$

Otherwise the objective value tends to $-\infty$. The last equality (7) is obtained by changing the variable from $\alpha_i$ to $\gamma_i = 2\alpha_i/\nu$.

For the positive (resp. negative) label, we introduce the uncertainty set $\mathcal{U}_p$ (reps. $\mathcal{U}_n$) defined by the reduced convex-hull, that is,

$$o \in \{p,n\}, \quad \mathcal{U}_o = \left\{ \sum_{i\in M_o}\gamma_i\boldsymbol{x}_i \ :\ \sum_{i\in M_o}\gamma_i = 1,\ 0 \leq \gamma_i \leq \frac{2}{m\nu}, i \in M_o\right\}.$$

When the upper limit of $\gamma_i$ is less than one, the reduced convex-hull is a subset of the convex-hull of training samples. Hence, solving problem (7) is identical to solving the minimum distance problem with the uncertainty set of reduced convex hulls,

$$\inf_{\boldsymbol{x}_p,\boldsymbol{x}_n} \|\boldsymbol{x}_p - \boldsymbol{x}_n\| \quad \text{subject to } \boldsymbol{x}_p \in \mathcal{U}_p,\ \boldsymbol{x}_n \in \mathcal{U}_n.$$

If the loss function in $\nu$-SVM is scaled, such as,

$$\frac{1}{\nu}\|\boldsymbol{w}\|^2 - 2\rho + \frac{1}{m}\sum_{i=1}^{m}\frac{2}{\nu}\max\{\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b), 0\}, \tag{8}$$

the variable change from the Lagrange multipliers $\alpha_1,\ldots,\alpha_m$ to $\gamma_1,\ldots,\gamma_m$, as shown in (7), is not required to obtain uncertainty sets, $\mathcal{U}_p$ and $\mathcal{U}_n$.

## 3. Relation between Loss Functions and Uncertainty Sets

Here, we present an extension of ν-SVM with which we can investigate the relation between loss functions and uncertainty sets.

### 3.1 Uncertainty Sets Associated with Loss Functions

The decision function is defined as $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ on $\mathbb{R}^d$, and let $\ell : \mathbb{R} \to \mathbb{R}$ be a convex non-decreasing function. For training samples, $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)$, we propose the following learning method, which is an extension of ν-SVM with the expression (8),

$$\inf_{\boldsymbol{w},b,\rho} -2\rho + \frac{1}{m}\sum_{i=1}^{m} \ell(\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)) \ \text{ subject to } \ \|\boldsymbol{w}\|^2 \le \lambda^2,\ b \in \mathbb{R}, \rho \in \mathbb{R}. \tag{9}$$

The regularization effect is introduced by the constraint $\|\boldsymbol{w}\|^2 \le \lambda^2$, where $\lambda$ is a regularization parameter which may depend on the sample size. The above formulation makes the proof of statistical consistency in Section 6 rather simple. Note that ν-SVM is recovered by setting $\ell(z) = \max\{2z/\nu, 0\}$ with an appropriate $\lambda$.

Let us consider the role of the parameter $\rho$ in (9). As described in Section 2.3, $\rho$ in ν-SVM is chosen adaptively, and as a result, training samples with a small margin such as $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) < \rho$ suffer a penalty. The number of training samples suffering a penalty is determined by the parameter ν, and the optimal $\rho$ is the ν-quantile of the empirical margin distribution. As shown below, the ν parameter of ν-SVM is related to the slope of the loss function $\ell(z)$ in (9). In the extended formulation, the $\rho$ parameter in (9) is also adaptively estimated, and it is regarded as a soft-threshold; that is, training samples with margins less than $\rho$ suffer large penalties. The number of such training samples is determined by the extremal condition of (9) with respect to $\rho$:

$$\frac{1}{m}\sum_{i=1}^{m} \ell'(\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)) = 2,$$

where $\ell$ is assumed to have a derivative $\ell'$. In the generalized learning algorithm, the magnitude of the derivative $\ell'$ roughly controls the optimal $\rho$ and the samples size such that margins are smaller than $\rho$. Note that by placing a mild assumption on $\ell$, the first term $-2\rho$ in (9) prevents $\rho$ from going to $-\infty$. The factor 2 in $-2\rho$ can be replaced with an arbitrary positive constant, since multiplying a positive constant by the objective function does not change the optimal solution. However, the factor 2 makes the calculation and interpretation of the dual expression somewhat simpler, as described in the previous section.

We can derive the uncertainty set associated with the loss function $\ell$ in (9) in a similar way to what was done with ν-SVM. We introduce slack variables $\xi_i, i = 1, \ldots, m$ satisfying inequalities $\xi_i \ge \rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b), i = 1, \ldots, m$. Accordingly, the Lagrangian (9) becomes

$$L(\boldsymbol{w}, b, \rho, \boldsymbol{\xi}, \boldsymbol{\alpha}, \mu) = -2\rho + \frac{1}{m}\sum_{i=1}^{m} \ell(\xi_i) + \sum_{i=1}^{m} \alpha_i(\rho - y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - \xi_i) + \mu(\|\boldsymbol{w}\|^2 - \lambda^2),$$

where $\alpha_1, \ldots, \alpha_m$ and $\mu$ are non-negative Lagrange multipliers. We define the convex conjugate of $\ell(z)$ as

$$\ell^*(\alpha) = \sup_{z \in \mathbb{R}} \{z\alpha - \ell(z)\}.$$

The properties of the convex conjugate are summarized in Appendix A. The convex conjugate is mainly used to improve the computational efficiency of learning algorithms (Sun and Shawe-Taylor, 2010). Here, we use the convex conjugate of the loss function to connect seemingly different styles of learning algorithms.

The min-max theorem leads us to the dual problem as follows,

$$\begin{aligned}
&\inf_{\boldsymbol{w},b,\rho,\boldsymbol{\xi}} \sup_{\boldsymbol{\alpha}\geq\mathbf{0},\mu\geq 0} L(\boldsymbol{w},b,\rho,\boldsymbol{\xi},\boldsymbol{\alpha},\mu)\\
&= \sup_{\boldsymbol{\alpha}\geq\mathbf{0},\mu\geq 0} \inf_{\boldsymbol{w},b,\rho,\boldsymbol{\xi}} L(\boldsymbol{w},b,\rho,\boldsymbol{\xi},\boldsymbol{\alpha},\mu)\\
&= \sup_{\boldsymbol{\alpha}\geq\mathbf{0},\mu\geq 0} \inf_{\boldsymbol{w},b,\rho,\boldsymbol{\xi}} \left\{ \rho\left(\sum_{i=1}^m \alpha_i - 2\right) - b\sum_{i=1}^m \alpha_i y_i \right.\\
&\qquad\qquad \left. - \frac{1}{m}\sum_{i=1}^m (m\alpha_i\xi_i - \ell(\xi_i)) - \sum_{i=1}^m \alpha_i y_i \boldsymbol{x}_i^T\boldsymbol{w} + \mu(\|\boldsymbol{w}\|^2 - \lambda^2) \right\}\\
&= -\inf_{\boldsymbol{\alpha}\geq\mathbf{0},\mu\geq 0} \left\{ \frac{1}{m}\sum_{i=1}^m \ell^*(m\alpha_i) + \frac{1}{4\mu}\Big\|\sum_{i=1}^m \alpha_i y_i \boldsymbol{x}_i\Big\|^2 + \mu\lambda^2 \,:\, \sum_{i=1}^m \alpha_i - 2 = 0, \sum_{i=1}^m \alpha_i y_i = 0 \right\}\\
&= -\inf_{\boldsymbol{\alpha}} \left\{ \frac{1}{m}\sum_{i=1}^m \ell^*(m\alpha_i) + \lambda\Big\|\sum_{i\in M_p}\alpha_i\boldsymbol{x}_i - \sum_{i\in M_n}\alpha_i\boldsymbol{x}_i\Big\| \,:\, \sum_{i\in M_p}\alpha_i = \sum_{i\in M_n}\alpha_i = 1, \alpha_i\geq 0 \right\}. \qquad (10)
\end{aligned}$$

Section 6 presents a rigorous proof that by placing certain assumptions on $\ell(\xi)$, the min-max theorem works in the above Lagrangian function; that is, there is no duality gap. For each binary label, we define parametrized uncertainty sets, $\mathcal{U}_p[c]$ and $\mathcal{U}_n[c]$, by

$$o\in\{p,n\},\quad \mathcal{U}_o[c] = \left\{ \sum_{i\in M_o}\alpha_i\boldsymbol{x}_i \,:\, \alpha_i\geq 0,\ \sum_{i\in M_o}\alpha_i = 1,\ \frac{1}{m}\sum_{i\in M_o}\ell^*(m\alpha_i)\leq c \right\}. \qquad (11)$$

Accordingly, the optimization problem in (10) can be represented as

$$\begin{aligned}
&\inf_{c_p,c_n,\boldsymbol{z}_p,\boldsymbol{z}_n} c_p + c_n + \lambda\big\|\boldsymbol{z}_p - \boldsymbol{z}_n\big\|\\
&\quad \text{subject to } \ \boldsymbol{z}_p\in\mathcal{U}_p[c_p],\ \boldsymbol{z}_n\in\mathcal{U}_n[c_n],\ c_p, c_n\in\mathbb{R}.
\end{aligned} \qquad (12)$$

Let $\widehat{\boldsymbol{z}}_p$ and $\widehat{\boldsymbol{z}}_n$ be the optimal solution of $\boldsymbol{z}_p$ and $\boldsymbol{z}_n$ in (12). Let $\widehat{\boldsymbol{w}}$ be an optimal solution of $\boldsymbol{w}$ in (9). The saddle point of the above min-max problem (10) leads to the relation between $\widehat{\boldsymbol{z}}_p$, $\widehat{\boldsymbol{z}}_n$ and $\widehat{\boldsymbol{w}}$. Some calculation yields that $\widehat{\boldsymbol{w}} = \lambda(\widehat{\boldsymbol{z}}_p - \widehat{\boldsymbol{z}}_n)/\|\widehat{\boldsymbol{z}}_p - \widehat{\boldsymbol{z}}_n\|$ holds for $\widehat{\boldsymbol{z}}_p\neq\widehat{\boldsymbol{z}}_n$, and for $\widehat{\boldsymbol{z}}_p = \widehat{\boldsymbol{z}}_n$ any vector such that $\|\widehat{\boldsymbol{w}}\|^2\leq\lambda^2$ satisfies the KKT condition of (9).

The shape of uncertainty sets and the max-margin criterion respectively correspond to the loss function and the regularization principle. Moreover, the size of the uncertainty set is determined by the regularization parameter. Now let us show some examples of uncertainty sets (11) associated with popular loss functions. The index sets in the following examples, $M_p$ and $M_n$, are defined by (5) for the training samples $(\boldsymbol{x}_1,y_1),\ldots,(\boldsymbol{x}_m,y_m)$, and $m_p$ and $m_n$ be $m_p = |M_p|$ and $m_n = |M_n|$.

**Example 1 (ν-SVM)** *Problem* (9) *with* $\ell(z) = \max\{2z/\nu, 0\}$ *reduces to* ν*-SVM. The conjugate function of* $\ell$ *is*

$$\ell^*(\alpha) = \begin{cases} 0, & \alpha\in[0,2/\nu],\\ \infty, & \alpha\notin[0,2/\nu], \end{cases}$$

*and the associated uncertainty set is defined by*

$$o \in \{p,n\}, \quad \mathcal{U}_o[c] = \begin{cases} \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i \,:\, \sum_{i \in M_o} \alpha_i = 1, 0 \le \alpha_i \le \dfrac{2}{m\nu}, i \in M_o \right\}, & c \ge 0, \\ \emptyset, & c < 0. \end{cases}$$

*For $c \ge 0$, the uncertainty set consists of the reduced convex hull of the training samples, and it does not depend on the parameter c. In addition, a negative c is infeasible. Hence, the optimal solutions of $c_p$ and $c_n$ in the problem* (12) *are $c_p = c_n = 0$, and the problem reduces to a simple minimum distance problem.*

**Example 2 (Truncated quadratic loss)** *Let us now consider $\ell(z) = (\max\{1+z, 0\})^2$. The conjugate function is*

$$\ell^*(\alpha) = \begin{cases} -\alpha + \dfrac{\alpha^2}{4}, & \alpha \ge 0, \\ \infty, & \alpha < 0. \end{cases}$$

*For $o \in \{p,n\}$, we define $\bar{\boldsymbol{x}}_o$ and $\widehat{\Sigma}_o$ as the empirical mean and the empirical covariance matrix of the samples $\{\boldsymbol{x}_i : i \in M_o\}$, that is,*

$$\bar{\boldsymbol{x}}_o = \frac{1}{m_o} \sum_{i \in M_o} \boldsymbol{x}_i, \quad \widehat{\Sigma}_o = \frac{1}{m_o} \sum_{i \in M_o} (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_o)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_o)^T.$$

*Suppose that $\widehat{\Sigma}_o$ is invertible. Then, the uncertainty set corresponding to the truncated quadratic loss is*

$$\begin{aligned} o \in \{p,n\}, \quad \mathcal{U}_o[c] &= \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i \,:\, \sum_{i \in M_o} \alpha_i = 1, \alpha_i \ge 0, i \in M_o, \sum_{i \in M_o} \alpha_i^2 \le \frac{4(c+1)}{m} \right\} \\ &= \left\{ \boldsymbol{z} \in \mathrm{conv}\{\boldsymbol{x}_i : i \in M_o\} \,:\, (\boldsymbol{z} - \bar{\boldsymbol{x}}_o)^T \widehat{\Sigma}_o^{-1} (\boldsymbol{z} - \bar{\boldsymbol{x}}_o) \le \frac{4(c+1)m_o}{m} \right\}. \end{aligned}$$

*To prove the second equality, let us define a matrix $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{m_o}) \in \mathbb{R}^{d \times m_o}$. For $\boldsymbol{\alpha}_o = (\alpha_i)_{i \in M_o}$ satisfying the constraints, we get*

$$\boldsymbol{z} = \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i = (X - \bar{\boldsymbol{x}}_o \mathbf{1}^T) \boldsymbol{\alpha}_o + \bar{\boldsymbol{x}}_o,$$

*where $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^{m_o}$. The singular value decomposition of the matrix $X - \bar{\boldsymbol{x}}_o \mathbf{1}^T$ and the constraint $\|\boldsymbol{\alpha}_o\|^2 \le 4(c+1)/m$ yield the second equality. A similar uncertainty set is used in minimax probability machine (MPM) (Lanckriet et al., 2003) and maximum margin MPM (Nath and Bhattacharyya, 2007), though the constraint, $\boldsymbol{z} \in \mathrm{conv}\{\boldsymbol{x}_i : i \in M_o\}$, is not imposed.*

**Example 3 (exponential loss)** *The loss function $\ell(z) = e^z$ is used in Adaboost (Freund and Schapire, 1997; Friedman et al., 1998). The conjugate function is equal to*

$$\ell^*(\alpha) = \begin{cases} -\alpha + \alpha \log \alpha, & \alpha \ge 0, \\ \infty, & \alpha < 0. \end{cases}$$

*Hence, the corresponding uncertainty set is*

$$\mathcal{U}_o[c] = \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i : \sum_{i \in M_o} \alpha_i = 1, \alpha_i \geq 0, i \in M_o, \sum_{i \in M_o} \alpha_i \log \frac{\alpha_i}{1/m_o} \leq c + 1 + \log \frac{m_o}{m} \right\}$$

*for $o \in \{p,n\}$. The Kullback-Leibler divergence from the weights $\alpha_i, i \in M_o$ to the uniform weight is bounded from above in the uncertainty set.*

### 3.2 Statistical Models Associated with Uncertainty Sets

The extended minimum distance problem (12) with the parametrized uncertainty set (11) corresponds to the loss function in (9). We will show the relation between decision functions and conditional probabilities in a similar way to what is shown in Section 2.1. However, instead of the linear decision function $\boldsymbol{w}^T \boldsymbol{x} + b$, we will consider any measurable function $f \in L_0$.

In the learning algorithm (9), the loss function $-2\rho + \ell(\rho - yf(x))$ is used for estimating the decision function. When the sample size tends to infinity, the objective function converges in probability to $\mathbb{E}[-2\rho + \ell(\rho - yf(x))]$. We will show a minimum solution of the expected loss for $f \in L_0$. As described in Section 2.1, it is sufficient to minimize the loss function conditional on $x$. Suppose that $\rho^*$ is the optimal solution of $\mathbb{E}[-2\rho + \ell(\rho - yf(x))]$, and let us minimize $\mathbb{E}[-2\rho^* + \ell(\rho^* - yf(x))|x]$ with respect to $f(x)$, which leads to solving

$$\begin{aligned} &\frac{\partial}{\partial f(x)} \mathbb{E}[-2\rho^* + \ell(\rho^* - yf(x))|x] \\ &= -P(y = +1|x)\ell'(\rho^* - f(x)) + P(y = -1|x)\ell'(\rho^* + f(x)) = 0. \end{aligned}$$

The extremal condition yields

$$P(y = +1|x) = \frac{\ell'(\rho^* + f(x))}{\ell'(\rho^* + f(x)) + \ell'(\rho^* - f(x))} \tag{13}$$

for the optimal solution $\rho^* \in \mathbb{R}$ and $f \in L_0$. An estimator of the conditional probability can be obtained by substituting estimated parameters into the above expression. Given the uncertainty set (11), the corresponding statistical model is defined as (13) via the loss function $\ell(z)$.

## 4. Revision of Uncertainty Sets

Section 3.1 derived parametrized uncertainty sets associated with convex loss functions. Conversely, if an uncertainty set is represented as the form of (11), a corresponding loss function exists. There are many mathematical tools to analyze loss-based estimators. However, if the uncertainty set does not have the form of (11), the corresponding loss function does not exist. One way to deal with the drawback is to revise the uncertainty set so that it possesses a corresponding loss function. This section is devoted to this idea.

Let us consider two different representations of a parametrized uncertainty set: the vertex representation, and the level-set representation. For index sets $M_p$ and $M_n$ defined in (5), let $m_p = |M_p|$ and $m_n = |M_n|$. For $o \in \{p,n\}$, let $L_o$ be a closed, convex, proper function on $\mathbb{R}^{m_o}$, and $L_o^*$ be the conjugate function of $L_o$. The argument of $L_o^*$ is represented by $\boldsymbol{\alpha}_o = (\alpha_i)_{i \in M_o}$. The *vertex*

*representation* of the uncertainty set is defined as

$$\mathcal{U}_o[c] = \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i : L_o^*(\boldsymbol{\alpha}_o) \leq c \right\}, \quad o \in \{p, n\}. \tag{14}$$

Example 2 uses the function $L_o^*(\boldsymbol{\alpha}_o) = \frac{m}{4} \sum_{i \in M_o} \alpha_i^2 - 1$. On the other hand, let $h_o : \mathbb{R}^d \to \mathbb{R}$ be a closed, convex, proper function and $h_o^*$ be the conjugate of $h_o$. The *level-set representation* of the uncertainty set is defined by

$$\mathcal{U}_o[c] = \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i : h_o^*\Big( \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i \Big) \leq c \right\}, \quad o \in \{p, n\}. \tag{15}$$

The function $h_o^*$ may depend on the population distribution. Now suppose that $h_o^*$ does not depend on sample points, $\boldsymbol{x}_i, i \in M_o$. In Example 2, the second expression of the uncertainty set involves the convex function $h_o^*(\boldsymbol{z}) = (\boldsymbol{z} - \bar{\boldsymbol{x}}_o)^T \widehat{\Sigma}_o^{-1} (\boldsymbol{z} - \bar{\boldsymbol{x}}_o)$. This function does not satisfy the assumption, since $h_o^*$ depends on the training samples via $\bar{\boldsymbol{x}}_o$ and $\widehat{\Sigma}_o$. Instead, the function $h_o^*(\boldsymbol{z}) = (\boldsymbol{z} - \boldsymbol{\mu}_o)^T \Sigma_o^{-1} (\boldsymbol{z} - \boldsymbol{\mu}_o)$ with the population mean $\boldsymbol{\mu}_o$ and the population covariance matrix $\Sigma_o$ satisfies the condition. When $\boldsymbol{\mu}_o$ and $\Sigma_o$ are replaced with the estimated parameters based on prior knowledge or samples that are different from the ones used for training, $h_o^*$ with the estimated parameters still satisfies the condition imposed above.

### 4.1 From Uncertainty Sets to Loss Functions

In popular learning algorithms using uncertainty sets such as hard-margin SVM, ν-SVM, and maximum margin MPM, the decision function is estimated by solving the minimum distance problem (2) with $\mathcal{U}_p = \mathcal{U}_p[\bar{c}_p]$ and $\mathcal{U}_n = \mathcal{U}_n[\bar{c}_n]$, where $\bar{c}_p$ and $\bar{c}_n$ are fixed constants. To investigate the statistical properties of learning algorithms using uncertainty sets, we will consider the primal expression of a variant of the minimum distance problem (2).

In Section 3, we expressed problem (12) as the dual form of (9). Here, let us consider the following optimization problem to obtain a loss function corresponding to a given uncertainty set:

$$\begin{aligned} &\min_{c_p, c_n, \boldsymbol{z}_p, \boldsymbol{z}_n} \; c_p + c_n + \lambda \|\boldsymbol{z}_p - \boldsymbol{z}_n\| \\ &\text{subject to} \;\; c_p, c_n \in \mathbb{R}, \\ &\qquad\qquad \boldsymbol{z}_p \in \mathcal{U}_p[c_p] \cap \mathrm{conv}\{\boldsymbol{x}_i : i \in M_p\}, \\ &\qquad\qquad \boldsymbol{z}_n \in \mathcal{U}_n[c_n] \cap \mathrm{conv}\{\boldsymbol{x}_i : i \in M_n\}. \end{aligned} \tag{16}$$

The constraints, $\boldsymbol{z}_o \in \mathrm{conv}\{\boldsymbol{x}_i : i \in M_o\}, o \in \{p, n\}$, are added because the corresponding uncertainty set (11) has them. Suppose that $\mathcal{U}_p[c_p]$ and $\mathcal{U}_n[c_n]$ have the vertex representation (14). Then, (16) is equivalent to

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \; L_p^*(\boldsymbol{\alpha}_p) + L_n^*(\boldsymbol{\alpha}_n) + \lambda \Big\| \sum_{i=1}^m \alpha_i y_i \boldsymbol{x}_i \Big\| \\ &\text{subject to} \;\; \sum_{i \in M_p} \alpha_i = 1, \; \sum_{j \in M_n} \alpha_j = 1, \; \alpha_i \geq 0 \; (i = 1, \ldots, m). \end{aligned}$$

If there is no duality gap, the corresponding primal formulation is

$$\begin{aligned} &\inf_{\boldsymbol{w}, b, \rho, \boldsymbol{\xi}_p, \boldsymbol{\xi}_n} \; -2\rho + L_p(\boldsymbol{\xi}_p) + L_n(\boldsymbol{\xi}_n), \\ &\text{subject to} \;\; \rho - y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \leq \xi_i, \; i = 1, \ldots, m, \quad \|\boldsymbol{w}\|^2 \leq \lambda^2, \end{aligned} \tag{17}$$

where $\boldsymbol{\xi}_o$ is defined as $\boldsymbol{\xi}_o = (\xi_i)_{i \in M_o}$ for $o \in \{p, n\}$.

In the primal expression (17), $L_p$ and $L_n$ are regarded as loss functions for the decision function $\boldsymbol{w}^T \boldsymbol{x} + b$ on the training samples. In general, however, the loss function is not represented as the empirical mean over training samples.

### 4.2 Revised Uncertainty Sets and Corresponding Loss Functions

The uncertainty sets can be revised such that the primal form (17) is represented as minimization of the empirical mean of a loss function. Theorem 1 below is the justification for this revision.

**Revision of uncertainty set defined by vertex representation:** Suppose that the uncertainty set is described by (14). For $o \in \{p, n\}$, we define $m_o$-dimensional vectors $\mathbf{1}_o = (1, \ldots, 1)$ and $\mathbf{0}_o = (0, \ldots, 0)$. For the convex function $L_o^* : \mathbb{R}^{m_o} \to \mathbb{R}$, we define $\bar{\ell}^* : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ by

$$\bar{\ell}^*(\alpha) = \begin{cases} L_p^*(\frac{\alpha}{m}\mathbf{1}_p) + L_n^*(\frac{\alpha}{m}\mathbf{1}_n) - L_p^*(\mathbf{0}_p) - L_n^*(\mathbf{0}_n) & \alpha \geq 0, \\ \infty, & \alpha < 0. \end{cases} \tag{18}$$

The revised uncertainty set $\bar{\mathcal{U}}_o[c]$, $o \in \{p, n\}$ is defined as

$$\bar{\mathcal{U}}_o[c] = \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i \,:\, \sum_{i \in M_o} \alpha_i = 1, \alpha_i \geq 0, i \in M_o, \frac{1}{m} \sum_{i \in M_o} \bar{\ell}^*(\alpha_i m) \leq c \right\}. \tag{19}$$

**Revision of uncertainty set defined by level-set representation:** Suppose that the uncertainty set is described by (15) and that the mean of the input vector $\boldsymbol{x}$ conditioned on the positive (resp. negative) label is given as $\boldsymbol{\mu}_p$ (resp. $\boldsymbol{\mu}_n$). The null vector is denoted as $\mathbf{0}$. We define the function $\bar{\ell}^* : \mathbb{R} \to \mathbb{R}$ by

$$\bar{\ell}^*(\alpha) = \begin{cases} h_p^*(\alpha\frac{m_p}{m}\boldsymbol{\mu}_p) + h_n^*(\alpha\frac{m_n}{m}\boldsymbol{\mu}_n) - h_p^*(\mathbf{0}) - h_n^*(\mathbf{0}) & \alpha \geq 0, \\ \infty, & \alpha < 0. \end{cases} \tag{20}$$

For $\bar{\ell}^*(\alpha)$ in (20), the revised uncertainty set $\bar{\mathcal{U}}_o[c], o \in \{p, n\}$ is defined in the same way as (19). We apply a parallel shift to the training samples so as to be $\boldsymbol{\mu}_p \neq \mathbf{0}$ or $\boldsymbol{\mu}_n \neq \mathbf{0}$.

Now let us explain why the revised uncertainty set is defined as it is. When the function $L_p^* + L_n^*$ is described in additive form such as $\sum_{i=1}^m g(\alpha_i)$ for a function $g$, the uncertainty set defined by the revision (18) does not change. Indeed, Theorem 1 below implies that the transformation of $L_p^* + L_n^*$ into $\frac{1}{m}\sum_{i=1}^m \bar{\ell}^*(\alpha_i m)$ is a projection onto the set of functions with an additive form. In other words, performing the revision twice is the same as performing it once. In addition, the second statement of Theorem 1 means that the projection is uniquely determined when we impose the condition in which the function values on the diagonal $\{(\alpha, \ldots, \alpha) \in \mathbb{R}^m \,:\, \alpha \geq 0\}$ remain unchanged.

**Theorem 1** *Let $L_o^* : \mathbb{R}^{m_o} \to \mathbb{R}$, $o \in \{p, n\}$ be convex functions and $\bar{\ell}^*$ be the function defined by* (18) *for given $L_p^*$ and $L_n^*$. Suppose that $\ell : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ is a closed, convex, proper function such that $\ell^*(0) = 0$ and $\ell^*(\alpha) = \infty$ for $\alpha < 0$ hold.*

1. *Suppose that*

$$L_p^*(\boldsymbol{\alpha}_p) + L_n^*(\boldsymbol{\alpha}_n) - L_p^*(\mathbf{0}_p) - L_n^*(\mathbf{0}_n) = \frac{1}{m}\sum_{i=1}^{m} \ell^*(\alpha_i m) \tag{21}$$

*holds for all non-negative* $\alpha_i$, $i = 1, \ldots, m$. *Then, the equality* $\bar{\ell}^* = \ell^*$ *holds.*

2. *Suppose further that*

$$L_p^*(\alpha \mathbf{1}_p) + L_n^*(\alpha \mathbf{1}_n) - L_p^*(\mathbf{0}_p) - L_n^*(\mathbf{0}_n) = \frac{1}{m}\sum_{i=1}^{m} \ell^*(\alpha m) = \ell^*(\alpha m)$$

*holds for all* $\alpha \geq 0$. *Then, the equality* $\bar{\ell}^* = \ell^*$ *holds.*

**Proof** Let us prove the first statement. From the definition of $\bar{\ell}^*$ and the assumption placed on $\ell^*$, the equality $\ell^*(\alpha) = \bar{\ell}^*(\alpha)$ holds for $\alpha < 0$. Next, suppose $\alpha \geq 0$. The assumption (21) leads to $L_p^*(\frac{\alpha}{m}\mathbf{1}_p) + L_n^*(\frac{\alpha}{m}\mathbf{1}_n) - L_p^*(\mathbf{0}_p) - L_n^*(\mathbf{0}_n) = \ell^*(\alpha)$. Hence, we have $\ell^* = \bar{\ell}^*$. The second statement of the theorem is straightforward. ■

Next, we show that the formula (20) is valid. We want to find a function $\bar{\ell}^*(\alpha)$ such that $h_p^*(\sum_{i\in M_p} \alpha_i \boldsymbol{x}_i) + h_n^*(\sum_{i\in M_n} \alpha_i \boldsymbol{x}_i) - h_p^*(\mathbf{0}) - h_n^*(\mathbf{0})$ is close to $\frac{1}{m}\sum_{i=1}^{m} \bar{\ell}^*(m\alpha_i)$ in some sense. To do so, we substitute $\alpha_i = \alpha/m$ into $h_o^*(\sum_{i\in M_o} \alpha_i \boldsymbol{x}_i)$, $o \in \{p, n\}$. In the large sample limit, $h_o^*(\sum_{i\in M_o} \frac{\alpha}{m} \boldsymbol{x}_i)$ is approximated by $h_o^*(\alpha \frac{m_o}{m} \boldsymbol{\mu}_o)$. Suppose that

$$h_p^*(\alpha \frac{m_p}{m} \boldsymbol{\mu}_p) + h_n^*(\alpha \frac{m_n}{m} \boldsymbol{\mu}_n) - h_p^*(\mathbf{0}) - h_n^*(\mathbf{0})$$

is represented as $\frac{1}{m}\sum_{i=1}^{m} \bar{\ell}^*(\frac{\alpha}{m} m) = \bar{\ell}^*(\alpha)$. As a result, we get (20).

The expanded minimum distance problem using the revised uncertainty sets $\bar{\mathcal{U}}_p[c]$ and $\bar{\mathcal{U}}_n[c]$ is

$$\min_{c_p, c_n, \boldsymbol{z}_p, \boldsymbol{z}_n} c_p + c_n + \lambda \|\boldsymbol{z}_p - \boldsymbol{z}_n\| \quad \text{subject to } \boldsymbol{z}_p \in \bar{\mathcal{U}}_p[c_p],\ \boldsymbol{z}_n \in \bar{\mathcal{U}}_n[c_n]. \tag{22}$$

The corresponding primal problem is

$$\inf_{w, b, \rho, \xi_p, \xi_n} -2\rho + \frac{1}{m}\sum_{i=1}^{m} \bar{\ell}(\xi_i) \qquad \text{subject to } \rho - y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \leq \xi_i,\ i = 1, \ldots, m,\ \ \|\boldsymbol{w}\|^2 \leq \lambda^2.$$

The revision of uncertainty sets leads to the empirical mean of the revised loss function $\bar{\ell}$. Asymptotic analysis can be used to study the statistical properties of the estimator given by the optimal solution of (22), since the objective in the primal expression is described by the empirical mean of the revised loss function.

Now let us show some examples to illustrate how revision of uncertainty sets works.

**Example 4** *Let* $L_o^*, o \in \{p, n\}$ *be the convex function* $L_o^*(\boldsymbol{\alpha}_o) = \boldsymbol{\alpha}_o^T C_o \boldsymbol{\alpha}_o$, *where* $C_o$ *is a positive definite matrix. When both* $C_p$ *and* $C_n$ *are the identity matrix, the following equality holds:*

$$L_p^*(\boldsymbol{\alpha}_p) + L_n^*(\boldsymbol{\alpha}_n) = \frac{1}{m}\sum_{i=1}^{m} \bar{\ell}^*(\alpha_i m) = \sum_{i=1}^{m} \alpha_i^2.$$

*The revised function defined by* (18) *is*

$$\bar{\ell}^*(\alpha) = \alpha^2 \frac{\mathbf{1}_p^T C_p \mathbf{1}_p + \mathbf{1}_n^T C_n \mathbf{1}_n}{m^2}$$

*for* $\alpha \geq 0$. *Accordingly, we get*

$$\frac{1}{m}\sum_{i=1}^{m} \bar{\ell}^*(\alpha_i m) = \frac{\mathbf{1}_p^T C_p \mathbf{1}_p + \mathbf{1}_n^T C_n \mathbf{1}_n}{m} \sum_{i=1}^{m} \alpha_i^2.$$

*Let* $k$ *be* $k = \mathbf{1}_p^T C_p \mathbf{1}_p + \mathbf{1}_n^T C_n \mathbf{1}_n$. *The revised uncertainty set is*

$$o \in \{p,n\}, \quad \bar{\mathcal{U}}_o[c] = \left\{ \sum_{i \in M_o} \alpha_i \boldsymbol{x}_i \,:\, \sum_{i \in M_o} \alpha_i = 1, \alpha_i \geq 0\,(i \in M_o), \sum_{i \in M_o} \alpha_i^2 \leq \frac{cm}{k} \right\}.$$

*For* $o \in \{p,n\}$, *let* $\bar{\boldsymbol{x}}_o$ *and* $\widehat{\Sigma}_o$ *be the empirical mean and the empirical covariance matrix,*

$$\bar{\boldsymbol{x}}_o = \frac{1}{m_o} \sum_{i \in M_o} \boldsymbol{x}_i, \quad \widehat{\Sigma}_o = \frac{1}{m_o} \sum_{i \in M_o} (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_o)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_o)^T.$$

*If* $\widehat{\Sigma}_o$ *is invertible, we have*

$$\bar{\mathcal{U}}_o[c] = \left\{ \boldsymbol{z} \in \mathrm{conv}\{\boldsymbol{x}_i : i \in M_o\} \,:\, (\boldsymbol{z} - \bar{\boldsymbol{x}}_o)^T \widehat{\Sigma}_o^{-1} (\boldsymbol{z} - \bar{\boldsymbol{x}}_o) \leq \frac{cmm_o}{k} \right\}.$$

*In the learning algorithm based on the revised uncertainty set, the estimator is obtained by solving*

$$\begin{aligned} &\min_{c_p, c_n, \boldsymbol{z}_p, \boldsymbol{z}_n} c_p + c_n + \lambda \|\boldsymbol{z}_p - \boldsymbol{z}_n\| \;\; \textit{subject to} \;\; \boldsymbol{z}_p \in \bar{\mathcal{U}}_p[c_p],\, \boldsymbol{z}_n \in \bar{\mathcal{U}}_n[c_n] \\ \Longleftrightarrow &\min_{c_p, c_n, \boldsymbol{z}_p, \boldsymbol{z}_n} c_p + c_n + \frac{m^2\lambda}{4k} \|\boldsymbol{z}_p - \boldsymbol{z}_n\| \;\; \textit{subject to} \;\; \boldsymbol{z}_p \in \bar{\mathcal{U}}_p\left[\frac{4c_p k}{m^2}\right],\, \boldsymbol{z}_n \in \bar{\mathcal{U}}_n\left[\frac{4c_n k}{m^2}\right]. \end{aligned}$$

*The corresponding primal expression is*

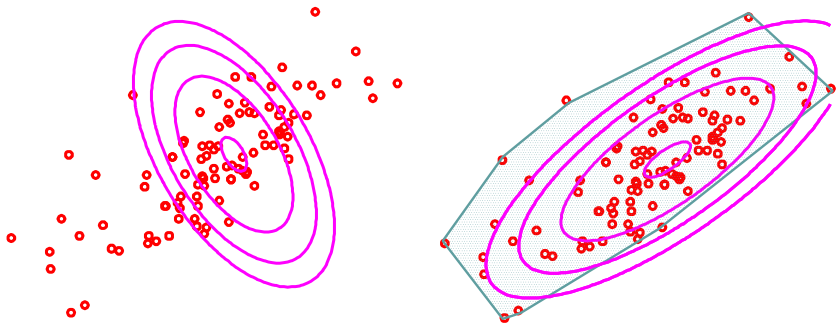
$$\min_{\boldsymbol{w}, b, \rho, \boldsymbol{\xi}} -2\rho + \frac{1}{m} \sum_{i \in M_p} \xi_i^2 \;\; \textit{subject to} \;\; \rho - y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \leq \xi_i,\, 0 \leq \xi_i, \forall i,\; \|\boldsymbol{w}\|^2 \leq \left(\frac{m^2\lambda}{4k}\right)^2.$$

**Example 5** *We define* $h_o^* : \mathcal{X} \to \mathbb{R}$ *for* $o \in \{p,n\}$ *by*

$$h_o^*(\boldsymbol{z}) = (\boldsymbol{z} - \boldsymbol{\mu}_o)^T C_o (\boldsymbol{z} - \boldsymbol{\mu}_o)$$

*where* $\boldsymbol{\mu}_o$ *is the mean vector of the input vector* $\boldsymbol{x}$ *conditioned on each label and* $C_o$ *is a positive definite matrix. In practice, the mean vector is estimated by using prior knowledge which is independent of training samples* $\{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, m\}$. *Suppose that* $\boldsymbol{\mu}_o \neq \mathbf{0}$. *Accordingly, for* $\alpha \geq 0$, *the revision of* (20) *leads to*

$$\begin{aligned} \bar{\ell}^*(\alpha) &= \left((\alpha\frac{m_p}{m} - 1)^2 - 1\right) \boldsymbol{\mu}_p^T C_p \boldsymbol{\mu}_p + \left((\alpha\frac{m_n}{m} - 1)^2 - 1\right) \boldsymbol{\mu}_n^T C_n \boldsymbol{\mu}_n \\ &= b_1 \alpha + b_2 \alpha^2, \end{aligned}$$



original uncertainty set $\mathcal{U}_p[c]$ revised uncertainty set $\bar{\mathcal{U}}_p[c]$

Figure 3: Training samples and uncertainty sets. Left panel: original uncertainty set for the positive label. Right panel: revised uncertainty set consisting of the intersection of ellipsoid and convex-hull of input vectors with the positive label.

*where $b_1$ and $b_2(>0)$ are constant numbers. Thus, we have*

$$\begin{aligned}\bar{\mathcal{U}}_o[c] &= \left\{ \sum_{i\in M_o} \alpha_i \boldsymbol{x}_i \,:\, \sum_{i\in M_o} \alpha_i = 1, \alpha_i \geq 0\,(i\in M_o), \sum_{i\in M_o} \alpha_i^2 \leq \frac{c-b_1}{mb_2} \right\} \\ &= \left\{ \boldsymbol{z} \in \mathrm{conv}\{\boldsymbol{x}_i : i\in M_o\} \,:\, (\boldsymbol{z}-\bar{\boldsymbol{x}}_o)^T \widehat{\Sigma}_o^{-1} (\boldsymbol{z}-\bar{\boldsymbol{x}}_o) \leq m_o \cdot \frac{c-b_1}{mb_2} \right\},\end{aligned}$$

*where $\bar{\boldsymbol{x}}_o$ and $\widehat{\Sigma}_o$ are the estimators of the mean vector and the covariance matrix for $\{\boldsymbol{x}_i : i\in M_o\}$. The corresponding loss function is obtained in the same way as Example 4. Figure 3 illustrates an example of the revision of the uncertainty set. In the left panel, the uncertainty set does not match the distribution of the training samples. On the other hand, the revised uncertainty set in the right panel well approximates the dispersal of the training samples.*

**Example 6** *Suppose that for $o\in\{p,n\}$, $\boldsymbol{\mu}_o$ is the mean vector and $\Sigma_o$ is the covariance matrix of the input vector conditioned on each label. We define the uncertainty set by*

$$o\in\{p,n\}, \quad \mathcal{U}_o[c] = \left\{ z \in \mathrm{conv}\{\boldsymbol{x}_i : i\in M_o\} \,:\, (\boldsymbol{z}-\boldsymbol{\mu})^T \Sigma_o^{-1} (\boldsymbol{z}-\boldsymbol{\mu}) \leq c, \forall \boldsymbol{\mu} \in \mathcal{A} \right\},$$

*where $\mathcal{A}$ denotes the estimation error of the mean vector $\boldsymbol{\mu}$. For a fixed radius $r>0$, $\mathcal{A}$ is defined as*

$$\mathcal{A} = \left\{ \boldsymbol{\mu} \in \mathcal{X} \,:\, (\boldsymbol{\mu}-\boldsymbol{\mu}_o)^T \Sigma_o^{-1} (\boldsymbol{\mu}-\boldsymbol{\mu}_o) \leq r^2 \right\}.$$

*The uncertainty set with the estimation error is used by Lanckriet et al. (2003) in MPM. The above uncertainty set is useful when the probability in the training phase is slightly different from that in the test phase. A brief calculation yields a representation of $\mathcal{U}_o[c]$ in terms of the level set of the convex function,*

$$h_o^*(\boldsymbol{z}) = \max_{\boldsymbol{\mu}\in\mathcal{A}} (\boldsymbol{z}-\boldsymbol{\mu})^T \Sigma_o^{-1} (\boldsymbol{z}-\boldsymbol{\mu}) = \left( \sqrt{(\boldsymbol{z}-\boldsymbol{\mu}_o)^T \Sigma_o^{-1} (\boldsymbol{z}-\boldsymbol{\mu}_o)} + r \right)^2.$$

*The revised uncertainty set $\bar{\mathcal{U}}_o[c]$ is defined by the function $\bar{\ell}^*$:*

$$\bar{\ell}^*(\alpha) = \left(\left|\alpha\frac{m_p}{m} - 1\right|\sqrt{\boldsymbol{\mu}_p^T\Sigma_p^{-1}\boldsymbol{\mu}_p} + r\right)^2 - \left(\sqrt{\boldsymbol{\mu}_p^T\Sigma_p^{-1}\boldsymbol{\mu}_p} + r\right)^2 + \left(\left|\alpha\frac{m_n}{m} - 1\right|\sqrt{\boldsymbol{\mu}_n^T\Sigma_n^{-1}\boldsymbol{\mu}_n} + r\right)^2 - \left(\sqrt{\boldsymbol{\mu}_n^T\Sigma_n^{-1}\boldsymbol{\mu}_n} + r\right)^2. \tag{23}$$

*Suppose that $\boldsymbol{\mu}_p \neq \mathbf{0}$ and $\boldsymbol{\mu}_n = \mathbf{0}$ hold. Let $d = \sqrt{\boldsymbol{\mu}_p^T\Sigma_p^{-1}\boldsymbol{\mu}_p}$ and $h = r/d(>0)$. The corresponding loss function is*

$$\bar{\ell}(z) = \frac{md^2}{m_p}u\big(\frac{z}{d^2}\big),$$

*where $u(z)$ as defined as*

$$u(z) = \begin{cases} 0, & z \leq -2h-2, \\ \big(\frac{z}{2}+1+h\big)^2, & -2h-2 \leq z \leq -2h, \\ z+2h+1, & -2h \leq z \leq 2h, \\ \frac{z^2}{4} + z(1-h) + (1+h)^2, & 2h \leq z. \end{cases} \tag{24}$$

*Figure 4 depicts the function $u(z)$ with $h = 1$. When $r = 0$ holds, $\bar{\ell}(z)$ reduces to the truncated quadratic function shown in Example 4 and 5. For positive r, $\bar{\ell}(z)$ is linear around $z = 0$. This implies that by introducing the confidence set of the mean vector $\mathcal{A}$, the penalty for the misclassification reduces from quadratic to linear around the decision boundary, though the original uncertainty set $\mathcal{U}_o[c]$ does not correspond to minimization of an empirical loss function.*

## 5. Kernel-Based Learning Algorithm Derived from Uncertainty Set

Suppose that we have training samples $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \{+1, -1\}$, where $\mathcal{X}$ is not necessarily a linear space. Let us define a kernel function $k : \mathcal{X}^2 \to \mathbb{R}$, and let $\mathcal{H}$ be the reproducing kernel Hilbert space (RKHS) endowed with the kernel function $k$; see Schölkopf and Smola (2002) for details about the kernel estimators in machine learning.

Let us start with the parametrized uncertainty sets $\mathcal{U}_p[c]$ and $\mathcal{U}_n[c]$ in $\mathcal{H}$. Given uncertainty sets, a kernel variant of (16) is expressed as

$$\begin{aligned} \inf_{c_p, c_n, f_p, f_n} \; & c_p + c_n + \lambda\|f_p - f_n\|_{\mathcal{H}} \\ \text{subject to } \; & c_p, c_n \in \mathbb{R}, \\ & f_p \in \mathcal{U}_p[c_p] \cap \operatorname{conv}\{k(\cdot, x_i) : i \in M_p\}, \\ & f_n \in \mathcal{U}_n[c_n] \cap \operatorname{conv}\{k(\cdot, x_j) : j \in M_n\}. \end{aligned} \tag{25}$$

Next, we find the corresponding loss function $\ell(z)$. Note that the revision of uncertainty sets presented in Section 4 can be used, if necessary. Suppose the uncertainty sets are represented as

$$\mathcal{U}_o[c] = \left\{ \sum_{i \in M_o} \alpha_i k(\cdot, x_i) \in \mathcal{H} : \frac{1}{m}\sum_{i \in M_o} \ell^*(m\alpha_i) \leq c \right\} \tag{26}$$

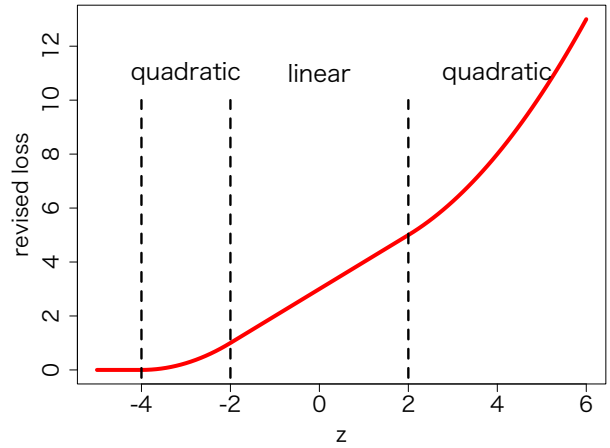



Figure 4: Loss function $u(z)$ in Example 6 that corresponds to the revised uncertainty set with the estimation error.

for $o \in \{p,n\}$. In the same way as in Section 3.1, we find that problem (25) is the dual representation of

$$\begin{aligned} &\min_{f,b,\rho} -2\rho + \frac{1}{m}\sum_{i=1}^{m} \ell(\rho - y_i(f(x_i)+b)) \\ &\text{subject to } \ f \in \mathcal{H}, b \in \mathbb{R}, \rho \in \mathbb{R}, \|f\|_{\mathcal{H}}^2 \le \lambda^2. \end{aligned} \tag{27}$$

We can obtain the estimated decision function $\widehat{f}+\widehat{b} \in \mathcal{H}+\mathbb{R}$ by solving the problem (27). A rigorous proof of the strong duality between (25) and (27) is presented in Section 6 and Appendix B.

**Example 7 (ellipsoidal uncertainty sets in RKHS)** *Let us consider an uncertainty set $\mathcal{U}[c]$ in RKHS $\mathcal{H}$ defined by*

$$\mathcal{U}[c] = \left\{ \sum_{i=1}^{m} \alpha_i k(\cdot, x_i) \,:\, \sum_{i=1}^{m} \alpha_i^2 \le c \right\} \subset \mathcal{H},$$

*where $x_1,\ldots,x_m$ are points in $\mathcal{X}$. The corresponding loss is the truncated quadratic loss. Let us define $\bar{k} \in \mathcal{H}$ as $\frac{1}{m}\sum_{i=1}^{m} k(\cdot,x_i)$. Furthermore, let us define the empirical variance operator $\widehat{\Sigma} : \mathcal{H} \to \mathcal{H}$ as*

$$\widehat{\Sigma} h = \frac{1}{m}\sum_{i=1}^{m} (k(\cdot,x_i) - \bar{k}) \langle k(\cdot,x_i) - \bar{k}, h \rangle_{\mathcal{H}}$$

*for $h \in \mathcal{H}$. Some calculation yields*

$$\begin{aligned} &\mathcal{U}[c] \cap \mathrm{conv}\{k(\cdot,x_i) \,:\, i = 1,\ldots,m\} \\ &= \left\{ \bar{k} + \widehat{\Sigma} h \,:\, \langle \widehat{\Sigma} h, h \rangle_{\mathcal{H}} \le mc - 1 \right\} \cap \mathrm{conv}\{k(\cdot,x_i) \,:\, i = 1,\ldots,m\}. \end{aligned}$$

*This is the kernel variant of the ellipsoidal uncertainty set in Example 2.*

By transforming the uncertainty-set-based learning into loss-based learning, we can obtain a statistical model for the conditional probability, as shown in Section 3.2. In addition, we can verify the statistical consistency of the learning algorithm with the (revised) uncertainty sets by taking the corresponding loss function into account. Other authors have proposed kernel-based learning algorithms with uncertainty sets (Lanckriet et al., 2003; Huang et al., 2004), but they did not deal with the issue of statistical consistency. In the following, we study the statistical properties of the learning algorithm based on (27).

## 6. Statistical Properties of Kernel-Based Learning Algorithms

Here, we prove that the expected 0-1 loss $\mathcal{E}(\widehat{f}+\widehat{b})$ converges to the Bayes risk $\mathcal{E}^*$ defined by (1). We also determine whether certain popular uncertainty sets produce consistent learning methods. All proofs are presented in Appendix B and Appendix C.

### 6.1 Assumptions for Statistical Consistency

Let us show four assumptions.

**Assumption 1 (universal kernel)** *The input space $\mathcal{X}$ is a compact metric space. The kernel function $k : \mathcal{X}^2 \to \mathbb{R}$ is continuous, and satisfies*

$$\sup_{x\in\mathcal{X}} \sqrt{k(x,x)} \leq K < \infty,$$

*where $K$ is a positive constant. In addition, $k$ is universal, that is, the RKHS associated with $k$ is dense in the set of all continuous functions on $\mathcal{X}$ with respect to the supremum norm (Steinwart and Christmann, 2008, Definition 4.52).*

**Assumption 2 (non-deterministic assumption)** *For the probability distribution of training samples, there exists a positive constant $\varepsilon > 0$ such that*

$$P(\{x \in \mathcal{X} : \varepsilon \leq P(+1|x) \leq 1-\varepsilon\}) > 0,$$

*where $P(y|x)$ is the conditional probability of the label $y$ for the input $x$.*

**Assumption 3 (basic assumptions on loss functions)** *The loss function $\ell : \mathbb{R} \to \mathbb{R}$ satisfies the following conditions.*

1. *$\ell$ is a non-decreasing, convex function that is non-negative, that is, $\ell(z) \geq 0$ for all $z \in \mathbb{R}$.*
2. *Let $\partial\ell(z)$ be the subdifferential of the loss function $\ell$ at $z \in \mathbb{R}$ (Rockafellar, 1970, Chapter 23). Then, the equality $\lim_{z\to\infty} \partial\ell(z) = \infty$ holds; that is, for any $M > 0$, there exists $z_0$ such that $g \geq M$ for all $z \geq z_0$ and all $g \in \partial\ell(z)$.*

Note that the second condition in Assumption 3 assures that $\ell$ is not a constant function and that $\lim_{z\to\infty} \ell(z) = \infty$.

**Assumption 4 (modified classification-calibrated loss)**

1. *$\ell(z)$ is first order differentiable for $z \geq -\ell(0)/2$, and $\ell'(z) > 0$ for $z \geq -\ell(0)/2$, where $\ell'$ is the derivative of $\ell$.*

2. *Let $\psi(\theta,\rho)$ be the function*

$$\psi(\theta,\rho) = \ell(\rho) - \inf_{z\in\mathbb{R}} \left\{ \frac{1+\theta}{2}\ell(\rho - z) + \frac{1-\theta}{2}\ell(\rho + z) \right\}, \quad 0 \leq \theta \leq 1,\ \rho \in \mathbb{R}.$$

*There exists a function $\widetilde{\psi}(\theta)$ and a positive real $\varepsilon > 0$ such that the following three conditions are satisfied:*

   *(a) $\widetilde{\psi}(0) = 0$ and $\widetilde{\psi}(\theta) > 0$ for $0 < \theta \leq \varepsilon$.*

   *(b) $\widetilde{\psi}(\theta)$ is a continuous and strictly increasing function on the interval $[0,\varepsilon]$.*

   *(c) The inequality $\widetilde{\psi}(\theta) \leq \inf_{\rho \geq -\ell(0)/2} \psi(\theta,\rho)$ holds for $0 \leq \theta \leq \varepsilon$.*

Appendix B presents a rigorous proof of the duality between (27) and (25) with the uncertainty set (26). Appendix C.3 presents sufficient conditions for the existence of the function $\widetilde{\psi}$ in Assumption 4.

Under Assumptions 1–4 and another mild assumption, we prove that the expected 0-1 loss $\mathcal{E}(\widehat{f} + \widehat{b})$ converges to the Bayes risk $\mathcal{E}^*$. In the mild assumption, the covering number of the RKHS $\mathcal{H}$ is taken into account. The details of the conferring number are shown in Appendix C.1.

#### 6.1.1 THEOREM (STATISTICAL CONSISTENCY)

*For the RKHS $\mathcal{H}$ and the loss function $\ell$, we assume Assumptions 1, 2, 3 and 4. Also, we assume that $\mathcal{H}$ satisfies the covering number condition, that is,* (41) *in Appendix C.1 converges to zero for any positive $\varepsilon$, when the sample size $m$ tends to infinity. Then, $\mathcal{E}(\widehat{f} + \widehat{b})$ converges to $\mathcal{E}^*$ in probability.*

Appendix C presents the necessary definitions, lemmas, and theorems, and Theorem 8 of Appendix C.1 and Theorem 9 of Appendix C.2 summarize the main results. The examples presented in Appendix C.3 show that some popular uncertainty sets and their revisions yield loss functions satisfying the above sufficient conditions.

### 6.2 Supplementary Explanations

Let us discuss Assumptions 1–4.

**Universal kernel:** The universality of RKHSs in Assumption 1 is usually assumed, when discussing the statistical consistency of kernel methods. If the RKHS under consideration is not universal, a decision function might exist that is not approximated well by any element in the RKHS. The Gaussian kernel is universal, while the polynomial kernel is not universal.

**Non-deterministic assumption:** In Assumption 2, the label $y$ is assigned in a non-deterministic way. The label assignment is deterministic when the conditional probability $P(y = +1|x)$ is equal to 0 or 1 for all $x$. Steinwart (2005) introduced the $y$-degenerated condition defined as

$$P(\{x \in \mathcal{X} : P(y|x) = 1\}) = 1$$

for a label $y \in \{+1, -1\}$. If the $y$-degenerated condition holds, the proof of the consistency is straightforward for standard learning methods such as $C$-SVM. Involved mathematical arguments are needed to prove consistency if the $y$-degenerated condition does not apply. Here, the deterministic assumption means

$$P(\{x \in \mathcal{X} : P(y|x) = 1 \text{ or } 0\}) = 1.$$

In our setup, the parameter $\rho$ is a variable, and it makes the situation somewhat difficult. Under the above deterministic assumption, the optimal value of (27) may go to $-\infty$, as the number of training samples tends to infinity; see Lemma 3 in Appendix C.1. In such a case, it would be impossible to make an empirical approximation of the objective value in (27). We introduced the non-deterministic assumption to avoid such a troublesome situation.

**Basic assumptions on loss functions:** Loss functions are based on Assumption 3 and Assumption 4. Conditions such that $\ell(z)$ is convex, non-decreasing, and bounded from below are standard ones, but the second condition in Assumption 3 is rather strong. The hinge loss and logistic loss do not satisfy this assumption, whereas the quadratic loss and exponential loss satisfy it. Assumption 3 is used to derive an upper bound of the optimal $\rho$ in (27); see Lemma 5 in Appendix C.1.

**Modified classification-calibrated loss:** Assumption 4 is related to the classification-calibrated loss. Bartlett et al. (2006) introduced classification-calibrated losses to analyze the statistical consistency of binary classification problems. Roughly speaking, if a loss function is classification-calibrated, the minimizer of the loss function produces the minimizer of the 0-1 loss. See Bartlett et al. (2006) for details about classification-calibrated losses. Suppose that the function $\ell(\rho - z)$ with a fixed $\rho$ is convex in $z \in \mathbb{R}$. Then, a sufficient condition for $\ell(\rho - z)$ to be a classification-calibrated loss is given as $\ell'(\rho) > 0$; that is, $\ell$ is differentiable at $\rho$ and the derivative is positive. In our setup, $\rho$ is variable, and hence the condition $\ell'(\rho) > 0$ is required for all possible values of $\rho$. As shown in the proof of Lemma 5 in Appendix C.1, the optimal $\rho$ of the problem (27) is bounded from below by $-\ell(0)/2$. Thus, we assumed the differentiability of $\ell(z)$ for $z \geq -\ell(0)/2$. The second condition of Assumption 4 defines the functions, $\psi(\theta, \rho)$ and $\widetilde{\psi}(\theta)$. Bartlett et al. (2006) defined the function $\psi(\theta, 0)$ and derived the quantitative relation between the classification calibrated loss and the 0-1 loss via $\psi(\theta, 0)$. We extended $\psi(\theta, 0)$ to $\psi(\theta, \rho)$ having a variable $\rho$. The functions $\psi(\theta, \rho)$ and $\widetilde{\psi}(\theta)$ describe a qualitative relation between the convex loss $\ell$ and the 0-1 loss. Appendix C.2 uses the function $\widetilde{\psi}(\theta)$ to prove that the convergence of the expected loss guarantees the convergence of the expected 0-1 loss to the Bayes risk.

Appendix C.3 describes the sufficient conditions for the existence of the function $\widetilde{\psi}(\theta)$ in Assumption 4. It shows some simple conditions under which a given loss function $\ell(z)$ will possess $\widetilde{\psi}(\theta)$. As a result, it is shown that the existence of $\widetilde{\psi}$ is guaranteed for the truncated quadratic loss, exponential loss and the loss function derived from the uncertainty set with the estimation error in Example 6; see the examples provide in Appendix C.3.

## 7. Experiments

We conducted some numerical experiments to examine the prediction performance of our revision of uncertainty sets methods. The results indicate that the method improves the estimator. In addition, we evaluated the estimation accuracy of the conditional probability.

We compared the kernel-based learning algorithms using the Gaussian kernel. So far, many studies have compared linear models and kernel-based models. The conclusion is that linear models outperform kernel-based models when the linear models have good approximations of the decision boundary. Otherwise, linear models have an approximation bias, and kernel-based estimators with a nice regularization outperform linear models. For this reason, we focused on kernel-based estimators.

The following methods were examined using the synthetic data and the standard benchmark data sets: $C$-SVM, MPM, unbiased MPM, and the learning method with (27). $C$-SVM is the one implemented in the `kernlab` library (Karatzoglou et al., 2004). In the unbiased MPM, the bias term $b$ of the model was estimated by minimizing the training error rate after estimating the function part, $\widehat{f} \in \mathcal{H}$. The unbiased estimator will outperform the original MPM when the probability of the class label is heavily unbalanced. The loss function $\ell(z)$ of the proposed method was the function $u(z)$ in (24). This loss function corresponds to the revised uncertainty set of the ellipsoidal uncertainty set with the estimation error. The parameter in the function $u(z)$ of (24) was set to $h = 0$ or $h = 1$. The kernel parameter and the regularization parameter were estimated by 5-fold cross validation.

We evaluated the learning results as follows. We used the test error over the test samples to evaluate the classification accuracy. We assessed the estimation accuracies of the conditional probabilities given by $C$-SVM and the proposed method. A $C$-SVM with such a probability estimation is included in the `kernlab` library; the probability model is shown in Karatzoglou et al. (2004). We used the squared loss to assess the estimation accuracy of the conditional probability:

$$\mathbb{E}\Big[\sum_{y=\pm 1}(\widehat{P}(y|x)-P(y|x))^2\Big] = \mathbb{E}\big[\widehat{P}(+1|x)^2+\widehat{P}(-1|x)^2\big] - 2\mathbb{E}\big[\widehat{P}(y|x)\big] + \mathbb{E}\big[P(+1|x)^2+P(-1|x)^2\big],$$

where $\widehat{P}(y|x)$ is an estimator of the true conditional probability $P(y|x)$. Since the last term of the above expression does not depend on the estimator, we used only the first two terms as the measure of estimation accuracy. As a result, given test samples $\{(\widetilde{x}_i,\widetilde{y}_i) : \ell = 1,\ldots,L\}$, the estimated conditional probability $\widehat{P}(y|x)$ can be approximately evaluated as follows:

$$\text{squared-loss} = \frac{1}{L}\sum_{\ell=1}^{L}\big[\widehat{P}(+1|\widetilde{x}_\ell)^2+\widehat{P}(-1|\widetilde{x}_\ell)^2\big] - \frac{2}{L}\sum_{\ell=1}^{L}\widehat{P}(\widetilde{y}_\ell|\widetilde{x}_\ell).$$

This measure works even for benchmark data sets in which the true probability is unknown. Note that the squared-loss above can take negative values, since the last term in the expansion of $\mathbb{E}\big[\sum_{y=\pm 1}(\widehat{P}(y|x)-P(y|x))^2\big]$ is not taken into account. We did not use the Kullback-Leibler divergence or logarithmic loss, since the estimator $\widehat{P}(y|x)$ can take zero.

### 7.1 Synthetic Data

The input points conditioned on the positive label were generated from a two dimensional normal distribution with mean $\boldsymbol{\mu}_p = (0,0)^T$ and variance-covariance matrix $\Sigma_p = I$, where $I$ is the identity

| $P(y=+1)$ | $C$-SVM | MPM | unbiased MPM | $h=0$ | $h=1$ |
|---|---|---|---|---|---|
| 0.2 | $15.81 \pm 1.17$ | $25.63 \pm 2.27$ | $16.51 \pm 1.48$ | $15.39 \pm 0.99$ | $15.37 \pm 0.93$ |
| 0.5 | $25.32 \pm 1.49$ | $25.33 \pm 1.64$ | $25.50 \pm 1.47$ | $24.81 \pm 1.18$ | $24.89 \pm 1.27$ |

Table 1: Test error (%) and standard deviation of each learning method. We compared $C$-SVM, MPM, unbiased MPM, and the proposed learning method using the loss function (24) with $h=0$ or $h=1$.

matrix. The conditional distribution of the input points with the negative label was a normal distribution with mean $\boldsymbol{\mu}_n = (1,1)^T$ and variance-covariance matrix $\Sigma_n = R^T \mathrm{diag}(0.5^2, 1.5^2) R$, where $R$ is the $\pi/3$ radian counterclockwise rotation matrix. The label probability was $P(y=+1) = 0.2$ or 0.5. The size of the training samples was $m = 400$. We computed test errors by averaging over 100 iterations. For $C$-SVM and the proposed method, we computed the average squared-loss of the estimated conditional probability. We also evaluated average absolute difference between the true conditional probability $P(+1|\boldsymbol{x})$ and the estimator $\widehat{P}(+1|\boldsymbol{x})$ on the test set, that is, the average of $\frac{1}{L}\sum_{\ell=1}^{L} |P(+1|\widetilde{\boldsymbol{x}}_\ell) - \widehat{P}(+1|\widetilde{\boldsymbol{x}}_\ell)|$ over 100 iterations. This is possible, since the true probability of synthetic data is known.

Table 1 shows the test errors of $C$-SVM, MPM, unbiased MPM, and the proposed method using the loss function (24) with $h=0$ or $h=1$. The table shows that the MPM has an estimation bias for unbalanced samples, that is, the case of $P(y=+1) = 0.2$. MPM is slightly better than unbiased MPM on the setup of the balanced data. Overall, the proposed method is better than the other learning methods. Indeed, the difference between it and $C$-SVM is statistically significant. On the other hand, the parameter $h$ in the loss function (24) does not significantly affect the experimental results.

Table 2 shows the accuracy of the estimated conditional probabilities measured by the squared loss and absolute difference. As shown in the lower table, the absolute error of the proposed method is about 5%, while the error of $C$-SVM is about 10%. The proposed method also outperforms $C$-SVM in terms of the squared-loss. $C$-SVM and the proposed method differ significantly in their estimation accuracy of the conditional probability, though the difference in classification error rate is less than 0.5%. Figure 5 presents the squared loss and absolute loss of the estimated conditional probability versus the size of the training samples for $C$-SVM and the proposed method with $h=0$ and $h=1$. The proposed method outperforms $C$-SVM. For each sample size, the parameter $h$ does not significantly affect the estimation accuracy, though the loss function $u(z)$ with $h=1$ is consistently slightly better than $h=0$.

### 7.2 Benchmark Data

The experiments used thirteen artificial and real-world data sets from the UCI, DELVE, and STATLOG benchmark repositories: `banana`, `breast-cancer`, `diabetes`, `german`, `heart`, `image`, `ringnorm`, `flare-solar`, `splice`, `thyroid`, `titanic`, `twonorm`, and `waveform`. All data sets are in the IDA benchmark repository. See Rätsch et al. (2001) and Rätsch et al. (2000) for details about the data sets. The properties of each data set are shown in Table 3, where "dim", "$P(y=+1)$","#train", "#test" and "rep." respectively denote the input dimension, the ratio of the positive labels in training samples, the size of training set, the size of test set, and the number of

| | squared-loss $\times 100$ | | |
|---|---|---|---|
| $P(y=+1)$ | $C$-SVM | $h=0$ | $h=1$ |
| 0.2 | $-75.43 \pm 1.78$ | $-77.36 \pm 1.07$ | $-77.42 \pm 1.08$ |
| 0.5 | $-65.63 \pm 1.89$ | $-67.90 \pm 1.07$ | $-67.83 \pm 1.21$ |

| | absolute difference (%) between $P(+1\|\boldsymbol{x})$ and $\widehat{P}(+1\|\boldsymbol{x})$ | | |
|---|---|---|---|
| $P(y=+1)$ | $C$-SVM | $h=0$ | $h=1$ |
| 0.2 | $9.37 \pm 1.96$ | $4.57 \pm 1.28$ | $4.43 \pm 1.14$ |
| 0.5 | $10.10 \pm 2.34$ | $5.11 \pm 1.10$ | $5.19 \pm 1.32$ |

Table 2: Squared loss and absolute loss of the estimated conditional probability $\widehat{P}(y|x)$. We compared $C$-SVM and the proposed method using the loss function (24) with $h=0$ and $h=1$.

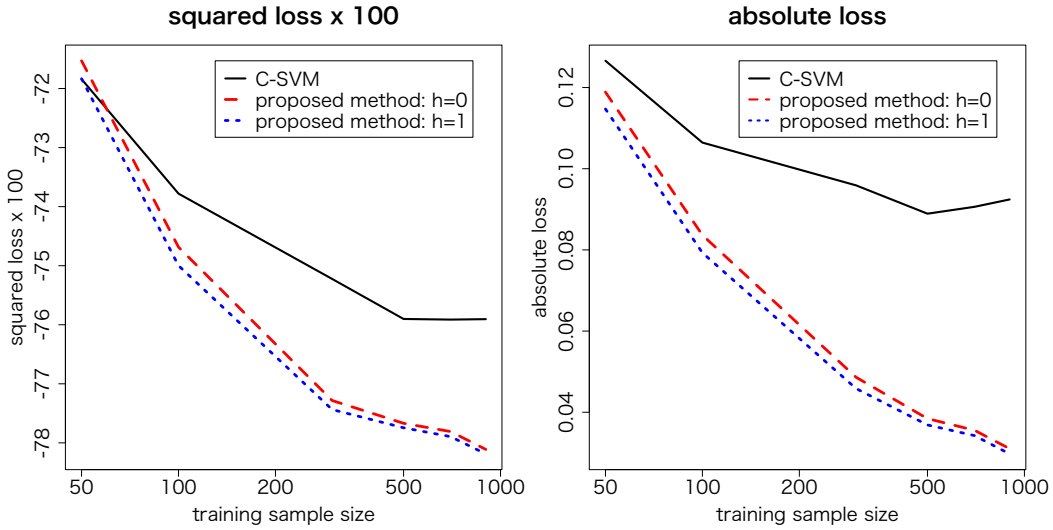



Figure 5: Squared loss and absolute loss of estimated conditional probability versus training sample size are presented for $C$-SVM and the proposed method with $h=0$ and $h=1$.

replications of learning to evaluate the average performance. Table 4 shows the test errors (%) and the standard deviation for the benchmark data sets.

First, we compared MPM, unbiased MPM and the proposed method with "$h=0$". The uncertainty set of MPM and unbiased MPM is an ellipsoid defined by the estimated covariance matrix. The corresponding loss function of the form of (9) does not exist, since the convex-hull of the input points is not taken into account. The uncertainty set of the proposed method with "$h=0$" is the intersection of an ellipsoid and the convex-hull of the input vectors. The revision of the ellipsoidal uncertainty set leads to the uncertainty set of our algorithm. The proposed method with "$h=0$" outperforms MPM and unbiased MPM for most data sets. Hence, the revision of uncertainty sets can improve the prediction accuracy of uncertainty-set-based learning.

| data set | dim | $P(y=+1)$ | #train | #test | rep. |
|---|---|---|---|---|---|
| banana | 2 | 0.454 | 400 | 4900 | 100 |
| breast-cancer | 9 | 0.294 | 200 | 77 | 100 |
| diabetis | 8 | 0.350 | 468 | 300 | 100 |
| flare-solar | 9 | 0.552 | 666 | 400 | 100 |
| german | 20 | 0.301 | 700 | 300 | 100 |
| heart | 13 | 0.445 | 170 | 100 | 100 |
| image | 18 | 0.574 | 1300 | 1010 | 20 |
| ringnorm | 20 | 0.497 | 400 | 7000 | 100 |
| splice | 60 | 0.483 | 1000 | 2175 | 20 |
| thyroid | 5 | 0.305 | 140 | 75 | 80 |
| titanic | 3 | 0.322 | 150 | 2051 | 100 |
| twonorm | 20 | 0.505 | 400 | 7000 | 100 |
| waveform | 21 | 0.331 | 400 | 4600 | 100 |

Table 3: The properties of each data sets: "dim", "$P(y=+1)$","#train", "#test" and "rep." respectively denote the input dimension, the ratio of the positive label in the training samples, the size of the training set, the size of the test set, and the number of replications of learning.

The boldface letters in Table 4 indicate the smallest average test error for each data set. Overall, $C$-SVM and the learning method "$h=1$" outperform the others. $C$-SVM is significantly better than the proposed method with "$h=1$" on `flare-solar`, `ringnorm` and `twonorm`, but the proposed method with "$h=1$" is significantly better than $C$-SVM on `banana`, `diabetis`, `german` and `waveform`. These results show that the proposed method with "$h=1$" is comparable to $C$-SVM. Table 5 shows the squared-losses for estimated conditional probabilities. It shows that the proposed method with "$h=1$" outperforms the others in the conditional probability estimation.

In Section 6, we proved the statistical consistency of learning methods derived from the uncertainty set approach. The numerical experiments described in this section indicate that learning methods derived from revised uncertainty sets are an alternative for solving classification problems involving conditional probability estimations.

## 8. Conclusion

We studied the relation between the loss function approach and the uncertainty set approach in binary classification problems. We showed that these two approaches are connected via the convex conjugate of the loss function. Given a loss function, there exists a corresponding parametrized uncertainty set. In general, however, the uncertainty set does not correspond to the empirical loss function. We presented a way of revising the uncertainty set so that it will correspond to an empirical loss function. On the basis of this revision, we proposed a kernel-based learning algorithm and proved statistical consistency. The way to estimate the conditional probability was also proposed. Numerical experiments showed that learning methods derived from revised uncertainty sets are alternatives means for solving classification problems involving conditional probability estimation.

Some problems remains with our methodology. The proof of the statistical consistency does not include the hinge loss used in $\nu$-SVM. Steinwart (2003) proved that $\nu$-SVM is statistically consistent with a nice choice of the regularization parameter. However, such a regularization parameter heavily depends on the true probability distribution; that is, the parameter $\nu$ should be twice the Bayes error

| | test error (%) | | | | |
|---|---|---|---|---|---|
| data set | $C$-SVM | MPM | unbiased MPM | $h=0$ | $h=1$ |
| banana | $10.74 \pm 0.60$ | $11.35 \pm 0.87$ | $11.49 \pm 0.93$ | $10.47 \pm 0.48$ | $\mathbf{10.45 \pm 0.47}$ |
| breast-cancer | $26.96 \pm 4.57$ | $34.77 \pm 4.53$ | $33.26 \pm 5.01$ | $\mathbf{26.60 \pm 4.64}$ | $26.77 \pm 4.60$ |
| diabetis | $23.94 \pm 2.05$ | $28.81 \pm 2.61$ | $28.42 \pm 2.46$ | $\mathbf{23.30 \pm 1.85}$ | $23.36 \pm 1.78$ |
| flare-solar | $\mathbf{33.71 \pm 2.11}$ | $34.92 \pm 1.73$ | $35.62 \pm 1.83$ | $34.09 \pm 1.65$ | $34.14 \pm 1.86$ |
| german | $23.84 \pm 2.32$ | $29.17 \pm 2.43$ | $28.53 \pm 2.58$ | $23.54 \pm 2.20$ | $\mathbf{23.28 \pm 2.08}$ |
| heart | $\mathbf{16.56 \pm 3.51}$ | $25.41 \pm 4.34$ | $25.82 \pm 4.17$ | $16.62 \pm 3.48$ | $16.70 \pm 3.17$ |
| image | $3.17 \pm 0.66$ | $\mathbf{3.11 \pm 0.58}$ | $3.30 \pm 0.73$ | $3.20 \pm 0.67$ | $3.21 \pm 0.62$ |
| ringnorm | $\mathbf{1.73 \pm 0.27}$ | $3.21 \pm 0.49$ | $2.81 \pm 0.38$ | $2.02 \pm 0.25$ | $2.01 \pm 0.24$ |
| splice | $\mathbf{11.03 \pm 0.74}$ | $12.25 \pm 1.71$ | $11.74 \pm 0.89$ | $11.10 \pm 0.72$ | $11.07 \pm 0.64$ |
| thyroid | $5.25 \pm 2.10$ | $6.58 \pm 2.96$ | $6.83 \pm 3.23$ | $5.27 \pm 2.16$ | $\mathbf{5.08 \pm 2.23}$ |
| titanic | $\mathbf{22.47 \pm 0.81}$ | $24.27 \pm 2.60$ | $22.47 \pm 1.23$ | $22.59 \pm 1.37$ | $22.62 \pm 1.36$ |
| twonorm | $\mathbf{2.67 \pm 0.41}$ | $4.50 \pm 0.65$ | $4.47 \pm 0.66$ | $2.98 \pm 0.31$ | $2.97 \pm 0.30$ |
| waveform | $10.22 \pm 0.68$ | $12.90 \pm 0.79$ | $12.73 \pm 0.94$ | $10.00 \pm 0.50$ | $\mathbf{9.96 \pm 0.44}$ |

Table 4: Test errors (%) and the standard deviation for benchmark data sets. We compared $C$-SVM, MPM, unbiased MPM, and the proposed method with loss functions (24) having $h = 0$ and $h = 1$: boldface letters indicate that the average squared loss is the smallest.

| | squared loss $\times 100$ | | |
|---|---|---|---|
| data set | $C$-SVM | $h=0$ | $h=1$ |
| banana | $-83.99 \pm 0.90$ | $-84.98 \pm 0.53$ | $\mathbf{-85.15 \pm 0.50}$ |
| breast-cancer | $-62.60 \pm 4.15$ | $-64.04 \pm 3.88$ | $\mathbf{-64.11 \pm 4.09}$ |
| diabetis | $-67.51 \pm 2.01$ | $\mathbf{-68.31 \pm 1.39}$ | $-68.13 \pm 1.52$ |
| flare-solar | $-56.52 \pm 1.67$ | $-59.69 \pm 1.10$ | $\mathbf{-59.79 \pm 1.15}$ |
| german | $-67.29 \pm 2.20$ | $-67.69 \pm 1.99$ | $\mathbf{-67.98 \pm 2.09}$ |
| heart | $\mathbf{-74.75 \pm 4.02}$ | $-74.14 \pm 3.23$ | $-74.59 \pm 3.37$ |
| image | $-94.71 \pm 0.81$ | $-94.67 \pm 0.68$ | $\mathbf{-94.72 \pm 0.72}$ |
| ringnorm | $\mathbf{-97.30 \pm 0.75}$ | $-96.27 \pm 0.27$ | $-96.27 \pm 0.29$ |
| splice | $\mathbf{-83.88 \pm 0.75}$ | $-83.30 \pm 0.57$ | $-83.32 \pm 0.54$ |
| thyroid | $-92.26 \pm 3.12$ | $-92.94 \pm 2.48$ | $\mathbf{-93.11 \pm 2.53}$ |
| titanic | $-65.06 \pm 1.10$ | $\mathbf{-66.26 \pm 1.28}$ | $-66.18 \pm 1.40$ |
| twonorm | $\mathbf{-95.92 \pm 0.60}$ | $-95.07 \pm 0.36$ | $-95.11 \pm 0.35$ |
| waveform | $-85.50 \pm 0.89$ | $-85.39 \pm 0.53$ | $\mathbf{-85.57 \pm 0.48}$ |

Table 5: Squared loss $\times 100$ of estimated conditional probability for $C$-SVM and the proposed method with loss functions (24) having $h = 0$ and $h = 1$: boldface letters indicate that the average test error is the smallest.

that cannot be obtained before the learning. We are currently investigating of the possibility of relaxing the assumptions so as to include the hinge loss and other popular loss functions such as the logistic loss. We focused on binary classification problems in this paper. An interesting direction of research is to extend the relation between loss-based learning and uncertainty-set-based learning to more general statistical problems such as ranking problems and multiclass classification problems. The statistical consistency of more general problem setups is an ongoing research topic, and we expect that the duality based on the convex conjugate can be used to devise a new approach to these problems.

The relation between the loss function approach and the uncertainty set approach is a useful tool for statistical modeling. In optimization and control theory, the modeling based on the uncertainty set is frequently applied to the real-world data; the reader may consult the modeling used in robust optimization and related work (Ben-Tal and Nemirovski, 2002). We believe that learning algorithms with revised uncertainty sets can bridge the gap between intuitive statistical modeling and nice statistical properties.

## Acknowledgments

The authors are grateful to anonymous reviewers for their helpful comments. TK was partially supported by JSPS KAKENHI Grant Number 24500340. AT was partially supported by Grant-in-Aid for Young Scientists (23710174). TS was partially supported by MEXT Kakenhi 22700289 and the Aihara Project, the FIRST program from JSPS, initiated by CSTP. An earlier and shorter version of this paper appeared in the proceedings of COLT2012, and the present version has benefited from comments from the COLT referees.

## Appendix A. Preliminaries on Convex Conjugates

A convex conjugate is a standard tool in convex analysis. The convex conjugate is also referred to as a Legendre transformation. We show a brief introduction of the convex conjugate. See Rockafellar (1970) for details.

Let $\ell : \mathbb{R}^k \to \mathbb{R}$ be a convex function. The *convex conjugate* $\ell^* : \mathbb{R}^k \to \mathbb{R} \cup \{+\infty\}$ of the function $\ell$ is defined by

$$\ell^*(\boldsymbol{\alpha}) = \sup_{\boldsymbol{z} \in \mathbb{R}^k} \boldsymbol{z}^T \boldsymbol{\alpha} - \ell(\boldsymbol{z}).$$

Note that $\ell^*(\boldsymbol{\alpha}) = +\infty$ can occur. Under a mild assumption, the equality $(\ell^*)^* = \ell$ holds.

Suppose that the function $\ell(\boldsymbol{z})$ is decomposed into

$$\ell(\boldsymbol{z}) = \ell_1(\boldsymbol{z}_1) + \ell_2(\boldsymbol{z}_2), \quad \boldsymbol{z} = (\boldsymbol{z}_1, \boldsymbol{z}_2) \in \mathbb{R}^k.$$

Then, the convex conjugate of $\ell(\boldsymbol{z})$ is the sum of convex conjugates of $\ell_1$ and $\ell_2$. Indeed,

$$\begin{aligned}
\ell^*(\boldsymbol{\alpha}) &= \sup_{\boldsymbol{z}} \boldsymbol{z}^T \boldsymbol{\alpha} - \ell_1(\boldsymbol{z}_1) - \ell_2(\boldsymbol{z}_2) \\
&= \sup_{\boldsymbol{z}_1, \boldsymbol{z}_2} \boldsymbol{z}_1^T \boldsymbol{\alpha}_1 + \boldsymbol{z}_2^T \boldsymbol{\alpha}_2 - \ell_1(\boldsymbol{z}_1) - \ell_2(\boldsymbol{z}_2) \\
&= \ell_1^*(\boldsymbol{\alpha}_1) + \ell_2^*(\boldsymbol{\alpha}_2), \quad \boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2) \in \mathbb{R}^k.
\end{aligned}$$

The formula above is used in Section 4.2.

## Appendix B. Proof of Strong Duality between (25) and (27)

We prove that there is no duality gap between (25) and (27).

**Lemma 2** *Suppose that both $M_p = \{i : y_i = +1\}$ and $M_n = \{i : y_i = -1\}$ are non-empty, that is, $m_p = |M_p|$ and $m_n = |M_n|$ are positive numbers. Under Assumption 1 and 3 in Section 6, there exists an optimal solution for* (27)*. Moreover, the dual problem of* (27) *yields the problem* (25) *with the uncertainty set* (26)*.*

**Proof** First, we prove the existence of an optimal solution. According to the standard argument on the kernel estimator, we can restrict the function part $f$ to be the form of

$$f(x) = \sum_{j=1}^{m} \alpha_j k(x, x_j).$$

Then, the problem is reduced to the finite-dimensional problem,

$$\begin{aligned} &\min_{\boldsymbol{\alpha}, b, \rho} -2\rho + \frac{1}{m}\sum_{i=1}^{m} \ell(\rho - y_i(\sum_{j=1}^{m} \alpha_j k(x_i, x_j) + b)) \\ &\text{subject to } \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \le \lambda^2. \end{aligned} \tag{28}$$

Let $\zeta_0(\boldsymbol{\alpha}, b, \rho)$ be the objective function of (28). Let us define $\mathcal{S}$ be the linear subspace in $\mathbb{R}^m$ spanned by the column vectors of the gram matrix $(k(x_i, x_j))_{i,j=1}^m$. We can impose the constraint $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \in \mathcal{S}$, since the orthogonal complement of $\mathcal{S}$ does not affect the objective function and the constraint in (28). We see that Assumption 1 and the reproducing property yield the inequality $\|y_i \sum_{j=1}^m \alpha_j k(\cdot, x_j)\|_\infty \le K\lambda$. Due to this inequality and the assumptions on the function $\ell$, the objective function $\zeta_0(\boldsymbol{\alpha}, b, \rho)$ is bounded below by

$$\zeta_1(b, \rho) = -2\rho + \frac{m_p}{m}\ell(\rho - b - K\lambda) + \frac{m_n}{m}\ell(\rho + b - K\lambda).$$

Hence, for any real number $c$, the inclusion relation

$$\begin{aligned} &\left\{ (\boldsymbol{\alpha}, b, \rho) \in \mathbb{R}^{m+2} \ : \ \zeta_0(\boldsymbol{\alpha}, b, \rho) \le c, \ \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \le \lambda^2, \boldsymbol{\alpha} \in \mathcal{S} \right\} \\ &\subset \left\{ (\boldsymbol{\alpha}, b, \rho) \in \mathbb{R}^{m+2} \ : \ \zeta_1(b, \rho) \le c, \ \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \le \lambda^2, \boldsymbol{\alpha} \in \mathcal{S} \right\} \end{aligned} \tag{29}$$

holds. Note that any vector $\boldsymbol{\alpha}$ satisfying $\sum_{i,j=1}^m \alpha_i \alpha_j k(x_i, x_j) \le \lambda^2$ and $\boldsymbol{\alpha} \in \mathcal{S}$ is included in a compact subset of $\mathbb{R}^m$. We shall prove that the subset (29) is compact, if they are not empty. We see that the two sets above are closed subsets, since both $\zeta_0$ and $\zeta_1$ are continuous. By the variable change from $(b, \rho)$ to $(u_1, u_2) = (\rho - b, \rho + b)$, $\zeta_1(b, \rho)$ is transformed to a convex function $\zeta_2(u_1, u_2)$ defined by

$$\zeta_2(u_1, u_2) = -u_1 + \frac{m_p}{m}\ell(u_1 - K\lambda) - u_2 + \frac{m_n}{m}\ell(u_2 - K\lambda).$$

The subgradient of $\ell(z)$ diverges to infinity, when $z$ tends to infinity. In addition, $\ell(z)$ is a non-decreasing and non-negative function. Hence we have

$$\lim_{|u_1| \to \infty} -u_1 + \frac{m_p}{m}\ell(u_1 - K\lambda) = \infty.$$

The same limit holds for $-u_2 + \frac{m_n}{m}\ell(u_2 - K\lambda)$. Hence, the level set of $\zeta_2(u_1,u_2)$ is closed and bounded, that is, compact. As a result, the level set of $\zeta_1(b,\rho)$ is also compact. Therefore, the subset (29) is also compact in $\mathbb{R}^{m+2}$. This implies that (28) has an optimal solution.

Next, we prove the duality between (25) and (27). Since (28) has an optimal solution, the problem with the slack variables $\xi_i, i = 1,\ldots,m$,

$$\begin{aligned}
&\min_{\alpha,b,\rho,\xi} -2\rho + \frac{1}{m}\sum_{i=1}^{m}\ell(\xi_i)\\
&\text{subject to } \sum_{i,j=1}^{m}\alpha_i\alpha_j k(x_i,x_j) \le \lambda^2,\\
&\qquad\qquad \rho - y_i\Big(\sum_{j=1}^{m}\alpha_i k(x_i,x_j) + b\Big) \le \xi_i,\ i = 1,\ldots,m,
\end{aligned}$$

also has an optimal solution and the finite optimal value. In addition, the above problem clearly satisfies the Slater condition (Bertsekas et al., 2003, Assumption 6.4.2). Indeed, at a feasible solution, $\alpha = \mathbf{0}, b = 0, \rho = 0$ and $\xi_i = 1, i = 1,\ldots,m$, the constraint inequalities are all inactive for positive $\lambda$. Hence, Proposition 6.4.3 in Bertsekas et al. (2003) ensures that the min-max theorem holds, that is, there is no duality gap. Then, in the same way as (10), we obtain (25) with the uncertainty set (26) as the dual problem of (27). ■

## Appendix C. Proof of Consistency

We define some notations. For a measurable function $f : \mathcal{X} \to \mathbb{R}$ or $f \in \mathcal{H}$, and a real number $\rho \in \mathbb{R}$, we define the expected loss $\mathcal{R}(f,\rho)$ and the regularized expected loss $\mathcal{R}_\lambda(f,\rho)$ by

$$\begin{aligned}
\mathcal{R}(f,\rho) &= -2\rho + \mathbb{E}[\ell(\rho - yf(x))],\\
\mathcal{R}_\lambda(f,\rho) &= -2\rho + \mathbb{E}[\ell(\rho - yf(x))] + \theta(\|f\|_{\mathcal{H}}^2 \le \lambda^2),
\end{aligned}$$

where $\lambda$ is a positive number and $\theta(A)$ equals 0 when $A$ is true and $\infty$ otherwise. Let $\mathcal{R}^*$ be the infimum of $\mathcal{R}(f,\rho)$,

$$\mathcal{R}^* = \inf\{\mathcal{R}(f,\rho) : f \in L_0, \rho \in \mathbb{R}\}.$$

For the set of training samples, $T = \{(x_1,y_1),\ldots,(x_m,y_m)\}$, the empirical loss $\widehat{\mathcal{R}}_T(f,\rho)$ and the regularized empirical loss $\widehat{\mathcal{R}}_{T,\lambda}(f,\rho)$ are defined by

$$\begin{aligned}
\widehat{\mathcal{R}}_T(f,\rho) &= -2\rho + \frac{1}{m}\sum_{i=1}^{m}\ell(\rho - y_i f(x_i)),\\
\widehat{\mathcal{R}}_{T,\lambda}(f,\rho) &= -2\rho + \frac{1}{m}\sum_{i=1}^{m}\ell(\rho - y_i f(x_i)) + \theta(\|f\|_{\mathcal{H}}^2 \le \lambda^2).
\end{aligned}$$

The subscript $T$ is dropped if it is clear from the context. By a slight abuse of notation, for a function $f \in \mathcal{H}$ and a real number $b$, the regularized expected loss $\mathcal{R}_\lambda(f+b,\rho)$ denotes

$$\mathcal{R}_\lambda(f+b,\rho) = -2\rho + \mathbb{E}[\ell(\rho - y(f(x)+b))] + \theta(\|f\|_{\mathcal{H}}^2 \le \lambda^2).$$

The similar notation $\widehat{\mathcal{R}}_{T,\lambda}(f+b,\rho)$ is also used for the regularized empirical loss, that is,

$$\widehat{\mathcal{R}}_{T,\lambda}(f+b,\rho) = -2\rho + \frac{1}{m}\sum_{i=1}^{m}\ell(\rho - y_i(f(x_i)+b)) + \theta(\|f\|_{\mathcal{H}}^2 \le \lambda^2).$$

For the observed training samples $T$, clearly the problem (27) is identical to the minimization of $\widehat{\mathcal{R}}_{T,\lambda}(f+b,\rho)$. We define $\widehat{f},\widehat{b}$ and $\widehat{\rho}$ as an optimal solution of

$$\min_{f,b,\rho}\widehat{\mathcal{R}}_{T,\lambda_m}(f+b,\rho), \quad f\in\mathcal{H}, b\in\mathbb{R}, \rho\in\mathbb{R}, \tag{30}$$

where the regularization parameter $\lambda_m$ may depend on the sample size, $m$.

In this section, we prove that the error rate $\mathcal{E}(\widehat{f}+\widehat{b})$ converges to the Bayes risk $\mathcal{E}^*$. The proof consists of two parts. In Section C.1, we prove that the expected loss for the estimated decision function, $\mathcal{R}(\widehat{f}+\widehat{b},\widehat{\rho})$, converges to the infimum of the expected loss $\mathcal{R}^*$, where $\widehat{f},\widehat{b}$ and $\widehat{\rho}$ are optimal solutions of (30). Here, we apply the mathematical tools developed by Steinwart (2005). In Section C.2, we prove the convergence of the error rate $\mathcal{E}(\widehat{f}+\widehat{b})$ to the Bayes risk $\mathcal{E}^*$. In the proof, the concept of the classification-calibrated loss (Bartlett et al., 2006) plays an important role.

In the following, Assumptions 1–4 are presented in Section 6.

### C.1 Convergence to Optimal Expected Loss

In this section, we prove that $\mathcal{R}(\widehat{f}+\widehat{b},\widehat{\rho})$ converges to $\mathcal{R}^*$. Following lemmas show the relation between the expected loss and the regularized expected loss.

**Lemma 3** *Under Assumption 2 and Assumption 3, we have* $\mathcal{R}^* > -\infty$.

**Proof** Let $S\subset\mathcal{X}$ be the subset $S=\{x\in\mathcal{X} : \varepsilon\le P(+1|x)\le 1-\varepsilon\}$, then Assumption 2 assures $P(S)>0$. Due to the non-negativity of the loss function $\ell$, we have

$$\begin{aligned}\mathcal{R}(f,\rho) &\ge -2\rho + \int_S\Big\{P(+1|x)\ell(\rho-f(x)) + P(-1|x)\ell(\rho+f(x))\Big\}P(dx)\\ &= \int_S\Big\{-\frac{2}{P(S)}\rho + P(+1|x)\ell(\rho-f(x)) + P(-1|x)\ell(\rho+f(x))\Big\}P(dx).\end{aligned}$$

For given $\eta$ satisfying $\varepsilon\le\eta\le 1-\varepsilon$, we define the function $\xi(f,\rho)$ by

$$\xi(f,\rho) = -\frac{2}{P(S)}\rho + \eta\ell(\rho-f) + (1-\eta)\ell(\rho+f), \quad f,\rho\in\mathbb{R}.$$

We derive a lower bound $\inf\{\xi(f,\rho) : f,\rho\in\mathbb{R}\}$. Since $\ell(z)$ is a finite-valued convex function on $\mathbb{R}$, the subdifferential $\partial\xi(f,\rho)\subset\mathbb{R}^2$ is given as

$$\partial\xi(f,\rho) = \left\{\begin{pmatrix}0\\-2/P(S)\end{pmatrix} + \eta u\begin{pmatrix}-1\\1\end{pmatrix} + (1-\eta)v\begin{pmatrix}1\\1\end{pmatrix} : u\in\partial\ell(\rho-f), v\in\partial\ell(\rho+f)\right\}.$$

Formulas of the subdifferential are presented in Theorem 23.8 and Theorem 23.9 of Rockafellar (1970). We prove that there exist $f^*$ and $\rho^*$ such that $(0,0)^T\in\partial\xi(f^*,\rho^*)$ holds. Since the second condition in Assumption 3 holds for the convex function $\ell$, the union $\cup_{z\in\mathbb{R}}\partial\ell(z)$ includes all the

positive real numbers. Hence, there exist real numbers $z_1$ and $z_2$ satisfying $\frac{1}{\eta P(S)} \in \partial\ell(z_1)$ and $\frac{1}{(1-\eta)P(S)} \in \partial\ell(z_2)$. Then, for $f^* = (z_2 - z_1)/2$, $\rho^* = (z_1 + z_2)/2$, the null vector is an element of $\partial\xi(f^*, \rho^*)$. Since $\xi(f, \rho)$ is convex in $(f, \rho)$, the minimum value of $\xi(f, \rho)$ is attained at $(f^*, \rho^*)$. Define $z_{\mathrm{up}}$ as a real number satisfying

$$g > \frac{1}{\varepsilon P(S)}, \quad \forall g \in \partial\ell(z_{\mathrm{up}}).$$

Since $\varepsilon \leq \eta \leq 1 - \varepsilon$ is assumed, both $z_1$ and $z_2$ are less than $z_{\mathrm{up}}$ due to the monotonicity of the subdifferential. Then, the inequality

$$\xi(f, \rho) \geq \xi(f^*, \rho^*) = -\frac{z_1 + z_2}{P(S)} + \eta\ell(z_1) + (1-\eta)\ell(z_2) \geq -\frac{2z_{\mathrm{up}}}{P(S)}$$

holds for all $f, \rho \in \mathbb{R}$ and all $\eta$ such that $\varepsilon \leq \eta \leq 1 - \varepsilon$. The right-side of the expression above depends only on $P(S)$ and $\varepsilon$. Hence, for any measurable function $f \in L_0$ and $\rho \in \mathbb{R}$, we have

$$\mathcal{R}(f, \rho) \geq \int_S \frac{-2z_{\mathrm{up}}}{P(S)} P(dx) \geq -2z_{\mathrm{up}}.$$

As a result, we have $\mathcal{R}^* \geq -2z_{\mathrm{up}} > -\infty$. ■

**Lemma 4** *Under Assumption 1, 2 and 3, we have*

$$\lim_{\lambda \to \infty} \inf\{\mathcal{R}_\lambda(f + b, \rho) : f \in \mathcal{H}, b, \rho \in \mathbb{R}\} = \mathcal{R}^*. \tag{31}$$

**Proof** Corollary 5.29 of Steinwart and Christmann (2008) ensures that the equality

$$\inf\{\mathbb{E}[\ell(\rho - y(f(x) + b))] : f \in \mathcal{H}, b \in \mathbb{R}\} = \inf\{\mathbb{E}[\ell(\rho - yf(x))] : f \in L_0\}$$

holds for any $\rho \in \mathbb{R}$. Thus, we have

$$\inf\{\mathcal{R}(f + b, \rho) : f \in \mathcal{H}, b \in \mathbb{R}\} = \inf\{\mathcal{R}(f, \rho) : f \in L_0\}$$

for any $\rho \in \mathbb{R}$. Then, the equality

$$\inf\{\mathcal{R}(f + b, \rho) : f \in \mathcal{H}, b, \rho \in \mathbb{R}\} = \mathcal{R}^*$$

holds. Under Assumption 2 and Assumption 3, we have $\mathcal{R}^* > -\infty$ due to Lemma 3. Then, for any $\varepsilon > 0$, there exist $\lambda_\varepsilon > 0, f_\varepsilon \in \mathcal{H}$, $b_\varepsilon \in \mathbb{R}$ and $\rho_\varepsilon \in \mathbb{R}$ such that $\|f_\varepsilon\|_{\mathcal{H}} \leq \lambda_\varepsilon$ and $\mathcal{R}(f_\varepsilon + b_\varepsilon, \rho_\varepsilon) \leq \mathcal{R}^* + \varepsilon$ hold. For all $\lambda \geq \lambda_\varepsilon$ we have

$$\inf\{\mathcal{R}_\lambda(f + b, \rho) : f \in \mathcal{H}, b, \rho \in \mathbb{R}\} \leq \mathcal{R}_\lambda(f_\varepsilon + b_\varepsilon, \rho_\varepsilon) = \mathcal{R}(f_\varepsilon + b_\varepsilon, \rho_\varepsilon) \leq \mathcal{R}^* + \varepsilon.$$

On the other hand, it is clear that the inequality $\mathcal{R}^* \leq \inf\{\mathcal{R}_\lambda(f + b, \rho) : f \in \mathcal{H}, b, \rho \in \mathbb{R}\}$ holds. Thus we obtain Equation (31). ■

We derive an upper bound on the norm of the optimal solution in (30).

**Lemma 5** *Suppose* $\lim_{m\to\infty}\lambda_m = \infty$. *Under Assumption 1, 2 and 3, there are positive constants c and C and a natural number M such that the optimal solution of* (30) *satisfies*

$$\|\widehat{f}\|_{\mathcal{H}} \leq \lambda_m, \quad |\widehat{b}| \leq C\lambda_m, \quad |\widehat{\rho}| \leq C\lambda_m \tag{32}$$

*with the probability greater than* $1-e^{-cm}$ *for* $m \geq M$.

More precisely, $M$ depends on the label probability, the function value $\ell(0)$ and the divergence speed of the sequence $\{\lambda_m\}$, and $c$ depends only on the label probability. We can choose $C = K+1$, where $K$ is defined in Assumption 1.

**Proof** Under Assumption 2, the label probabilities, $P(y=+1)$ and $P(y=-1)$, are positive. We assume that the inequalities

$$\frac{1}{2}P(Y=+1) < \frac{m_p}{m}, \quad \frac{1}{2}P(Y=-1) < \frac{m_n}{m} \tag{33}$$

hold. Applying Chernoff bound, we see that there exists a positive constant $c>0$ depending only on the marginal probability of the label such that (33) holds with the probability higher than $1-e^{-cm}$.

Lemma 2 ensures that the problem (30) has optimal solution, $\widehat{f},\widehat{b},\widehat{\rho}$. The first inequality in (32), that is, $\|\widehat{f}\|_{\mathcal{H}} \leq \lambda_m$, is clearly satisfied. Then, we have $\|\widehat{f}\|_\infty \leq K\|\widehat{f}\|_{\mathcal{H}} \leq K\lambda_m$ from the reproducing property of the RKHSs. The definition of the estimator and the non-negativity of $\ell$ yield that

$$-2\widehat{\rho} \leq -2\widehat{\rho} + \frac{1}{m}\sum_{i=1}^{m}\ell(\widehat{\rho} - y_i(\widehat{f}(x_i)+\widehat{b})) \leq \widehat{\mathcal{R}}_{T,\lambda_m}(0,0) = \ell(0).$$

Then, we have

$$\widehat{\rho} \geq -\frac{\ell(0)}{2}. \tag{34}$$

Next, we consider the optimality condition of $\widehat{\mathcal{R}}_{T,\lambda_m}$. According to the calculus of subdifferential introduced in Section 23 of Rockafellar (1970), the derivative of the objective function with respect to $\rho$ leads to an optimality condition,

$$0 \in -2 + \frac{1}{m}\sum_{i=1}^{m}\partial\ell(\widehat{\rho} - y_i(\widehat{f}(x_i)+\widehat{b})).$$

The monotonicity and non-negativity of the subdifferential and the bound of $\|f\|_\infty$ lead to

$$\begin{aligned}
2 &\geq \frac{1}{m}\sum_{i=1}^{m}\partial\ell(\widehat{\rho} - y_i\widehat{b} - K\lambda_m) \\
&= \frac{1}{m}\sum_{i=1}^{m_p}\partial\ell(\widehat{\rho} - \widehat{b} - K\lambda_m) + \frac{1}{m}\sum_{j=1}^{m_n}\partial\ell(\widehat{\rho} + \widehat{b} - K\lambda_m) \\
&\geq \frac{1}{m}\sum_{i=1}^{m_p}\partial\ell(\widehat{\rho} - \widehat{b} - K\lambda_m).
\end{aligned} \tag{35}$$

The above expression means that there exists a number in the subdifferential such that the inequality holds, where $\sum_{i=1}^{m_p}\partial\ell$ denotes the $m_p$-fold sum of the set $\partial\ell$, that is, $\{a_1+\cdots+a_{m_p} : a_i \in \partial\ell, i=$

$1,\ldots,m_p\}$. Let $z_p$ be a real number satisfying $\frac{2m}{m_p} < \partial\ell(z_p)$, that is, all elements in $\partial\ell(z_p)$ are greater than $\frac{2m}{m_p}$. Then, the inequality $\widehat{\rho} - \widehat{b} - K\lambda_m < z_p$ should hold. Otherwise the inequality (35) does not hold. In the same way, for $z_n$ satisfying $\frac{2m}{m_n} < \partial\ell(z_n)$, we have $\widehat{\rho} + \widehat{b} - K\lambda_m < z_n$. The existence of $z_p$ and $z_n$ is guaranteed by Assumption 3. Hence, the inequalities

$$-\frac{\ell(0)}{2} \leq \widehat{\rho} \leq K\lambda_m + \max\{z_p, z_n\}, \qquad |\widehat{b}| \leq \frac{\ell(0)}{2} + K\lambda_m + \max\{z_p, z_n\} \tag{36}$$

hold, in which $\widehat{\rho} \geq -\ell(0)/2$ is used in the second inequality. Define $\bar{z}$ as a positive real number such that

$$\forall g \in \partial\ell(\bar{z}), \quad \max\left\{\frac{4}{P(Y=+1)}, \frac{4}{P(Y=-1)}\right\} < g.$$

Inequalities in (33) lead to

$$\max\left\{\frac{2m}{m_p}, \frac{2m}{m_n}\right\} < \max\left\{\frac{4}{P(Y=+1)}, \frac{4}{P(Y=-1)}\right\}.$$

Hence, we can choose $\bar{z} > 0$ satisfying $\max\{z_p, z_n\} < \bar{z}$. Note that $\bar{z}$ depends only on the label probability. Suppose that $\ell(0)/2 + \bar{z} \leq \lambda_m$ holds for $m \geq M$. Then from (36) we have

$$|\widehat{\rho}| \leq (K+1)\lambda_m, \quad |\widehat{b}| \leq (K+1)\lambda_m.$$

for $m \geq M$. Then we obtain (32) with $C = K+1$, when (33) holds. ■

Let us define the covering number for a metric space.

**Definition 6 (covering number)** *For a metric space $\mathcal{G}$, the covering number of $\mathcal{G}$ is defined as*

$$\mathcal{N}(\mathcal{G}, \varepsilon) = \min\left\{n \in \mathbb{N} : g_1, \ldots, g_n \in \mathcal{G} \text{ such that } \mathcal{G} \subset \bigcup_{i=1}^{n} B(g_i, \varepsilon)\right\},$$

*where $B(g, \varepsilon)$ denotes the closed ball with center $g$ and radius $\varepsilon$.*

According to Lemma 5, the optimal solution $(\widehat{f}, \widehat{b}, \widehat{\rho})$ is included in the set

$$\mathcal{G}_m = \{(f, b, \rho) \in \mathcal{H} \times \mathbb{R}^2 : \|f\|_{\mathcal{H}} \leq \lambda_m, |b| \leq C\lambda_m, |\rho| \leq C\lambda_m\}$$

with high probability. Suppose that the norm $\|f\|_\infty + |b| + |\rho|$ is introduced on $\mathcal{G}_m$. We define the function parametrized by $(f, b, \rho)$,

$$L(x, y; f, b, \rho) = -2\rho + \ell(\rho - y(f(x) + b)),$$

and the function set

$$\mathcal{L}_m = \{L(x, y; f, b, \rho) : (f, b, \rho) \in \mathcal{G}_m\}.$$

The supremum norm is defined on $\mathcal{L}_m$. The expected loss and the empirical loss, $\mathcal{R}(f + b, \rho)$ and $\widehat{\mathcal{R}}_T(f + b, \rho)$, are represented as the expectation of $L(x, y; f, b, \rho)$ with respect to the population

distribution and the empirical distribution, respectively. Since $\ell : \mathbb{R} \to \mathbb{R}$ is a finite-valued convex function, $\ell$ is locally Lipschitz continuous. Then, for any sample size $m$, there exists a constant $\kappa_m$ depending on $m$ such that

$$|\ell(z) - \ell(z')| \leq \kappa_m |z - z'| \tag{37}$$

holds for all $z$ and $z'$ satisfying $|z|, |z'| \leq (K + 2C)\lambda_m$. Then, for any $(f, b, \rho), (f', b', \rho') \in \mathcal{G}_m$, we have

$$\begin{aligned} |L(x, y; f, b, \rho) - L(x, y; f', b', \rho')| &\leq 2|\rho - \rho'| + \kappa_m(|\rho - \rho'| + |b - b'| + \|f - f'\|_\infty) \\ &\leq (2 + \kappa_m)(|\rho - \rho'| + |b - b'| + \|f - f'\|_\infty) \end{aligned}$$

The covering number of $\mathcal{L}_m$ is evaluated by using that of $\mathcal{G}_m$ as follows:

$$\mathcal{N}(\mathcal{L}_m, \varepsilon) \leq \mathcal{N}\big(\mathcal{G}_m, \frac{\varepsilon}{2 + \kappa_m}\big). \tag{38}$$

Let the metric space $\mathcal{F}_m$ be

$$\mathcal{F}_m = \{f \in \mathcal{H} : \|f\|_{\mathcal{H}} \leq \lambda_m\}$$

with the supremum norm, then we also have

$$\begin{aligned} \mathcal{N}\left(\mathcal{G}_m, \frac{\varepsilon}{2 + \kappa_m}\right) &\leq \mathcal{N}\left(\mathcal{F}_m, \frac{\varepsilon}{3(2 + \kappa_m)}\right) \left(\frac{2C\lambda_m}{\frac{\varepsilon}{3(2+\kappa_m)}}\right)^2 \\ &= \mathcal{N}\left(\mathcal{F}_m, \frac{\varepsilon}{3(2 + \kappa_m)}\right) \left(\frac{6C\lambda_m(2 + \kappa_m)}{\varepsilon}\right)^2. \end{aligned} \tag{39}$$

An upper bound of the covering number of $\mathcal{F}_m$ endowed with the supremum norm is given by Cucker and Smale (2002) and Zhou (2002).

We prove the uniform convergence of $\widehat{\mathcal{R}}(f + b, \rho)$.

**Lemma 7** *Let $b_m$ be $b_m = 4C\lambda_m + \ell((K + 2C)\lambda_m)$ in which $C$ is the positive constant defined in Lemma 5. Under Assumption 1 and 3, the inequality*

$$\begin{aligned} &P\left(\sup_{(f,b,\rho)\in\mathcal{G}_m} |\widehat{\mathcal{R}}(f + b, \rho) - \mathcal{R}(f + b, \rho)| \geq \varepsilon\right) \\ &\leq 2\mathcal{N}(\mathcal{L}_m, \varepsilon/3) \exp\left\{-\frac{2m\varepsilon^2}{9b_m^2}\right\} \qquad (40) \\ &\leq 2\mathcal{N}\left(\mathcal{F}_m, \frac{\varepsilon}{9(2 + \kappa_m)}\right) \left(\frac{18C\lambda_m(2 + \kappa_m)}{\varepsilon}\right)^2 \exp\left\{-\frac{2m\varepsilon^2}{9b_m^2}\right\} \qquad (41) \end{aligned}$$

*holds, where $\kappa_m$ is the Lipschitz constant defined by* (37).

**Proof** Since $\|f\|_\infty \leq K\lambda_m$ holds for $f \in \mathcal{H}$ such that $\|f\|_{\mathcal{H}} \leq \lambda_m$, we have the following inequality

$$\begin{aligned}
&\sup_{\substack{(x,y)\in\mathcal{X}\times\{+1,-1\}\\(f,b,\rho)\in\mathcal{G}_m}} L(x,y;f,b,\rho) - \inf_{\substack{(x,y)\in\mathcal{X}\times\{+1,-1\}\\(f,b,\rho)\in\mathcal{G}_m}} L(x,y;f,b,\rho) \\
&\leq 2C\lambda_m + \sup_{\substack{(x,y)\in\mathcal{X}\times\{+1,-1\}\\(f,b,\rho)\in\mathcal{G}_m}} \ell(\rho - y(f(x)+b)) - (-2C\lambda_m) \\
&\leq 4C\lambda_m + \ell(C\lambda_m + K\lambda_m + C\lambda_m) \\
&= b_m.
\end{aligned}$$

In the same way as the proof of Lemma 3.4 in Steinwart (2005), Hoeffding's inequality leads to the upper bound (40). Equation (41) is the direct conclusion of (38) and (39). ■

We present the main theorem of this section.

**Theorem 8** *Suppose that* $\lim_{m\to\infty}\lambda_m = \infty$ *holds. Suppose that Assumption 1, 2 and 3 hold. Moreover we assume that* (41) *converges to zero for any* $\varepsilon > 0$*, when the sample size* $m$ *tends to infinity. Then,* $\mathcal{R}(\widehat{f}+\widehat{b},\widehat{\rho})$ *converges to* $\mathcal{R}^*$ *in probability in the large sample limit of the data set* $T = \{(x_i,y_i) : i = 1,\ldots,m\}$.

Later on we show an example in which 7 converges to zero.

**Proof** Lemma 4 assures that, for any $\gamma > 0$, there exists sufficiently large $M_1$ such that

$$|\inf\{\mathcal{R}_{\lambda_m}(f+b,\rho) : f \in \mathcal{H}, b,\rho \in \mathbb{R}\} - \mathcal{R}^*| \leq \gamma$$

holds for all $m \geq M_1$. Thus, there exist $f_\gamma, b_\gamma$ and $\rho_\gamma$ such that

$$|\mathcal{R}_{\lambda_m}(f_\gamma + b_\gamma, \rho_\gamma) - \mathcal{R}^*| \leq 2\gamma$$

and $\|f_\gamma\|_{\mathcal{H}} \leq \lambda_m$ hold for $m \geq M_1$. Due to the law of large numbers, the inequality

$$|\widehat{\mathcal{R}}_T(f_\gamma + b_\gamma, \rho_\gamma) - \mathcal{R}(f_\gamma + b_\gamma, \rho_\gamma)| \leq \gamma$$

holds with high probability, say $1-\delta_m$, for $m \geq M_2$. The boundedness property in Lemma 5 leads to

$$P((\widehat{f},\widehat{b},\widehat{\rho}) \in \mathcal{G}_m) \geq 1 - e^{-cm}$$

for $m \geq M_3$. In addition, by the uniform bound shown in Lemma 7, the inequality

$$\sup_{(f,b,\rho)\in\mathcal{G}_m} |\widehat{\mathcal{R}}_T(f+b,\rho) - \mathcal{R}(f+b,\rho)| \leq \gamma$$

holds with probability $1-\delta'_m$. Hence, the probability such that the inequality

$$|\widehat{\mathcal{R}}_T(\widehat{f}+\widehat{b},\widehat{\rho}) - \mathcal{R}(\widehat{f}+\widehat{b},\widehat{\rho})| \leq \gamma$$

holds is greater than $1-e^{-cm}-\delta'_m$ for $m \geq M_3$. Let $M_0$ be $M_0 = \max\{M_1, M_2, M_3\}$. Then, for any $\gamma > 0$, the following inequalities hold with probability higher than $1-e^{-cm}-\delta'_m-\delta_m$ for $m \geq M_0$,

$$\begin{aligned}
\mathcal{R}(\widehat{f}+\widehat{b},\widehat{\rho}) &\leq \widehat{\mathcal{R}}_T(\widehat{f}+\widehat{b},\widehat{\rho})+\gamma \\
&\leq \widehat{\mathcal{R}}_T(f_\gamma+b_\gamma,\rho_\gamma)+\gamma \\
&\leq \mathcal{R}(f_\gamma+b_\gamma,\rho_\gamma)+2\gamma \\
&= \mathcal{R}_{\lambda_m}(f_\gamma+b_\gamma,\rho_\gamma)+2\gamma \\
&\leq \mathcal{R}^*+4\gamma.
\end{aligned} \tag{42}$$

The second inequality (42) above is given as

$$\widehat{\mathcal{R}}_T(\widehat{f}+\widehat{b},\widehat{\rho}) = \widehat{\mathcal{R}}_{T,\lambda_m}(\widehat{f}+\widehat{b},\widehat{\rho}) \leq \widehat{\mathcal{R}}_{T,\lambda_m}(f_\gamma+b_\gamma,\rho_\gamma) = \widehat{\mathcal{R}}_T(f_\gamma+b_\gamma,\rho_\gamma).$$

■

We show the order of $\lambda_m$ admitting the assumption in Theorem 8.

**Example 8** *Suppose that $\mathcal{X} = [0,1]^n \subset \mathbb{R}^n$ and the Gaussian kernel is used. According to Zhou (2002), we have*

$$\log \mathcal{N}\left(\mathcal{F}_m, \frac{\varepsilon}{9(2+\kappa_m)}\right) = O\left(\left(\log \frac{\lambda_m}{\frac{\varepsilon}{9(2+\kappa_m)}}\right)^{n+1}\right) = O\big(\big(\log(\lambda_m\kappa_m)\big)^{n+1}\big).$$

*For any $\varepsilon > 0$,* (41) *is bounded above by*

$$\exp\left\{O\left(-\frac{m}{b_m^2}+(\log(\lambda_m\kappa_m))^{n+1}\right)\right\}.$$

*For the truncated quadratic loss, we have*

$$\begin{aligned}
&\kappa_m \leq 2((K+2C)\lambda_m+1) = O(\lambda_m), \\
&b_m \leq 4C\lambda_m+((K+2C)\lambda_m+1)^2 = O(\lambda_m^2).
\end{aligned}$$

*Let us define $\lambda_m = m^\alpha$ with $0 < \alpha < 1/4$. Then, for any $\varepsilon > 0$,* (41) *converges to zero when m tends to infinity. In the same way, for the exponential loss we obtain*

$$\kappa_m = O(e^{(K+2C)\lambda_m}), \quad b_m = O(e^{(K+2C)\lambda_m}).$$

*Hence, $\lambda_m = (\log m)^\alpha$ with $0 < \alpha < 1$ assures the convergence of* (41).

### C.2 Convergence to Bayes Risk

We prove that the expected 0-1 loss $\mathcal{E}(\widehat{f}+\widehat{b})$ converges to the Bayes risk $\mathcal{E}^*$, when the sample size $m$ tends to infinity.

**Theorem 9** *Suppose that $\mathcal{R}(\widehat{f}+\widehat{b},\widehat{\rho})$ converges to $\mathcal{R}^*$ in probability, when the sample size m tends to infinity. For the RKHS $\mathcal{H}$ and the loss function $\ell$, we assume Assumption 1, 3 and 4. Then, $\mathcal{E}(\widehat{f}+\widehat{b})$ converges to $\mathcal{E}^*$ in probability.*

As a result, we find that the prediction error rate of $\widehat{f}+\widehat{b}$ converges to the Bayes risk under Assumption 1, 2, 3, 4, and the assumption on the covering number in Theorem 8.

**Proof** Suppose that $\rho$ satisfies $\rho \geq -\ell(0)/2$. Since $\ell'(\rho) > 0$ holds, the loss function $\ell(\rho - z)$ is classification-calibrated (Bartlett et al., 2006). Hence, for $\rho \geq -\ell(0)/2$ Theorem 1 and Theorem 2 of Bartlett et al. (2006) guarantee that $\psi(\theta,\rho)$ in Assumption 4 satisfies $\psi(0,\rho) = 0$, $\psi(\theta,\rho) > 0$ for $0 < \theta \leq 1$ and that $\psi(\theta,\rho)$ is continuous and strictly increasing in $\theta \in [0,1]$. In addition, for all $f \in \mathcal{H}$ and $b \in \mathbb{R}$ the inequality

$$\psi(\mathcal{E}(f+b) - \mathcal{E}^*, \rho) \leq \mathbb{E}[\ell(\rho - y(f(x)+b))] - \inf_{f\in\mathcal{H}, b\in\mathbb{R}} \mathbb{E}[\ell(\rho - y(f(x)+b))]$$

holds. Here we used the equality

$$\inf\{\mathbb{E}[\ell(\rho - y(f(x)+b))] : f \in \mathcal{H}, b \in \mathbb{R}\} = \inf\{\mathbb{E}[\ell(\rho - y(f(x)+b))] : f \in L_0, b \in \mathbb{R}\},$$

which is shown in Corollary 5.29 of Steinwart and Christmann (2008). Hence, we have

$$\begin{aligned}\psi(\mathcal{E}(\widehat{f}+\widehat{b}) - \mathcal{E}^*, \widehat{\rho}) &\leq \mathbb{E}[\ell(\widehat{\rho} - y(\widehat{f}(x)+\widehat{b}))] - \inf_{f\in\mathcal{H}, b\in\mathbb{R}} \mathbb{E}[\ell(\widehat{\rho} - y(f(x)+b))] \\ &= \mathcal{R}(\widehat{f}+\widehat{b}, \widehat{\rho}) - \inf_{f\in\mathcal{H}, b\in\mathbb{R}} \mathcal{R}(f+b, \widehat{\rho}),\end{aligned}$$

since $\widehat{\rho} \geq -\ell(0)/2$ holds due to (34). Since $\mathcal{R}(\widehat{f}+\widehat{b}, \widehat{\rho})$ is assumed to converge to $\mathcal{R}^*$ in probability, for any $\varepsilon > 0$ the inequality

$$\mathcal{R}^* \leq \inf_{f\in\mathcal{H}, b\in\mathbb{R}} \mathcal{R}(f+b, \widehat{\rho}) \leq \mathcal{R}(\widehat{f}+\widehat{b}, \widehat{\rho}) \leq \mathcal{R}^* + \varepsilon$$

holds with high probability for sufficiently large $m$. Thus, $\psi(\mathcal{E}(\widehat{f}+\widehat{b}) - \mathcal{E}^*, \widehat{\rho})$ converges to zero in probability. The inequality

$$0 \leq \widetilde{\psi}(\mathcal{E}(\widehat{f}+\widehat{b}) - \mathcal{E}^*) \leq \psi(\mathcal{E}(\widehat{f}+\widehat{b}) - \mathcal{E}^*, \widehat{\rho})$$

and the assumption on the function $\widetilde{\psi}$ ensure that $\mathcal{E}(\widehat{f}+\widehat{b})$ converges to $\mathcal{E}^*$ in probability, when $m$ tends to infinity. ■

### C.3 Sufficient Conditions for Existence of the Function $\widetilde{\psi}$ in Assumption 4

We present some sufficient conditions for existence of the function $\widetilde{\psi}$ in Assumption 4.

**Lemma 10** *Suppose that the first condition in Assumption 3 and the first condition in Assumption 4 hold. In addition, suppose that $\ell$ is first-order continuously differentiable on $\mathbb{R}$. Let $d$ be $d = \sup\{z \in \mathbb{R} : \ell'(z) = 0\}$, where $\ell'$ is the derivative of $\ell$. When $\ell'(z) > 0$ holds for all $z \in \mathbb{R}$, we define $d = -\infty$. We assume the following conditions:*

1. *$d < -\ell(0)/2$.*
2. *$\ell(z)$ is second-order continuously differentiable on the open interval $(d, \infty)$.*

3. $\ell''(z) > 0$ *holds on* $(d, \infty)$.

4. $1/\ell'(z)$ *is convex on* $(d, \infty)$.

*Then, for any* $\theta \in [0,1]$*, the function* $\psi(\theta, \rho)$ *is non-decreasing as the function of* $\rho$ *for* $\rho \geq -\ell(0)/2$.

When the condition in Lemma 10 is satisfied, we can choose $\psi(\theta, -\ell(0)/2)$ as $\widetilde{\psi}(\theta)$ for $0 \leq \theta \leq 1$, since $\psi(\theta, -\ell(0)/2)$ is classification-calibrated under the first condition in Assumption 4.

**Proof** For $\theta = 0$ and $\theta = 1$, we can directly confirm that the lemma holds. In the following, we assume $0 < \theta < 1$ and $\rho \geq -\ell(0)/2$. We consider the following optimization problem involved in $\psi(\theta, \rho)$,

$$\inf_{z \in \mathbb{R}} \frac{1+\theta}{2}\ell(\rho - z) + \frac{1-\theta}{2}\ell(\rho + z). \tag{43}$$

The function in the infimum is a finite-valued convex function for $z \in \mathbb{R}$, and diverges to infinity when $z$ tends to $\pm\infty$. Thus the problem (43) has an optimal solution $z^* \in \mathbb{R}$. The optimality condition leads to the equality

$$(1+\theta)\ell'(\rho - z^*) - (1-\theta)\ell'(\rho + z^*) = 0.$$

We assumed that both $1+\theta$ and $1-\theta$ are positive and that $\rho \geq -\ell(0)/2 > d$ holds. Hence, both $\ell'(\rho - z^*)$ and $\ell'(\rho + z^*)$ should not be zero. Indeed, if one of them is equal to zero, the other is also zero, and we have $\rho - z^* \leq d$ and $\rho + z^* \leq d$. We find that these inequalities contradict $\rho > d$. As a result, we have $\rho - z^* > d$ and $\rho + z^* > d$, that is, $|z^*| < \rho - d$. In addition, we have

$$\frac{1+\theta}{2} = \frac{\ell'(\rho + z^*)}{\ell'(\rho + z^*) + \ell'(\rho - z^*)}.$$

Since $\ell''(z) > 0$ holds on $(d, \infty)$, the second derivative of the objective in (43) satisfies the positivity condition,

$$(1+\theta)\ell''(\rho - z) + (1-\theta)\ell''(\rho + z) > 0$$

for all $z$ such that $\rho - z > d$ and $\rho + z > d$. Therefore, $z^*$ is uniquely determined. For a fixed $\theta \in (0,1)$, the optimal solution can be described as the function of $\rho$, that is, $z^* = z(\rho)$. By the implicit function theorem, $z(\rho)$ is continuously differentiable with respect to $\rho$. Then, the derivative of $\psi(\theta, \rho)$ is given as

$$\begin{aligned}
\frac{\partial}{\partial \rho}\psi(\theta, \rho) &= \frac{\partial}{\partial \rho}\left\{\ell(\rho) - \frac{1+\theta}{2}\ell(\rho - z(\rho)) - \frac{1-\theta}{2}\ell(\rho + z(\rho))\right\} \\
&= \ell'(\rho) - \frac{1+\theta}{2}\ell'(\rho - z(\rho))\left(1 - \frac{\partial z}{\partial \rho}\right) - \frac{1-\theta}{2}\ell'(\rho + z(\rho))\left(1 + \frac{\partial z}{\partial \rho}\right) \\
&= \ell'(\rho) - \frac{\ell'(\rho + z(\rho))}{\ell'(\rho + z(\rho)) + \ell'(\rho - z(\rho))}\ell'(\rho - z(\rho))\left(1 - \frac{\partial z}{\partial \rho}\right) \\
&\quad - \frac{\ell'(\rho - z(\rho))}{\ell'(\rho + z(\rho)) + \ell'(\rho - z(\rho))}\ell'(\rho + z(\rho))\left(1 + \frac{\partial z}{\partial \rho}\right) \\
&= \ell'(\rho) - \frac{2\ell'(\rho - z(\rho))\ell'(\rho + z(\rho))}{\ell'(\rho + z(\rho)) + \ell'(\rho - z(\rho))}.
\end{aligned}$$

The convexity of $1/\ell'(z)$ for $z > d$ leads to

$$0 < \frac{1}{\ell'(\rho)} \leq \frac{1}{2\ell'(\rho + z(\rho))} + \frac{1}{2\ell'(\rho - z(\rho))} = \frac{\ell'(\rho + z(\rho)) + \ell'(\rho - z(\rho))}{2\ell'(\rho - z(\rho))\ell'(\rho + z(\rho))}.$$

Hence, we have

$$\frac{\partial}{\partial \rho}\psi(\theta, \rho) \geq 0$$

for $\rho \geq -\ell(0)/2 > d$ and $0 < \theta < 1$. As a result, we see that $\psi(\theta, \rho)$ is non-decreasing as the function of $\rho$. ■

We give another sufficient condition for existence of the function $\widetilde{\psi}$ in Assumption 4.

**Lemma 11** *Suppose that the first condition in Assumption 3 and the first condition in Assumption 4 hold. Let d be* $d = \sup\{z \in \mathbb{R} : \partial\ell(z) = \{0\}\}$. *When* $0 \notin \partial\ell(z)$ *holds for all* $z \in \mathbb{R}$, *we define* $d = -\infty$. *Suppose that the inequality* $-\ell(0)/2 > d$ *holds. For* $\rho \geq -\ell(0)/2$ *and* $z \geq 0$, *we define* $\xi(z, \rho)$ *by*

$$\xi(z, \rho) = \begin{cases} \dfrac{\ell(\rho + z) + \ell(\rho - z) - 2\ell(\rho)}{z\ell'(\rho)}, & z > 0, \\ 0, & z = 0. \end{cases}$$

*Suppose that there exists a function* $\bar{\xi}(z)$ *for* $z \geq 0$ *such that the following conditions hold:*

1. $\bar{\xi}(z)$ *is continuous and strictly increasing on* $z \geq 0$, *and satisfies* $\bar{\xi}(0) = 0$ *and* $\lim_{z\to\infty} \bar{\xi}(z) > 1$.
2. $\sup_{\rho \geq -\ell(0)/2} \xi(z, \rho) \leq \bar{\xi}(z)$ *holds.*

*Then, there exists a function* $\widetilde{\psi}$ *defined in the second condition of Assumption 4.*

Note that Lemma 11 does not require the second order differentiability of the loss function.

**Proof** We use the result of Bartlett et al. (2006). For a fixed $\rho$, the function $\xi(z, \rho)$ is continuous for $z \geq 0$, and the convexity of $\ell$ leads to the non-negativity of $\xi(z, \rho)$. Moreover, the convexity and the non-negativity of $\ell(z)$ lead to

$$\xi(z, \rho) \geq \frac{\ell(\rho + z) - \ell(\rho)}{z\ell'(\rho)} - \frac{\ell(\rho)}{z\ell'(\rho)} \geq 1 - \frac{\ell(\rho)}{z\ell'(\rho)}$$

for $z > 0$ and $\rho \geq -\ell(0)/2$, where $\ell(\rho)$ and $\ell'(\rho)$ are positive for $\rho > -\ell(0)/2$. The above inequality and the continuity of $\xi(\cdot, \rho)$ ensure that there exists $z$ satisfying $\xi(z, \rho) = \theta$ for all $\theta$ such that $0 \leq \theta < 1$. We define the inverse function $\xi_\rho^{-1}$ by

$$\xi_\rho^{-1}(\theta) = \inf\{z \geq 0 : \xi(z, \rho) = \theta\}$$

for $0 \leq \theta < 1$. For a fixed $\rho \geq -\ell(0)/2$, the loss function $\ell(\rho - z)$ is classification-calibrated (Bartlett et al., 2006). Hence, Lemma 3 in Bartlett et al. (2006) leads to the inequality

$$\psi(\theta, \rho) \geq \ell'(\rho)\frac{\theta}{2}\xi_\rho^{-1}\left(\frac{\theta}{2}\right),$$

for $0 \leq \theta < 1$. Define $\bar{\xi}^{-1}$ by

$$\bar{\xi}^{-1}(\theta) = \inf\{z \geq 0 : \bar{\xi}(z) = \theta\}.$$

From the definition of $\bar{\xi}(z)$, $\bar{\xi}^{-1}(\theta)$ is well-defined for all $\theta \in [0,1)$. Since $\xi(z,\rho) \leq \bar{\xi}(z)$ holds, we have $\xi_\rho^{-1}(\theta/2) \geq \bar{\xi}^{-1}(\theta/2)$. In addition, $\ell'(\rho)$ is non-decreasing as the function of $\rho$. Thus, we have

$$\psi(\theta,\rho) \geq \ell'(-\ell(0)/2)\frac{\theta}{2}\bar{\xi}^{-1}(\frac{\theta}{2})$$

for all $\rho \geq -\ell(0)/2$ and $0 \leq \theta < 1$. Then, we can choose

$$\widetilde{\psi}(\theta) = \ell'(-\ell(0)/2)\frac{\theta}{2}\bar{\xi}^{-1}(\frac{\theta}{2}).$$

It is straightforward to confirm that the conditions of Assumption 4 are satisfied. ■

We show some examples in which the existence of $\widetilde{\psi}$ is confirmed from above lemmas.

**Example 9** *For the truncated quadratic loss $\ell(z) = (\max\{z+1,0\})^2$, the first condition in Assumption 3 and the first condition in Assumption 4 hold. The inequality $-\ell(0)/2 = -1/2 > \sup\{z : \ell'(z) = 0\} = -1$ in the sufficient condition of Lemma 10 holds. For $z > -1$, it is easy to see that $\ell(z)$ is second-order differentiable and that $\ell''(z) > 0$ holds. In addition, for $z > -1$, $1/\ell'(z)$ is equal to $1/(2z+2)$ which is convex on $(-1,\infty)$. Therefore, the function $\widetilde{\psi}(\theta) = \psi(\theta,-1/2)$ satisfies the second condition in Assumption 4.*

**Example 10** *For the exponential loss $\ell(z) = e^z$, we have $1/\ell'(z) = e^{-z}$. Hence, due to Lemma 10, $\psi(\theta,\rho)$ is non-decreasing in $\rho$. Indeed, we have $\psi(\theta,\rho) = (1-\sqrt{1-\theta^2})e^\rho$.*

**Example 11** *In Example 6, we presented the uncertainty set with estimation errors. The uncertainty sets are defined based on the revised function $\bar{\ell}(z)$ in* (23)*. Here, we use a similar function defined by*

$$\bar{\ell}^*(\alpha) = \begin{cases} (|\alpha w - 1| + h)^2 - (1+h)^2, & \alpha \geq 0, \\ \infty, & \alpha < 0, \end{cases} \tag{44}$$

*for the construction of uncertainty sets. The function of the form* (44) *is derived by setting $\boldsymbol{\mu}_p^T \Sigma_p^{-1} \boldsymbol{\mu}_p = 1$ and $\boldsymbol{\mu}_n = \mathbf{0}$ in* (23)*. Here, $w$ and $h$ are positive constants, and we suppose $w > 1/2$. The corresponding loss function is given as $\bar{\ell}(z)$. Then we have $\bar{\ell}(z) = u(z/w)$ defined in* (24)*. For $w > 1/2$, we can confirm that $\sup\{z : \bar{\ell}'(z) = 0\} < -\bar{\ell}(0)/2$ holds. Since $u(z)$ is not strictly convex, Lemma 10 does not work. Hence, we apply Lemma 11. A simple calculation yields that $\bar{\ell}'(-\bar{\ell}(0)/2) \geq (4w-1)/(4w^2) > 0$ for any $h \geq 0$. Note that $\bar{\ell}(z)$ is differentiable on $\mathbb{R}$. Thus, the monotonicity of $\bar{\ell}'$ for the convex function leads to*

$$\xi(z,\rho) = \frac{1}{\bar{\ell}'(\rho)}\left(\frac{\bar{\ell}(\rho+z)-\bar{\ell}(\rho)}{z} - \frac{\bar{\ell}(\rho)-\bar{\ell}(\rho-z)}{z}\right) \leq \frac{\bar{\ell}'(\rho+z)-\bar{\ell}'(\rho-z)}{\bar{\ell}'(\rho)}.$$

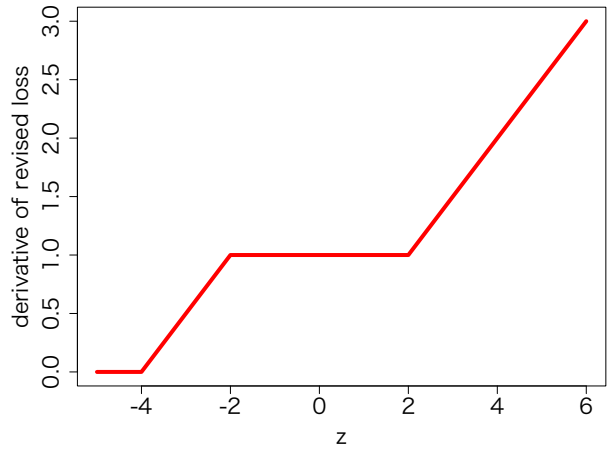



Figure 6: The derivative of the loss function corresponding to the revised uncertainty set with the estimation error.

*Figure 6 depicts the derivative of* $\bar{\ell}$ *with* $h = 1$ *and* $w = 1$. *Since the derivative* $\bar{\ell}'(z)$ *is Lipschitz continuous and the Lipschitz constant is equal to* $1/(2w)$, *we have* $\bar{\ell}'(\rho + z) - \bar{\ell}'(\rho - z) \leq z/w$. *Therefore, the inequality*

$$\sup_{\rho \geq -\bar{\ell}(0)/2} \xi(z, \rho) \leq \sup_{\rho \geq -\bar{\ell}(0)/2} \frac{z/w}{\bar{\ell}'(\rho)} = \frac{z/w}{\bar{\ell}'(-\bar{\ell}(0)/2)} \leq \frac{4w}{4w-1} z \leq 2z$$

*holds. We see that* $\bar{\xi}(z) = 2z$ *satisfies the sufficient condition of Lemma 11. The inequality*

$$\bar{\ell}'(-\bar{\ell}(0)/2) \frac{\theta}{2} \bar{\xi}^{-1}(\frac{\theta}{2}) \geq \frac{4w-1}{32w^2} \theta^2$$

*ensures that* $\widetilde{\psi}(\theta) = \frac{4w-1}{32w^2}\theta^2$ *is a valid choice. Therefore, the loss function corresponding to the revised uncertainty set in Example 6 satisfies the sufficient conditions for the statistical consistency.*

## References

S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.*, 54(2):317–331, 1997.

P. L. Bartlett and A. Tewari. Sparseness vs estimating conditional probabilities: Some asymptotic results. *Journal of Machine Learning Research*, 8:775–790, April 2007.

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.

A. Ben-Tal and A. Nemirovski. Robust optimization - methodology and applications. *Math. Program.*, 92(3):453–480, 2002.

A. Ben-Tal, L. El-Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, Princeton, 2009.

K. P. Bennett and E. J. Bredensteiner. Duality and geometry in SVM classifiers. In *Proceedings of International Conference on Machine Learning*, pages 57–64, 2000.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic, 2004.

D. Bertsekas, A. Nedic, and A. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, Belmont, MA, 2003.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

D. J. Crisp and C. J. C. Burges. A geometric interpretation of ν-SVM classifiers. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 244–250. MIT Press, 2000.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2002.

T. Evgeniou, M. Pontil, and T. Poggio. A unified framework for regularization networks and support vector machines. *Laboratory, Massachusetts Institute of Technology*, 1999.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, aug 1997.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

T. Hastie, R. Tibishirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

K. Huang, H. Yang, I. King, M. R. Lyu, and Laiwan Chan. The minimum error minimax probability machine. *Journal of Machine Learning Research*, 5:1253–1286, 2004.

A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004.

G. R. G. Lanckriet, L. El Ghaoui, C. Bhattacharyya, and M. I. Jordan. A robust minimax approach to classification. *Journal of Machine Learning Research*, 3:555–582, 2003.

M. E. Mavroforakis and S. Theodoridis. A geometric approach to support vector machine (svm) classification. *IEEE Transactions on Neural Networks*, 17(3):671–682, 2006.

J. S. Nath and C. Bhattacharyya. Maximum margin classifiers with specified false positive and false negative error rates. In C. Apte, B. Liu, S. Parthasarathy, and D. Skillicorn, editors, *Proceedings of the seventh SIAM International Conference on Data mining*, pages 35–46. SIAM, 2007.

G. Rätsch, B. Schölkopf, A.J. Smola, S. Mika, T. Onoda, and K.-R. Müller. *Robust Ensemble Learning.*, pages 207–220. MIT Press, Cambridge, MA, 2000.

G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for adaboost. *Machine Learning*, 42(3): 287–320, 2001.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, USA, 1970.

R. T. Rockafellar and S. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking & Finance*, 26(7):1443–1472, 2002.

R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

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

B. Schölkopf, A. Smola, R. Williamson, and P. Bartlett. New support vector algorithms. *Neural Computation*, 12(5):1207–1245, 2000.

I. Steinwart. On the optimal parameter choice for ν-support vector machines. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(10):1274–1284, 2003.

I. Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer Publishing Company, Incorporated, 1st edition, 2008.

S. Sun and J. Shawe-Taylor. Sparse semi-supervised learning using conjugate functions. *Journal of Machine Learning Research*, 11:2423–2455, 2010.

A. Takeda and M. Sugiyama. ν-support vector machine as conditional value-at-risk minimization. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1056–1063, 2008.

A. Takeda, H. Mitsugi, and T. Kanamori. A unified classification model based on robust optimization. *Neural Computation*, 25(3):759–804, 2013.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

T. Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statitics*, 32(1):56–85, 2004.

D.-X. Zhou. The covering number in learning theory. *Journal of Complexity*, 18(3):739–767, 2002.

Journal of Machine Learning Research 14 (2013) 1505-1511 Submitted 5/12; Revised 3/13; Published 6/13

# A Risk Comparison of Ordinary Least Squares vs Ridge Regression

**Paramveer S. Dhillon** DHILLON@CIS.UPENN.EDU
*Department of Computer and Information Science*
*University of Pennsylvania*
*Philadelphia, PA 19104, USA*

**Dean P. Foster** FOSTER@WHARTON.UPENN.EDU
*Department of Statistics*
*Wharton School, University of Pennsylvania*
*Philadelphia, PA 19104, USA*

**Sham M. Kakade** SKAKADE@MICROSOFT.COM
*Microsoft Research*
*One Memorial Drive*
*Cambridge, MA 02142, USA*

**Lyle H. Ungar** UNGAR@CIS.UPENN.EDU
*Department of Computer and Information Science*
*University of Pennsylvania*
*Philadelphia, PA 19104, USA*

**Editor:** Gabor Lugosi

## Abstract

We compare the risk of ridge regression to a simple variant of ordinary least squares, in which one simply projects the data onto a finite dimensional subspace (as specified by a principal component analysis) and then performs an ordinary (un-regularized) least squares regression in this subspace. This note shows that the risk of this ordinary least squares method (PCA-OLS) is within a constant factor (namely 4) of the risk of ridge regression (RR).

**Keywords:** risk inflation, ridge regression, pca

## 1. Introduction

Consider the fixed design setting where we have a set of $n$ vectors $\mathcal{X} = \{X_i\}$, and let $\mathbf{X}$ denote the matrix where the $i^{th}$ row of $\mathbf{X}$ is $X_i$. The observed label vector is $Y \in \mathbb{R}^n$. Suppose that:

$$Y = \mathbf{X}\beta + \varepsilon,$$

where $\varepsilon$ is independent noise in each coordinate, with the variance of $\varepsilon_i$ being $\sigma^2$.

The objective is to learn $\mathbb{E}[Y] = \mathbf{X}\beta$. The expected loss of a vector $\beta$ estimator is:

$$L(\beta) = \frac{1}{n}\mathbb{E}_{\mathbb{Y}}[\|Y - \mathbf{X}\beta\|^2],$$

Let $\hat{\beta}$ be an estimator of $\beta$ (constructed with a sample $Y$). Denoting

$$\mathbf{\Sigma} := \frac{1}{n}\mathbf{X}^T\mathbf{X},$$

©2013 Paramveer S. Dhillon, Dean P. Foster, Sham M. Kakade and Lyle H. Ungar.

we have that the risk (i.e., expected excess loss) is:

$$\text{Risk}(\hat{\beta}) := \mathbb{E}_{\hat{\beta}}[L(\hat{\beta}) - L(\beta)] = \mathbb{E}_{\hat{\beta}}\|\hat{\beta} - \beta\|_{\Sigma}^2,$$

where $\|x\|_{\Sigma} = x^{\top}\Sigma x$ and where the expectation is with respect to the randomness in $Y$.

We show that a simple variant of ordinary (un-regularized) least squares always compares favorably to ridge regression (as measured by the risk). This observation is based on the following bias variance decomposition:

$$\text{Risk}(\hat{\beta}) = \underbrace{\mathbb{E}\|\hat{\beta} - \bar{\beta}\|_{\Sigma}^2}_{\text{Variance}} + \underbrace{\|\bar{\beta} - \beta\|_{\Sigma}^2}_{\text{Prediction Bias}}, \tag{1}$$

where $\bar{\beta} = \mathbb{E}[\hat{\beta}]$.

### 1.1 The Risk of Ridge Regression (RR)

Ridge regression or Tikhonov Regularization (Tikhonov, 1963) penalizes the $\ell_2$ norm of a parameter vector $\beta$ and "shrinks" it towards zero, penalizing large values more. The estimator is:

$$\hat{\beta}_{\lambda} = \underset{\beta}{\text{argmin}}\{\|Y - \mathbf{X}\beta\|^2 + \lambda\|\beta\|^2\}.$$

The closed form estimate is then:

$$\hat{\beta}_{\lambda} = (\Sigma + \lambda\mathbf{I})^{-1}\left(\frac{1}{n}\mathbf{X}^T Y\right).$$

Note that

$$\hat{\beta}_0 = \hat{\beta}_{\lambda=0} = \underset{\beta}{\text{argmin}}\{\|Y - \mathbf{X}\beta\|^2\},$$

is the ordinary least squares estimator.

Without loss of generality, rotate $\mathbf{X}$ such that:

$$\Sigma = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_p),$$

where the $\lambda_i$'s are ordered in decreasing order.

To see the nature of this shrinkage observe that:

$$[\hat{\beta}_{\lambda}]_j := \frac{\lambda_j}{\lambda_j + \lambda}[\hat{\beta}_0]_j,$$

where $\hat{\beta}_0$ is the ordinary least squares estimator.

Using the bias-variance decomposition, (Equation 1), we have that:

**Lemma 1**

$$\text{Risk}(\hat{\beta}_{\lambda}) = \frac{\sigma^2}{n}\sum_j\left(\frac{\lambda_j}{\lambda_j + \lambda}\right)^2 + \sum_j \beta_j^2 \frac{\lambda_j}{(1 + \frac{\lambda_j}{\lambda})^2}.$$

The proof is straightforward and is provided in the appendix.

## 2. Ordinary Least Squares with PCA (PCA-OLS)

Now let us construct a simple estimator based on $\lambda$. Note that our rotated coordinate system where $\Sigma$ is equal to $diag(\lambda_1, \lambda_2, \ldots, \lambda_p)$ corresponds the PCA coordinate system.

Consider the following ordinary least squares estimator on the "top" PCA subspace — it uses the least squares estimate on coordinate $j$ if $\lambda_j \geq \lambda$ and 0 otherwise

$$[\hat{\beta}_{PCA,\lambda}]_j = \begin{cases} [\hat{\beta}_0]_j & \text{if } \lambda_j \geq \lambda \\ 0 & \text{otherwise} \end{cases}.$$

The following claim shows this estimator compares favorably to the ridge estimator (for every $\lambda$)– no matter how the $\lambda$ is chosen, for example, using cross validation or any other strategy.

Our main theorem (Theorem 2) bounds the Risk Ratio/Risk Inflation[1] of the PCA-OLS and the RR estimators.

**Theorem 2** *(Bounded Risk Inflation) For all $\lambda \geq 0$, we have that:*

$$0 \leq \frac{\text{Risk}(\hat{\beta}_{PCA,\lambda})}{\text{Risk}(\hat{\beta}_\lambda)} \leq 4,$$

*and the left hand inequality is tight.*

**Proof** Using the bias variance decomposition of the risk we can write the risk as:

$$\text{Risk}(\hat{\beta}_{PCA,\lambda}) = \frac{\sigma^2}{n} \sum_j \mathbb{1}_{\lambda_j \geq \lambda} + \sum_{j:\lambda_j < \lambda} \lambda_j \beta_j^2.$$

The first term represents the variance and the second the bias.

The ridge regression risk is given by Lemma 1. We now show that the $j^{th}$ term in the expression for the PCA risk is within a factor 4 of the $j^{th}$ term of the ridge regression risk. First, let's consider the case when $\lambda_j \geq \lambda$, then the ratio of $j^{th}$ terms is:

$$\frac{\frac{\sigma^2}{n}}{\frac{\sigma^2}{n}\left(\frac{\lambda_j}{\lambda_j+\lambda}\right)^2 + \beta_j^2 \frac{\lambda_j}{(1+\frac{\lambda_j}{\lambda})^2}} \leq \frac{\frac{\sigma^2}{n}}{\frac{\sigma^2}{n}\left(\frac{\lambda_j}{\lambda_j+\lambda}\right)^2} = \left(1+\frac{\lambda}{\lambda_j}\right)^2 \leq 4.$$

Similarly, if $\lambda_j < \lambda$, the ratio of the $j^{th}$ terms is:

$$\frac{\lambda_j \beta_j^2}{\frac{\sigma^2}{n}\left(\frac{\lambda_j}{\lambda_j+\lambda}\right)^2 + \beta_j^2 \frac{\lambda_j}{(1+\frac{\lambda_j}{\lambda})^2}} \leq \frac{\lambda_j \beta_j^2}{\frac{\lambda_j \beta_j^2}{(1+\frac{\lambda_j}{\lambda})^2}} = \left(1+\frac{\lambda_j}{\lambda}\right)^2 \leq 4.$$

Since, each term is within a factor of 4 the proof is complete. ■

It is worth noting that the converse is not true and the ridge regression estimator (RR) can be arbitrarily worse than the PCA-OLS estimator. An example which shows that the left hand inequality is tight is given in the Appendix.

1. Risk Inflation has also been used as a criterion for evaluating feature selection procedures (Foster and George, 1994).

## 3. Experiments

First, we generated synthetic data with $p = 100$ and varying values of $n$= {20, 50, 80, 110}. The data was generated in a fixed design setting as $Y = \mathbf{X}\beta + \varepsilon$ where $\varepsilon_i \sim \mathcal{N}(0,1) \ \ \forall i = 1,\ldots,n$. Furthermore, $\mathbf{X}_{n\times p} \sim MVN(\mathbf{0},\mathbf{I})$ where MVN($\mu,\Sigma$) is the Multivariate Normal Distribution with mean vector $\mu$, variance-covariance matrix $\Sigma$ and $\beta_j \sim \mathcal{N}(0,1) \ \ \forall j = 1,\ldots,p$.

The results are shown in Figure 1. As can be seen, the risk ratio of PCA (PCA-OLS) and ridge regression (RR) is never worse than 4 and often its better than 1 as dictated by Theorem 2.

Next , we chose two real world data sets, namely USPS (n=1500, p=241) and BCI (n=400, p=117).[2]

Since we do not know the true model for these data sets, we used all the $n$ observations to fit an OLS regression and used it as an estimate of the true parameter $\beta$. This is a reasonable approximation to the true parameter as we estimate the ridge regression (RR) and PCA-OLS models on a small subset of these observations. Next we choose a random subset of the observations, namely $0.2 \times p$, $0.5 \times p$ and $0.8 \times p$ to fit the ridge regression (RR) and PCA-OLS models.

The results are shown in Figure 2. As can be seen, the risk ratio of PCA-OLS to ridge regression (RR) is again within a factor of 4 and often PCA-OLS is better, that is, the ratio $< 1$.

## 4. Conclusion

We showed that the risk inflation of a particular ordinary least squares estimator (on the "top" PCA subspace) is within a factor 4 of the ridge estimator. It turns out the converse is not true — this PCA estimator may be arbitrarily better than the ridge one.

## Appendix A.

**Proof of Lemma 1.** We analyze the bias-variance decomposition in Equation 1. For the variance,

$$\begin{aligned}
\mathbb{E}_Y \|\hat{\beta}_\lambda - \bar{\beta}_\lambda\|_\Sigma^2 &= \sum_j \lambda_j \mathbb{E}_Y([\hat{\beta}_\lambda]_j - [\bar{\beta}_\lambda]_j)^2 \\
&= \sum_j \frac{\lambda_j}{(\lambda_j+\lambda)^2} \frac{1}{n^2} \mathbb{E}\left[\sum_{i=1}^n (Y_i - \mathbb{E}[Y_i])[X_i]_j \sum_{i'=1}^n (Y_i' - \mathbb{E}[Y_i'])[X_i']_j\right] \\
&= \sum_j \frac{\lambda_j}{(\lambda_j+\lambda)^2} \frac{\sigma^2}{n} \sum_{i=1}^n Var(Y_i)[X_i]_j^2 \\
&= \sum_j \frac{\lambda_j}{(\lambda_j+\lambda)^2} \frac{\sigma^2}{n} \sum_{i=1}^n [X_i]_j^2 \\
&= \frac{\sigma^2}{n} \sum_j \frac{\lambda_j^2}{(\lambda_j+\lambda)^2}.
\end{aligned}$$

2. The details about the data sets can be found here: `http://olivier.chapelle.cc/ssl-book/benchmarks.html`.

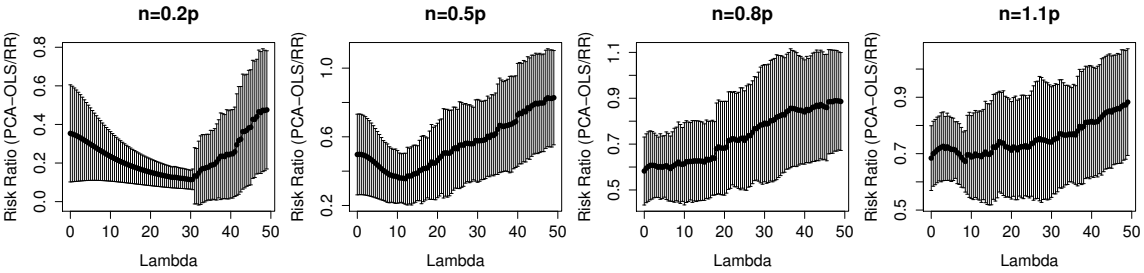



Figure 1: Plots showing the risk ratio as a function of λ, the regularization parameter and $n$, for the synthetic data set. p=100 in all the cases. The error bars correspond to one standard deviation for 100 such random trials.

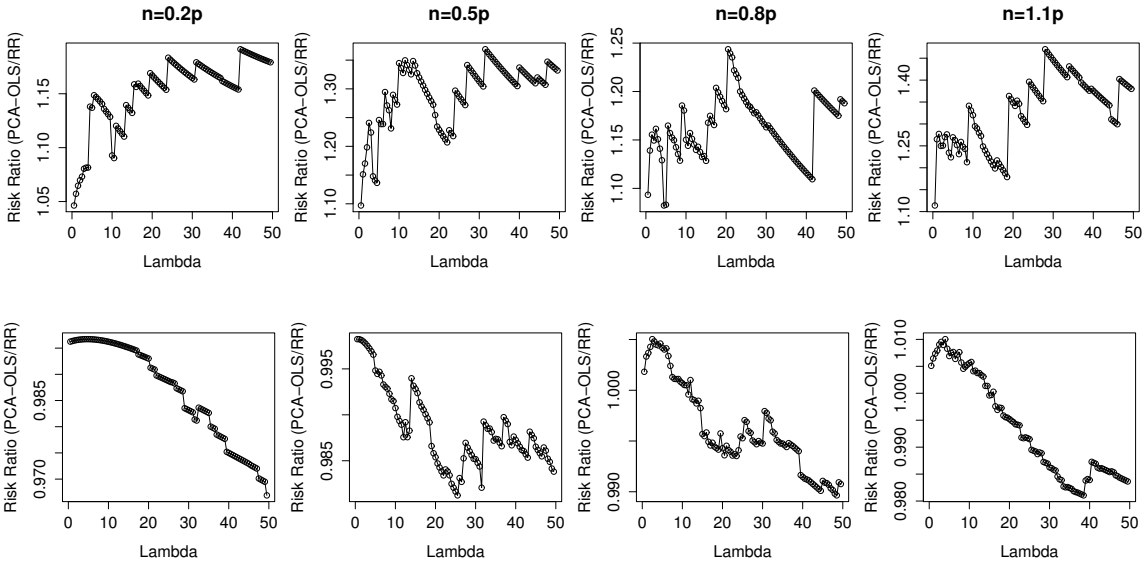



Figure 2: Plots showing the risk ratio as a function of λ, the regularization parameter and $n$, for two real world data sets (BCI and USPS–top to bottom).

Similarly, for the bias,

$$\begin{aligned}\|\bar{\beta}_\lambda - \beta\|_\Sigma^2 &= \sum_j \lambda_j ([\bar{\beta}_\lambda]_j - [\beta]_j)^2 \\ &= \sum_j \beta_j^2 \lambda_j \left(\frac{\lambda_j}{\lambda_j+\lambda} - 1\right)^2 \\ &= \sum_j \beta_j^2 \frac{\lambda_j}{(1+\frac{\lambda_j}{\lambda})^2},\end{aligned}$$

which completes the proof. ■

*The risk for RR can be arbitrarily worse than the PCA-OLS estimator.*

Consider the standard OLS setting described in Section 1 in which $\mathbf{X}$ is $n \times p$ matrix and $Y$ is a $n \times 1$ vector.

Let $\mathbf{X} = diag(\sqrt{1+\alpha}, 1, \ldots, 1)$, then $\mathbf{\Sigma} = \mathbf{X}^\top \mathbf{X} = diag(1+\alpha, 1, \ldots, 1)$ for some $(\alpha > 0)$ and also choose $\beta = [2+\alpha, 0, \ldots, 0]$. For convenience let's also choose $\sigma^2 = n$.

Then, using Lemma 1, we get the risk of RR estimator as

$$\text{Risk}(\hat{\beta}_\lambda) = \left( \underbrace{\left(\frac{1+\alpha}{1+\alpha+\lambda}\right)^2}_{\text{I}} + \underbrace{\frac{(p-1)}{(1+\lambda)^2}}_{\text{II}} \right) + \underbrace{(2+\alpha)^2 \times \frac{(1+\alpha)}{(1+\frac{1+\alpha}{\lambda})^2}}_{\text{III}}.$$

Let's consider two cases

- **Case 1:** $\lambda < (p-1)^{1/3} - 1$, then $II > (p-1)^{1/3}$.
- **Case 2:** $\lambda > 1$, then $1 + \frac{1+\alpha}{\lambda} < 2+\alpha$, hence $III > (1+\alpha)$.

Combining these two cases we get $\forall\lambda$, $\text{Risk}(\hat{\beta}_\lambda) > min((p-1)^{1/3}, (1+\alpha))$. If we choose $p$ such that $p-1 = (1+\alpha)^3$, then $\text{Risk}(\hat{\beta}_\lambda) > (1+\alpha)$.

The PCA-OLS risk (From Theorem 2) is:

$$\text{Risk}(\hat{\beta}_{PCA,\lambda}) = \sum_j \mathbb{1}_{\lambda_j \geq \lambda} + \sum_{j:\lambda_j < \lambda} \lambda_j \beta_j^2.$$

Considering $\lambda \in (1, 1+\alpha)$, the first term will contribute 1 to the risk and rest everything will be 0. So the risk of PCA-OLS is 1 and the risk ratio is

$$\frac{\text{Risk}(\hat{\beta}_{PCA,\lambda})}{\text{Risk}(\hat{\beta}_\lambda)} \leq \frac{1}{(1+\alpha)}.$$

Now, for large $\alpha$, the risk ratio $\approx 0$.

## References

D. P. Foster and E. I. George. The risk inflation criterion for multiple regression. *The Annals of Statistics*, pages 1947–1975, 1994.

A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math Dokl 4*, pages 501–504, 1963.

Journal of Machine Learning Research 14 (2013) 1513-1531 Submitted 10/12; Published 6/13

# On the Learnability of Shuffle Ideals

**Dana Angluin** DANA.ANGLUIN@YALE.EDU
**James Aspnes** JAMES.ASPNES@YALE.EDU
*Department of Computer Science*
*Yale University*
*New Haven, CT 06520 USA*

**Sarah Eisenstat** SEISENST@MIT.EDU
*CSAIL, MIT*
*Cambridge, MA 02139 USA*

**Aryeh Kontorovich** KARYEH@CS.BGU.AC.IL
*Department of Computer Science*
*Ben-Gurion University of the Negev*
*Beer Sheva, Israel 84105*

**Editor:** Mehryar Mohri

## Abstract

PAC learning of unrestricted regular languages is long known to be a difficult problem. The class of shuffle ideals is a very restricted subclass of regular languages, where the shuffle ideal generated by a string $u$ is the collection of all strings containing $u$ as a subsequence. This fundamental language family is of theoretical interest in its own right and provides the building blocks for other important language families. Despite its apparent simplicity, the class of shuffle ideals appears quite difficult to learn. In particular, just as for unrestricted regular languages, the class is not properly PAC learnable in polynomial time if RP $\neq$ NP, and PAC learning the class improperly in polynomial time would imply polynomial time algorithms for certain fundamental problems in cryptography. In the positive direction, we give an efficient algorithm for properly learning shuffle ideals in the statistical query (and therefore also PAC) model under the uniform distribution.

**Keywords:** PAC learning, statistical queries, regular languages, deterministic finite automata, shuffle ideals, subsequences

## 1. Introduction

Inferring regular languages from examples is a classic problem in learning theory. A brief sampling of areas where various automata show up as the underlying formalism include natural language processing (speech recognition, morphological analysis), computational linguistics, robotics and control systems, computational biology (phylogeny, structural pattern recognition), data mining, time series and music (Koskenniemi, 1983; de la Higuera, 2005; Mohri, 1996; Mohri et al., 2002; Mohri, 1997; Mohri et al., 2010; Rambow et al., 2002; Sproat et al., 1996). Thus, developing efficient formal language learning techniques and understanding their limitations is of a broad and direct relevance in the digital realm.

Perhaps the currently most widely studied theoretical model of learning is Valiant's PAC model, which allows for a clean, elegant theory while retaining some measure of empirical plausibility (Valiant, 1984). Since PAC learnability is characterized by finite VC-dimension and the concept

©2013 Dana Angluin, James Aspnes, Sarah Eisenstat and Aryeh Kontorovich.

class of $n$-state deterministic finite state automata (DFA) has VC-dimension $\Theta(n \log n)$ (Ishigami and Tani, 1997), the PAC learning problem is solved, in an information theoretic sense, by constructing a DFA on $n$ states consistent with a given labeled sample. Unfortunately, as shown in the works of Angluin (1978), Gold (1978) and Pitt and Warmuth (1993) under standard complexity assumptions, finding small consistent automata is a computationally intractable task. Furthermore, attempts to circumvent the combinatorial search over automata by learning with a different representation class are thwarted by cryptographic hardness results. The papers of Pitt and Warmuth (1990) and Kearns and Valiant (1994) prove the existence of small automata and "hard" distributions over $\{0,1\}^n$ so that any efficient learning algorithm that achieves a polynomial advantage over random guessing will break various cryptographic hardness assumptions.

In a modified model of PAC, and with additional structural assumptions, a class of probabilistic finite state automata was shown by Clark and Thollard (2004) and Palmer and Goldberg (2007) to be learnable. If the target automaton and sampling distribution are assumed to be "simple", efficient probably exact learning is possible (Parekh and Honavar, 2001). When the learner is allowed to make membership queries, it follows by the results of Angluin (1987) that DFAs are learnable in this augmented PAC model.

The prevailing paradigm in regular language learning has been to make structural regularity assumptions about the family of languages and/or the sampling distribution in question and to employ a state merging heuristic. Indeed, over the years a number of clever and sophisticated combinatorial approaches have been proposed for learning DFAs. Typically, an initial automaton or prefix tree consistent with the sample is first created. Then, starting with the trivial partition with one state per equivalence class, classes are merged while preserving an invariant congruence property. The automaton learned is obtained by merging states according to the resulting classes. Thus, the choice of the congruence determines the algorithm and generalization bounds are obtained from the structural regularity assumptions. This rough summary broadly characterizes the techniques of Angluin (1982), Oncina and García (1992), Ron et al. (1998), Clark and Thollard (2004), Parekh and Honavar (2001) and Palmer and Goldberg (2007), and until recently this appears to have been the only general purpose technique available for learning finite automata.

More recently, Kontorovich et al. (2006), Cortes et al. (2007) and Kontorovich et al. (2008) proposed a substantial departure from the state merging paradigm. Their approach was to embed a specific family of regular languages (the piecewise-testable ones) in a Hilbert space via a kernel and to identify languages with hyperplanes. A unifying feature of this methodology is that rather than building an automaton, the learning algorithm outputs a classifier defined as a weighted sum of simple automata. In subsequent work by Kontorovich and Nadler (2009) this approach was extended to learning general discrete concepts. These results, however, provided only margin based generalization guarantees, which are weaker than true PAC bounds.

A promising research direction is to investigate the question of efficient PAC learnability for restricted subclasses of the regular sets. One approach is to take existing efficient PAC algorithms in other domains, for example, for classes of propositional formulas over the boolean cube $\{0,1\}^n$, or classes of geometric concepts such as axis-aligned boxes in $\mathbb{R}^n$, discretize the representation if necessary, and consider the resulting sets of strings to be formal languages. If the languages have finite cardinality, they are trivially regular, although they may or may not have succinct deterministic finite state acceptors.

Another approach is to consider classes of regular languages defined by structural restrictions on the automata or grammars that accept or generate them. Ergün et al. (1995) consider the learnability

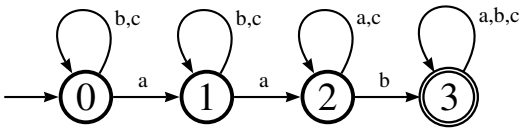



Figure 1: The canonical DFA for recognizing the shuffle ideal of $u = aab$ over $\Sigma = \{a,b,c\}$, which accepts precisely those strings that contain $u$ as a subsequence.

of bounded-width branching programs, and show that there is an efficient algorithm to PAC learn width-2 branching programs, though not properly, and an efficient proper PAC learning algorithm for width-2 branching programs with respect to the uniform distribution. They also show that PAC learning width-3 branching programs is as hard as PAC learning DNF formulas, a problem whose status remains open.

In this paper we study the PAC learnability of another restricted class of regular languages, the shuffle ideals. The shuffle ideal generated by a string $u$ is the collection of all strings containing $u$ as a (not necessarily contiguous) subsequence (see Figure 1 for an illustration). Despite being a particularly simple subfamily of the regular languages, shuffle ideals play a prominent role in formal language theory. Their boolean closure forms the important family known as *piecewise-testable* languages, defined and characterized by Simon (1975). The rich structure of this language family has made it an object of intensive study, with deep connections to computability, complexity theory, and semigroups (see the papers of Lothaire (1983) and Klíma and Polák (2008) and the references therein). On a more applied front, the shuffle ideals capture some rudimentary phenomena in human language morphology (Kontorovich et al., 2003).

In Section 3 we show that shuffle ideals of known length are exactly learnable in the statistical query model under the uniform distribution, though not efficiently. Permitting approximate learning, the algorithm can be made efficient; this in turn yields efficient proper PAC learning under the uniform distribution. On the other hand, in Section 4 we show that the shuffle ideals are not properly PAC learnable under general distributions unless RP=NP. In Section 5 we show that a polynomial time improper PAC learning algorithm for the class of shuffle ideals would imply the existence of polynomial time algorithms to break the RSA cryptosystem, factor Blum integers, and test quadratic residuosity. These two negative results are analogous to those for general regular languages represented by deterministic finite automata.

## 2. Preliminaries

Throughout this paper, we consider a fixed finite alphabet $\Sigma$, whose size will be denoted by $s$. We assume $s \geq 2$. The elements of $\Sigma^*$ will be referred to as *strings* with their length denoted by $|\cdot|$; the empty string is $\lambda$. The concatenation of strings $u_1$ and $u_2$ is denoted by $u_1 \cdot u_2$ or $u_1 u_2$. The string $u$ is a *prefix* of a string $v$ if there exists a string $w$ such that $v = uw$. Similarly, $u$ is a *suffix* of $v$ if there exists a string $w$ such that $v = wu$. We use exponential notation for repeated concatenation of a string with itself, that is, $u^n$ is the concatenation of $n$ copies of $u$.

Define the binary relation $\sqsubseteq$ on $\Sigma^*$ as follows: $u \sqsubseteq v$ holds if there is a witness $\vec{i} = (i_1 < i_2 < \ldots < i_{|u|})$ such that $v_{i_j} = u_j$ for all $j \in [|u|]$. When there are several witnesses for $u \sqsubseteq v$, we may partially order them coordinate-wise, referring to the unique minimal element as the *leftmost* embedding. The unique maximal element is the *rightmost* embedding. If $u \sqsubseteq v$ then the *leftmost span* of $u$ in $v$

is the shortest prefix $v_1$ of $v$ such that $u \sqsubseteq v_1$ and the *rightmost span* of $u$ in $v$ is the shortest suffix $v_2$ of $v$ such that $u \sqsubseteq v_2$.

Formally, the (principal) *shuffle ideal* generated by $u \in \Sigma^\ell$ is the regular language

$$\text{Ш}(u) = \{x \in \Sigma^* : u \sqsubseteq x\} = \Sigma^* u_1 \Sigma^* u_2 \Sigma^* \ldots \Sigma^* u_\ell \Sigma^*$$

(an example is given in Figure 1). The shuffle ideal of string $u$ consists of all strings $v$ over the given alphabet such that $u \sqsubseteq v$. The term *shuffle ideal* comes from algebra (Lothaire, 1983; Păun, 1994) and dates back to the paper of Eilenberg and Mac Lane (1953).

The following lemmas will be useful in the sequel. The first is immediate from the definitions; the second formalizes the obvious method of determining whether $u \sqsubseteq v$ and finding a leftmost embedding if so.

**Lemma 1** *Suppose $u = u_1u_2u_3$ and $v = v_1v_2v_3$ are strings such that $u \sqsubseteq v$ and $v_1$ is the leftmost span of $u_1$ in $v$ and $v_3$ is the rightmost span of $u_3$ in $v$. Then $u_2 \sqsubseteq v_2$.*

**Lemma 2** *Evaluating the relation $u \sqsubseteq x$ is feasible in time $O(|x|)$.*

**Proof** If $u = \lambda$, then $u$ is certainly a subsequence of $x$. If $u = au'$ where $a \in \Sigma$, we search for the leftmost occurrence of $a$ in $x$. If there is no such occurrence, then $u$ is certainly not a subsequence of $x$. Otherwise, we write $x = yax'$, where $y$ contains no occurrence of $a$; then $u$ is a subsequence of $x$ if and only if $u'$ is a subsequence of $x'$, so we continue recursively with $u'$ and $x'$. The total time for this algorithm is $O(|x|)$. ■

We assume a familiarity with the basics of the PAC learning model, as defined in the textbook of Kearns and Vazirani (1994). To recap, consider the instance space $\mathcal{X} = \Sigma^*$, concept class $\mathcal{C} \subseteq 2^{\mathcal{X}}$, and hypothesis class $\mathcal{H} \subseteq 2^{\mathcal{X}}$. An *algorithm* $\mathcal{L}$ is given access to a labeled sample $S = (X_i, Y_i)_{i=1}^m$, where the $X_i$ are drawn iid from some unknown distribution $P$ over $\mathcal{X}$ and $Y_i = f(X_i)$ for some unknown *target* $f \in \mathcal{C}$, and produces a *hypothesis* $h \in \mathcal{H}$. We say that $\mathcal{L}$ efficiently PAC learns $\mathcal{C}$ if for any $\varepsilon, \delta > 0$ there is an $m_0 \in \mathbb{N}$ such that for all $f \in \mathcal{C}$ and all distributions $P$, the hypothesis $h_m$ generated by $\mathcal{L}$ based on a sample of size $m \geq m_0$ satisfies

$$P^m[P(\{x \in \mathcal{X} : h_m(x) \neq f(x)\}) > \varepsilon] < \delta;$$

moreover, we require that both $m_0$ and $\mathcal{L}$'s runtime be at most polynomial in $\varepsilon^{-1}$, $\delta^{-1}$ and the sizes of $f$ and $X_i$. The learning is said to be *proper* if $\mathcal{H} = \mathcal{C}$ and *improper* otherwise. If the learning algorithm achieves $\varepsilon = 0$, the learning is said to be *exact* (Bshouty, 1997; Bshouty et al., 2005).

Most learning problems can be cleanly decomposed into a computational and an information theoretic component. The information theoretic aspects of learning automata are well understood. As mentioned above, the VC-dimension of a collection of DFAs grows polynomially with maximal number of states, and so any small DFA consistent with the training sample will, with high probability, have small generalization error. For shuffle ideals, an even simpler bound can be derived. If $n$ is an upper bound on the length of the string $u \in \Sigma^*$ generating the target shuffle ideal, then our concept class contains exactly

$$\sum_{\ell=0}^{n} |\Sigma|^\ell = O(|\Sigma|^n)$$

members. Thus, with probability at least $1-\delta$, any shuffle ideal consistent with a sample of size $m$ will achieve a generalization error of

$$O\left(\frac{n\log|\Sigma|-\log\delta}{m}\right).$$

Hence, the problem of properly PAC learning shuffle ideals has been reduced to finding one that is consistent with a given sample. This is shown to be computationally hard under adversarial distributions (Theorem 7), but feasible under the uniform one (Theorem 6). Actually, our positive result is somewhat stronger: since we show learnability in the statistical query (SQ) model of Kearns (1998), this implies a noise tolerant PAC result. In addition, in Section 5 we show that the existence of a polynomial time improper PAC learning algorithm for shuffle ideals would imply the existence of polynomial time algorithms for certain cryptographic problems.

## 3. SQ Learning Under the Uniform Distribution

The main result of this section is that shuffle ideals are efficiently PAC learnable under the uniform distribution. To be more precise, we are dealing with the instance space $X=\Sigma^n$ endowed with the uniform distribution, which assigns a weight of $|\Sigma|^{-n}$ to each element of $X$. Our learning algorithm is most naturally expressed in the language of *statistical queries* (Kearns, 1998; Kearns and Vazirani, 1994). In the original definition, a statistical query $\chi$ is a binary predicate of a random instance-label pair, and the oracle returns the value $\mathbf{E}\chi$, additively perturbed by some amount not exceeding a specified tolerance parameter. We will consider a somewhat richer class of queries.

### 3.1 Constructing and Analyzing the Queries

For $u\in\Sigma^{\le n}$ and $a\in\Sigma$, we define the query $\chi_{u,a}(\cdot,\cdot)$ by

$$\chi_{u,a}(x,y)=\begin{cases}0, & u\not\sqsubseteq x'\\ y(\mathbb{1}_{\{\sigma=a\}}-\mathbb{1}_{\{\sigma\neq a\}}/(s-1)), & u\sqsubseteq x'\end{cases}$$

where $x'$ is the prefix of $x$ of length $(n-1)$, $\sigma$ is the symbol in $x$ following the leftmost embedding of $u$ and $\mathbb{1}_{\{\pi\}}$ represents the 0-1 truth value of the predicate $\pi$ (recall that $s=|\Sigma|$). Our definition of the query $\chi_{u,a}$ is legitimate because (i) it can be efficiently evaluated (Lemma 2) and (ii) it can be expressed as a linear combination of $O(1)$ standard binary queries (also efficiently computable). In words, the function $\chi_{u,a}$ computes the mapping $(x,y)\mapsto\mathbb{R}$ as follows. If $u$ is not a subsequence of $x'$, $\chi_{u,a}(x,y)=0$. Otherwise, $\chi_{u,a}$ checks whether the symbol $\sigma$ in $x$ following the leftmost embedding of $u$ is equal to $a$, and, if $x$ is a positive example ($y=+1$), returns 1 if $\sigma=a$, or $-1/(s-1)$ if $\sigma\neq a$. If $x$ is a negative example ($y=-1$) then the signs of the values returned are inverted.

Suppose for now that the length $L=|\bar{u}|$ of the target shuffle ideal $\bar{u}$ is known. Our learning algorithm uses statistical queries to recover $\bar{u}\in\Sigma^L$ one symbol at a time. It starts with the empty string $u=\lambda$. Having recovered $u=\bar{u}_1,\ldots,\bar{u}_\ell$, $\ell<L$, we infer $\bar{u}_{\ell+1}$ as follows. For each $a\in\Sigma$, the SQ oracle is called with the query $\chi_{u,a}$ and a tolerance $0<\tau<1$ to be specified later. Our key technical observation is that the value of $\mathbf{E}\chi_{u,a}$ effectively selects the next symbol of $\bar{u}$:

**Lemma 3**

$$\mathbf{E}\chi_{u,a}=\begin{cases}+\frac{2}{s}P(L,n,s), & a=\bar{u}_{\ell+1}\\ -\frac{2}{s(s-1)}P(L,n,s), & a\neq\bar{u}_{\ell+1}\end{cases}$$

*where*

$$P(L,n,s) = \binom{n-1}{L-1}\left(\frac{1}{s}\right)^{L-1}\left(1-\frac{1}{s}\right)^{n-L}.$$

**Proof** Fix an unknown string $\bar{u}$ of length $L \geq 1$; by assumption, we have recovered in $u = u_1 \ldots u_\ell = \bar{u}_1 \ldots \bar{u}_\ell$ the first $\ell$ symbols of $\bar{u}$. Let $u' = \bar{u}0^\infty$ be the extension of $\bar{u}$ obtained by padding it on the right with infinitely many 0 symbols (we assume $0 \in \Sigma$).

Let $X$ be a random variable representing the uniformly chosen sample string $x$. Let $T$ be the largest value for which $u'_1 \ldots u'_T$ is a subsequence of $X$. Let $\xi = \mathbb{1}_{\{T \geq L\}}$ be the indicator for the event that $X$ is a positive instance, that is, that $\bar{u}_1 \ldots \bar{u}_L = u'_1 \ldots u'_L$ is a subsequence of $X$.

Observe that $T$ has a binomial distribution:

$$T \sim \text{Binom}(n, 1/s);$$

indeed, as we sweep across $X$, each position $X_i$ has a $1/s$ chance of being the next unused symbol of $u'$. An immediate consequence of this fact is that $\Pr[\xi = 1]$ is exactly $\sum_{k=L}^{n}\binom{n}{k}(1/s)^k(1-1/s)^{n-k}$.

Now fix $\ell < L$ and let $I_\ell$ be defined as follows. If $\ell = 0$ then $I_\ell = 0$, and if $u_1 \ldots u_\ell$ is not a subsequence of $X_1 \ldots X_{n-1}$ then $I_\ell = n-1$. Otherwise, $I_\ell$ is the position of $u_\ell$ in the leftmost embedding of $u_1 \ldots u_\ell$ in $X_1 \ldots X_{n-1}$. Then $I_\ell + 1$ is the position of $\sigma$ as defined in (3.1), or $n$ if $u_1 \ldots u_\ell \not\sqsubseteq X_1 \ldots X_{n-1}$.

We define two additional random variables, $T_A$ and $T_B$. $T_A$ is the length of the longest prefix of $u'$ that is a subsequence of $X$ with $X_{I_\ell+1}$ excluded:

$$T_A = \max\left\{t : u'_1 \ldots u'_t \sqsubseteq X_1 \ldots X_{I_\ell} X_{I_\ell+2} \ldots X_n\right\}.$$

Intuitively, $T_B$ is the length of the longest prefix of $u'$ with $u'_{\ell+1}$ excluded that is a subsequence of $X$ with $X_{I_\ell+1}$ excluded. Formally, let $v_1 v_2 \ldots$ be the sequence $u'_1 u'_2 \ldots$ with the element $u'_{\ell+1}$ excluded, that is, $v_i = u'_i$ if $i \leq \ell$ and $v_i = u'_{i+1}$ if $i \geq \ell+1$.

$$T_B = \max\left\{t : v_1 \ldots v_t \sqsubseteq X_1 \ldots X_{I_\ell} X_{I_\ell+2} \ldots X_n\right\}.$$

Like $T$, $T_A$ and $T_B$ are binomially distributed, but now

$$T_A, T_B \sim \text{Binom}(n-1, 1/s).$$

The reason is that we always omit one position in $X$ (the one following $u_\ell$ if $u_\ell$ appears before $X_n$ or $X_n$ if it does not), and for each other position, there is still an independent $1/s$ chance that it is the next symbol in $u'$ (or $u'$ with $u'_{\ell+1}$ excluded.)

An important fact is that $X_{I_\ell+1}$ is independent of the values of $T_A$ and $T_B$, though of course $T_A$ and $T_B$ are not independent of each other. This is not immediately obvious: whether $X_{I_\ell+1}$ equals $u'_{\ell+1}$ or not affects the interpretation of later symbols in $X$. However, the probability that each symbol $X_{I_\ell+2} \ldots$ is the next unused symbol in $u'$ (or $v$) is still an independent $1/s$ whether $X_{I_\ell+1}$ consumes a symbol of $u'$ (or $v$) or not. The joint distribution of $T_A$ and $T_B$ is not affected.

We now compute $\mathbf{E}\chi_{u,a}$ by averaging over the choices in the joint distribution of $T_A$ and $T_B$. If $T_A \geq L$, then $\bar{u}$ is a subsequence of $X_1 \ldots X_{I_\ell} X_{I_\ell+2} \ldots X_n$, and $X$ is a positive example ($y = +1$) no matter how $X_{I_\ell+1}$ is chosen. In this case, each symbol in $\Sigma$ contributes 1 to the conditional expected value with probability $1/s$ and $-\frac{1}{s-1}$ with probability $\frac{s-1}{s}$; the net contribution is 0.

If $X$ is a positive example, then $\bar{u}$ is a subsequence of $X$ and a leftmost embedding of $\bar{u}$ in $X$ embeds $u_1 \dots u_\ell$ in $X_1 \dots X_{I_\ell}$ and embeds $u_{\ell+1} \dots u_L$ in $X_{I_\ell+1} \dots X_n$. Thus, no matter what symbol is chosen for $X_{I_\ell+1}$, $u_{\ell+2} \dots u_L$ is a subsequence of $X_{I_\ell+2} \dots X_n$, and $T_B$ must be at least $L-1$. Thus, if $T_A \geq L$ then $T_B \geq L-1$. Moreover, if $T_B < L-1$, $X$ must be a negative example ($y=-1$) no matter how $X_{I_\ell+1}$ is chosen. In this case, the probability-$(1/s)$ contribution of $-1$ is exactly offset by the probability-$\left(\frac{s-1}{s}\right)$ contribution of $\frac{1}{s-1}$, and the conditional expected value is 0.

Thus the only case in which there may be a non-zero contribution to the expected value is when $T_A < L$ and $T_B \geq L-1$, that is, when the choice of $X_{I_\ell+1}$ may affect the label of $X$. The example $X$ is positive if and only if $X_{I_\ell+1} = \bar{u}_{\ell+1}$, which occurs if $\sigma = \bar{u}_{\ell+1}$. Thus the conditional expectation for $a = \bar{u}_{\ell+1}$ is

$$1 \cdot \Pr[\sigma = \bar{u}_{\ell+1}] + \frac{1}{s-1} \cdot \Pr[\sigma \neq \bar{u}_{\ell+1}] = \frac{1}{s} + \frac{1}{s-1} \cdot \frac{s-1}{s} = 2/s.$$

For $a \neq \bar{u}_{\ell+1}$, the conditional expectation is is $-\frac{2}{s(s-1)}$. This can be computed directly by considering cases, or by observing that the change to $\sum_{a \in \Sigma} \chi_{u,a}(x) = 0$ always, and that all $a \neq \bar{u}_{\ell+1}$ induce same expectation by symmetry.

Finally we need to determine $\Pr[T_A < L \wedge T_B \geq L-1]$. We may write

$$\Pr[T_B \geq L-1 \wedge T_A < L] = \Pr[T_B \geq L-1] - \Pr[T_B \geq L-1 \wedge T_A \geq L]$$

because $T_A \geq L$ implies $T_B \geq L-1$,

$$\Pr[T_B \geq L-1 \wedge T_A \geq L] = \Pr[T_A \geq L],$$

and thus

$$\Pr[T_B \geq L-1 \wedge T_A < L] = \Pr[T_B \geq L-1] - \Pr[T_A \geq L].$$

Because $T_A$ and $T_B$ are binomially distributed, $\Pr[T_B \geq L-1 \wedge T_A < L]$ is

$$\sum_{i=L-1}^{n-1} \binom{n-1}{i} \left(\tfrac{1}{s}\right)^i \left(1-\tfrac{1}{s}\right)^{n-1-i} - \sum_{i=L}^{n-1} \binom{n-1}{i} \left(\tfrac{1}{s}\right)^i \left(1-\tfrac{1}{s}\right)^{n-1-i}$$

which is

$$\binom{n-1}{L-1} \left(\tfrac{1}{s}\right)^{L-1} \left(1-\tfrac{1}{s}\right)^{n-L} = P(L,n,s).$$

This concludes the proof of Lemma 3. ■

### 3.2 Specifying the Query Tolerance $\tau$

The analysis in Lemma 3 implies that to identify the next symbol of $\bar{u} \in \Sigma^L$ it suffices to distinguish the two possible expected values of $\mathbf{E}\chi_{u,a}$, which differ by $(2/(s-1))P(L,n,s)$. If the query tolerance is set to one third of this value, that is,

$$\tau = \frac{2}{3(s-1)} P(L,n,s)$$

then $s$ statistical queries for each prefix of $\bar{u}$ suffice to learn $\bar{u}$ exactly.

**Theorem 4** *When the length $L$ of the target string $\bar{u}$ is known, $\bar{u}$ is exactly identifiable with $O(Ls)$ statistical queries at tolerance $\tau = \frac{2}{3(s-1)}P(L,n,s)$.*

In the above SQ algorithm there is no need for a precision parameter $\varepsilon$ because the learning is *exact*, that is, $\varepsilon = 0$. Nor is there a need for a confidence parameter $\delta$ because each statistical query is guaranteed to return an answer within the specified tolerance, in contrast to the PAC setting where the parameter $\delta$ protects the learner against an "unlucky" sample.

However, if the relationship between $n$ and $L$ is such that $P(L,n,s)$ is very small, then the tolerance $\tau$ will be very small, and this first SQ algorithm cannot be considered efficient. If we allow an approximately correct hypothesis ($\varepsilon > 0$), we can modify the above algorithm to use a polynomially bounded tolerance.

**Theorem 5** *When the length $L$ of the target string $\bar{u}$ is known, $\bar{u}$ is approximately identifiable to within $\varepsilon > 0$ with $O(Ls)$ statistical queries at tolerance $\tau = 2\varepsilon/(9(s-1)n)$.*

**Proof** We modify the SQ algorithm to make an initial statistical query with tolerance $\varepsilon/3$ to estimate $\Pr[\xi = 1]$, the probability that $x$ is a positive example. If the answer is $\leq 2\varepsilon/3$, then $\Pr[\xi = 1] \leq \varepsilon$ and the algorithm outputs a hypothesis that classifies all examples as negative. If the answer is $\geq 1 - 2\varepsilon/3$, then $\Pr[\xi = 1] \geq 1 - \varepsilon$ and the algorithm outputs a hypothesis that classifies all examples as positive.

Otherwise, $\Pr[\xi = 1]$ and $\Pr[\xi = 0]$ are both at least $\varepsilon/3$, and the first SQ algorithm is used. We now show that $P(L,n,s) \geq \varepsilon/(3n)$, establishing the bound on the tolerance. Let $Q(L,n,s) = \binom{n}{L}\left(\frac{1}{s}\right)^L\left(1-\frac{1}{s}\right)^{n-L}$ and note that $Q(L,n,s) = (n/Ls)P(L,n,s)$. If $L \leq n/s$ then $Q(L,n,s)$ is at least as large as every term in the sum

$$\Pr[\xi = 0] = \sum_{k=0}^{L-1} \binom{n}{k} \left(\frac{1}{s}\right)^k \left(1-\frac{1}{s}\right)^{n-k}$$

and therefore $Q(L,n,s) \geq \varepsilon/(3L)$ and $P(L,n,s) \geq \varepsilon/(3n)$. If $L > n/s$ then $Q(L,n,s)$ is at least as large as every term in the sum

$$\Pr[\xi = 1] = \sum_{k=L}^{n} \binom{n}{k} \left(\frac{1}{s}\right)^k \left(1-\frac{1}{s}\right)^{n-k}$$

and therefore $P(L,n,s) \geq Q(L,n,s) \geq \varepsilon/(3n)$. ■

### 3.3 PAC Learning

The main result of this section is now obtained by a standard transformation of an SQ algorithm to a PAC algorithm.

**Theorem 6** *The concept class $\mathcal{C} = \left\{\text{Ш}(u) : u \in \Sigma^{\leq n}\right\}$ is efficiently properly PAC learnable under the uniform distribution.*

**Proof** We assume that the algorithm receives as inputs $n$, $L$, $\varepsilon$ and $\delta$. Because there are only $n+1$ choices of $L$, a standard method may be used to iterate through them. We simulate the modified SQ

algorithm by drawing a sample of labeled examples and using them to estimate the answers to the $O(Ls)$ calls to the SQ oracle with queries at tolerance $\tau = 2\varepsilon/(9(s-1)n)$, as described by Kearns (1998). According to the result of Kearns (1998, Theorem 1),

$$O\left(\frac{1}{\tau^2}\log\frac{|C|}{\delta}\right) = O\left(\frac{s^2n^2}{\varepsilon^2}(n\log s - \log\delta)\right)$$

examples suffice to determine correct answers to all the queries at the desired tolerance, with probability at least $1-\delta$. ■

Our learning algorithm and analysis are rather strongly tied to the uniform distribution. If this assumption is omitted, it might now happen that $\Pr[T_B \geq L-1 \wedge T_A < L]$ is small even though positive and negative examples are mostly balanced, or there might be intractable correlations between $\sigma$ and the values of $T_A$ and $T_B$. It seems that genuinely new ideas will be required to handle nonuniform distributions.

## 4. Proper PAC Learning Under General Distributions Is Hard Unless NP=RP

This hardness result follows a standard paradigm (see Kearns and Vazirani, 1994). We show that the problem of deciding whether a given labeled sample admits a consistent shuffle ideal is NP-complete. A standard argument then shows that any proper PAC learner for shuffle ideals can be efficiently manipulated into solving the decision problem, yielding an algorithm in RP. Thus, assuming RP $\neq$ NP, there is no polynomial time algorithm that properly learns shuffle ideals.

**Theorem 7** *For any alphabet of size at least 2, given two disjoint sets of strings $S, T \subset \Sigma^*$, the problem of determining whether there exists a string $u$ such that $u \sqsubseteq x$ for each $x \in S$ and $u \not\sqsubseteq x$ for each $x \in T$ is NP-complete.*

We first prove a lemma that facilitates the representation of $n$ independent binary choices. Let $\Sigma = \{0,1\}$, let $n$ be a positive integer and define $A_n$ to be the set of $2^n$ binary strings described by the regular expression

$$((00000 + 00100)11)^n.$$

Define strings

$$\begin{aligned} v_0 &= 000100, \\ v_1 &= 001000, \\ d &= 11, \end{aligned}$$

and let $S_n$ consist of the two strings

$$\begin{aligned} s_0 &= (v_0 d)^n, \\ s_1 &= (v_1 d)^n. \end{aligned}$$

Define the strings

$$\begin{aligned} y_0 &= 00010, \\ y_1 &= 01000, \\ z &= 0000, \\ d_0 &= 1 \end{aligned}$$

and for each integer $i$ such that $1 \leq i \leq n$, define the strings

$$\begin{aligned} t_{i,0} &= (v_0 d)^{i-1} y_0 d (v_0 d)^{n-i}, \\ t_{i,1} &= (v_0 d)^{i-1} y_1 d (v_0 d)^{n-i}, \\ t_{i,2} &= (v_0 d)^{i-1} z d (v_0 d)^{n-i}, \\ t_{i,3} &= (v_0 d)^{i-1} v_0 d_0 (v_0 d)^{n-i}. \end{aligned}$$

The strings $t_{i,0}$, $t_{i,1}$ and $t_{i,2}$ are obtained from $s_0$ by replacing occurrence $i$ of $v_0$ by $y_0$, $y_1$, and $z$, respectively. The string $t_{i,3}$ is obtained from $s_0$ by replacing occurrence $i$ of $d$ by $d_0$. Let $T_n$ consist of all the strings $t_{i,j}$ for $1 \leq i \leq n$ and $0 \leq j \leq 3$.

The following lemma shows that the set of strings consistent with $S_n$ and $T_n$ is precisely the $2^n$ strings in $A_n$.

**Lemma 8** *Let $C_n$ be the set of strings $u$ such that $u$ is a subsequence of both strings in $S_n$ and not a subsequence of any string in $T_n$. Then $C_n = A_n$.*

**Proof** We first observe that for any positive integer $m$ and any string $u \in A_m$, the leftmost span of $u$ in $(v_0 d)^m$ is $(v_0 d)^m$ itself, and the leftmost span of $u$ in $(v_1 d)^m$ is $(v_1 d)^m$ itself. For $m = 1$, we have $u = 0000011$ or $u = 0010011$, while $v_0 d = 00010011$ and $v_1 d = 00100011$, and the result holds by inspection. Then a straightforward induction establishes the result for $m > 1$. Similarly, for any string $u \in A_m$, the rightmost span of $du$ in $d(v_0 d)^m$ is $d(v_0 d)^m$ itself, and the rightmost span of $du$ in $d(v_1 d)^m$ is $d(v_1 d)^m$ itself. In the base case we have $du = 110000011$ or $du = 110010011$, while $dv_0 d = 1100010011$ and $dv_1 d = 1100100011$, and the result holds by inspection. A straightforward induction establishes the result for $m > 1$.

Suppose $u \in A_n$. Then

$$u = u_1 d u_2 d \cdots u_n d,$$

where each $u_i$ is either 00000 or 00100. Clearly $u \sqsubseteq s_0$ and $u \sqsubseteq s_1$, because 00000 and 00100 are subsequences of $v_0$ and $v_1$.

Consider a string $t_{i,0} \in T_n$. Suppose that $u \sqsubseteq t_{i,0}$. Divide $u$ into three parts, $u = u' u_i u''$, where $u'$ is $u_1 d \cdots u_{i-1} d$ and $u'' = d u_{i+1} \cdots u_n d$. The leftmost span of $u'$ in $t_{i,0}$ is $(v_0 d)^{i-1}$, and the rightmost span of $u''$ in $t_{i,0}$ is $d(v_0 d)^{n-i}$, which implies that $u_i \sqsubseteq y_0$ by Lemma 1. But $u_i$ is either 00000 or 00100 and $y_0$ is 00010, which is a contradiction. So $u$ is not a subsequence of $t_{i,0}$. Similar arguments show that $u$ is not a subsequence of $t_{i,1}$ or $t_{i,2}$.

Now suppose $u \sqsubseteq t_{i,3}$. We divide $u$ into parts, $u = u' u_i d u_{i+1} u''$, where $u' = u_1 d \cdots u_{i-1} d$ and $u'' = d u_{i+2} \cdots u_n d$. The leftmost span of $u'$ in $t_{i,3}$ is $(v_0 d)^{i-1}$ and the rightmost span of $u''$ in $t_{i,3}$ is $d(v_0 d)^{n-i-1}$. By Lemma 1, we must have

$$u_i d u_{i+1} \sqsubseteq v_0 d_0 v_0.$$

That is, at least one of the strings

$$000001100000, 001001100000, 000001100100, 001001100100$$

must be a subsequence of 0001001000100, which is false, showing that $u$ is not a subsequence of $t_{i,3}$. Thus $u$ is not a subsequence of any string in $T_n$, and $u \in C_n$. Thus $A_n \subseteq C_n$.

For the reverse direction, suppose $u \in C_n$. We consider an embedding of $u$ in $s_0$ and divide $u$ into segments

$$u = u_1 d_1 u_2 d_2 \cdots u_n d_n,$$

where for each $i$, $u_i \sqsubseteq v_0$ and $d_i \sqsubseteq d$. If for any $i$ we have $d_i \sqsubseteq 1$, then $u \sqsubseteq t_{i,3}$, a contradiction. Thus $d_i = 11 = d$ for every $i$. Similarly, if $u_i$ is a subsequence of $y_0$, $y_1$ or $z$, then $u$ is a subsequence of $t_{i,0}$, $t_{i,1}$, or $t_{i,2}$, respectively, so we know that each $u_i$ is a subsequence of the string 000100, but not a subsequence of the strings 00010, 01000, or 0000. It is not difficult to check that the only possibilities for $u_i$ are

$$00000, 00100, 000100.$$

To eliminate the third possibility we use the fact that $u$ is a subsequence of $s_1$. Consider any string

$$w = w_1 d w_2 d \cdots w_n d,$$

where $w_i = 000100$ and each $w_j$ for $j \neq i$ is either 00000 or 00100. We may divide $w$ into parts $w = w'000100w''$ where $w' = w_1 d \cdots w_{i-1} d$ and $w'' = d w_{i+1} d \cdots w_n d$. If $w \sqsubseteq s_1$, then the leftmost span of $w'$ in $s_1$ is $(v_1 d)^{i-1}$, and the rightmost span of $w''$ in $s_1$ is $d(v_1 d)^{n-i}$, which by Lemma 1 means that 000100 must be a subsequence of $v_1 = 001000$, a contradiction. Thus no such $w$ is a subsequence of $s_1$, and we must have $u_i$ equal to 00000 or 00100 for all $i$, that is, $u$ must be in $A_n$. Thus $C_n \subseteq A_n$. ■

We now prove Theorem 7.

**Proof** To see that this decision problem is in NP, note that if $S$ is empty, then any string of length longer than the longest string in $T$ satisfies the necessary requirements, so that the answer in this case is necessarily "yes." If $S$ is nonempty, then no string longer than the shortest string in $S$ can be a subsequence of every string in $S$, so we need only guess a string $w$ whose length is bounded by that of the shortest string in $S$ and check whether $w$ is a subsequence of every string in $S$ and of no string in $T$, which takes time proportional to the sum of the lengths of all the input strings (Lemma 2).

To see that this problem is complete in NP, we reduce satisfiability of CNF formulas to this question. Given a CNF formula $\phi$ over the $n$ variables $x_i$ for $1 \leq i \leq n$, we construct two sets of binary strings $S$ and $T$ such that $\phi$ is satisfiable if and only if there exists a shuffle string $u$ that is a subsequence of every string in $S$ and of no string in $T$. The set $S$ is just the two strings $s_0$ and $s_1$ in the set $S_n$. The set $T$ is the strings in the set $T_n$ together with additional strings determined by the clauses of $\phi$. By Lemma 8, the strings consistent with $S_n$ and $T_n$ are the $2^n$ strings in $A_n$.

We use each $u = u_1 d u_2 d \cdots u_n d$ in $A_n$ to represent an assignment to the $n$ variables $x_i$ by choosing $x_i = 0$ if $u_i$ is 00000 and $x_i = 1$ if $u_i = 00100$. We construct additional elements of $T$ based on the clauses of the formula $\phi$ to exclude any strings representing assignments that do not satisfy $\phi$. For example, if clause $j$ of $\phi$ is

$$(x_3 \vee \overline{x}_6 \vee \overline{x}_{17}),$$

we add a string $t_j$ to $T$ obtained from $s_0$ by replacing occurrence 3 of $v_0$ by 00000, replacing occurrence 6 of $v_0$ by 00100, and occurrence 17 of $v_0$ by 00100, where we have chosen 00000 or 00100 to falsify the corresponding literal. The strings in $A_n$ that are subsequences of $t_j$ are exactly those that correspond to assignments that falsify clause $j$ of $\phi$, and adding $t_j$ to $T$ eliminates these strings from those consistent with $S$ and $T$. By adding one string $t_j$ to $T$ for each clause $j$ of $\phi$, we ensure that the only strings $u$ that are subsequences of both elements of $S$ and not subsequences of any element of $T$ are exactly those elements of $A_n$ that correspond to assignments that do not falsify any clause of $\phi$. Thus, there exists at least one string $u$ that is a subsequence of both strings in $S$ and not a subsequence of any string in $T$ if and only if $\phi$ is satisfiable.

Note that $S$ contains two strings of length $O(n)$, $T_n$ contains $4n$ strings of length $O(n)$, and $T$ additionally contains one string of length $O(n)$ for each clause of $\phi$, so the sizes of $S$ and $T$ are polynomial in the size of $\phi$. This completes the proof of Theorem 7. ■

## 5. Cryptographic Limitations on PAC Learning Shuffle Ideals

In this section we show that the problem of PAC learning any class of constant-depth, polynomial-size threshold formulas is efficiently reducible to the problem of PAC learning shuffle ideals. Because for some constant depth, the class of polynomial-size threshold formulas of that depth are capable of computing iterated product, the results of Kearns and Valiant (1994) imply that a polynomial time PAC algorithm to learn them would imply polynomial time algorithms for certain fundamental problems in cryptography, namely, inverting RSA encryption, factoring Blum integers, and testing quadratic residuosity. Thus, the class of shuffle ideals faces the same cryptographic limitations on PAC learnability as demonstrated by Kearns and Valiant for the class of general regular languages represented by deterministic finite automata.

A *threshold function* is a Boolean function with $m$ inputs and a threshold $t$. Its output is 1 if at least $t$ of its inputs are 1 and 0 otherwise. Thus, an OR of $m$ inputs is equivalent to a threshold function with threshold 1, and an AND of $m$ inputs is equivalent to a threshold function with threshold $m$. There are $m+2$ different threshold functions of $m$ inputs, corresponding to $t = 0, 1, \ldots, m+1$. The threshold $t = 0$ computes the constant function 1, while the threshold $t = m+1$ computes the constant function 0.

Given an integer $m > 1$, we define the class $T(n,m,d)$ of threshold formulas over the variables $V_n = \{x_1, x_2, \ldots, x_n\}$ of fan-in exactly $m$ and depth $d$ by induction on $d$ as follows. The formulas of depth $d = 0$ are the two constants 0 and 1 and the $2n$ literals $x_i$ and $\overline{x}_i$. For $d > 0$, the formulas of depth $d$ consist of a threshold function with $m$ inputs applied to a sequence of $m$ formulas of depth $d-1$. Note that a threshold function of $m$ inputs can be used to compute a threshold function of fewer inputs by insuring that the excess inputs are the constant function 0.

We can picture the elements of $T(n,m,d)$ as ordered full $m$-ary trees of depth $d$ whose internal nodes are labeled by threshold functions, and whose leaves are labeled by constants or literals. Thus, the total number of occurrences of constants or literals in a threshold formula of fan-in $m$ and depth $d$ is $O(m^d)$. If $d$ is a fixed constant and $m$ is bounded by a polynomial in $n$, the total size of such a formula is bounded by a polynomial in $n$. The same is true if $m$ is a fixed constant and $d$ is bounded by $O(\log n)$; in this case, the formulas compute functions in the class NC1 of constant fan-in, logarithmic depth Boolean circuits.

We now describe a reduction parameterized by $d$ that maps each threshold formula $f$ in $T(n,m,d)$ to a shuffle string $r_d(f)$, and each assignment $a$ to the variables $V_n$ to an assignment string $s_d(a)$, such that the assignment $a$ satisfies $f$ if and only if the shuffle string $r_d(f)$ is a subsequence of the assignment string $s_d(a)$. The string alphabet consists of the symbols 0 and 1 and a set of $d+1$ delimiters: $\#_0, \#_1, \ldots, \#_d$.

The base case is $d = 0$, where $f$ is a single constant 0 or 1 or a single literal $x_i$ or $\bar{x}_i$. In this case, the shuffle string is

$$r_0(f) = y_1 \#_0 y_2 \#_0 \ldots y_n \#_0,$$

where $y_j$ is defined as follows. If $f = 0$ then $y_j = 01$ for all $j$, and if $f = 1$ then $y_j = \lambda$ for all $j$. If $f = x_i$ then $y_j = \lambda$ for all $j \neq i$ and $y_i = 1$, while if $f = \bar{x}_i$ then $y_j = \lambda$ for all $j \neq i$ and $y_i = 0$.

If the assignment $a$ is given by a binary string $a_1 a_2 \ldots a_n$, indicating that $x_i$ is assigned the value $a_i$, then the string representing the assignment is just

$$s_0(a) = a_1 \#_0 a_2 \#_0 \ldots a_n \#_0.$$

It is clear that $r_0(f)$ is a subsequence of $s_0(a)$ if and only if the $n$ occurrences of $\#_0$ in each string are matched, and $y_j$ is a subsequence of $a_j$ for all $j = 1, 2, \ldots, n$. For $f = 0$ we have $y_j = 01$ for all $j$, so this holds for no $a$. For $f = 1$ we have $y_j = \lambda$ for all $j$, and this holds for every $a$. If $f$ is a literal, then this holds if and only if $y_i = a_i$, that is, if and only if $a$ satisfies $f$. Thus, when $f$ is a constant or a literal, $r_0(f)$ is a subsequence of $s_0(a)$ if and only if $a$ satisfies $f$.

In addition to defining the shuffle string and the assignment strings at each level, we also define a slack string. For level 0, the slack string $z_0$ is defined as follows.

$$z_0 = (01\#_0)^n,$$

That is, $z_0$ consists of $n$ repetitions of the string $01\#_0$. For level $d$, the slack string is designed to ensure that $r_d(f)$ is a subsequence of $z_d$ for any $f \in T(n,m,d)$; this clearly holds at level $d = 0$.

For the inductive case $d > 0$, we assume that the construction has been defined for $d-1$ using symbols 0, 1, and delimiters $\#_0, \ldots, \#_{d-1}$. Thus the level $d$ delimiter, $\#_d$, has not yet been used. Suppose $f$ is a depth $d$ threshold formula from $T(n,m,d)$, that is,

$$f = \theta(f_1, f_2, \ldots, f_m),$$

where each $f_i$ is a depth $d-1$ threshold formula and $\theta$ is a threshold function with threshold $t$. We define the shuffle string

$$r_d(f) = u_1 u_1 u_2 u_2 \cdots u_m u_m (\#_d)^{2t},$$

where for each $i = 1, 2, \ldots, m$,

$$u_i = r_{d-1}(f_i) \#_d.$$

That is, $r_d(f)$ consists of two copies of the level $d-1$ code for $f_i$, with each copy followed by the delimiter $\#_d$, for $i = 1, 2, \ldots, m$, followed by $t$ pairs of the delimiter $\#_d$. Note that $r_d(f)$ may contain up to $4m+2$ copies of $\#_d$.

Given an assignment $a$ to the variables $V_n$, we define a level $d$ assignment string

$$s_d(a) = v^{2m},$$

where

$$v = s_{d-1}(a) \#_d z_{d-1} \#_d.$$

That is, $s_d(a)$ is $2m$ copies of the string $v$ consisting of the level $d-1$ code for $a$, followed by $\#_d$, followed by the level $d-1$ slack string, followed by $\#_d$. Note that $s_d(a)$ contains exactly $4m$ copies of $\#_d$.

Finally, the level $d$ slack string is defined as follows.

$$z_d = (z_{d-1}\#_d)^{4m+2}.$$

A straightforward induction shows that for any threshold formula $f$ in $T(n,m,d)$, $r_d(f)$ is a subsequence of $z_d$, and for any assignment $a$ to the variables, $s_d(a)$ is also a subsequence of $z_d$.

**Lemma 9** *For all threshold formulas $f$ in $T(n,m,d)$ and assignments $a$ to the variables in $V_n$, $a$ satisfies $f$ if and only if $r_d(f)$ is a subsequence of $s_d(a)$.*

**Proof** This is proved by induction on $d$. For $d = 0$, the basis construction showed that for all constants or literals $f$ and assignments $a$, $a$ satisfies $f$ if and only if $r_0(f)$ is a subsequence of $s_0(a)$.

Inductively assume that the construction works for $d-1$. Suppose $f$ is a depth $d$ threshold formula, that is,

$$f = \theta(f_1, f_2, \ldots, f_m),$$

where each $f_i$ is a depth $d-1$ threshold formula and $\theta$ is a threshold function with threshold $t$. For any index $i$ and any assignment $a$ let

$$u_i = r_{d-1}(f_i)\#_d$$

and

$$v = s_{d-1}(a)\#_d z_{d-1}\#_d.$$

Because $r_{d-1}(f_i)$ is a subsequence of the slack string $z_{d-1}$, $u_iu_i$ is a subsequence of $vv$. Also, $u_iu_i$ is a subsequence of $v$ if and only if $r_{d-1}(f_i)$ is a subsequence of $s_{d-1}(a)$, which holds if and only if $a$ satisfies $f_i$, by the inductive assumption. If $u_iu_i$ is not a subsequence of $v$, then a leftmost embedding of $u_iu_i$ in $vv$ must match the first $\#_d$ in $u_iu_i$ to the second $\#_d$ in $vv$ and the second $\#_d$ in $u_iu_i$ to the fourth $\#_d$ in $vv$, thereby "consuming" all of $vv$ for the embedding.

Suppose $a$ satisfies $f$. Because $\theta$ is a threshold function with threshold $t$, there must be a set $T$ of at least $t$ indices $i$ such that $a$ satisfies $f_i$. By the inductive assumption, this means that $r_{d-1}(f_i)$ is a subsequence of $s_{d-1}(a)$ for each $i \in T$. For each $i \in T$, $u_iu_i$ is a subsequence of $v$. For $i \notin T$, $u_iu_i$ is a subsequence of $vv$ but not of $v$. Thus we can find a leftmost embedding of $r_d(f)$ in $s_d(a)$ by consuming one copy of $v$ from $s_d(a)$ for each $i \in T$ and two copies for each $i \notin T$, using at most $2m-t$ copies, and leaving at least $t$ copies, which allows us to embed the trailing sequence of $2t$ delimiters $\#_d$ in the remaining copies of $v$. Thus $r_d(f)$ is a subsequence of $s_d(a)$.

Conversely, suppose that $r_d(f)$ is a subsequence of $s_d(a)$, and consider a leftmost embedding. Considering the segments $u_iu_i$ of $r_d(f)$ from left to right, we see that the leftmost embedding consumes one copy of $v$ if $a$ satisfies $f_i$ and two copies if $a$ does not satisfy $f_i$. Thus, if $T$ is the set of indices $i$ such that $a$ satisfies $f_i$, then after embedding all $m$ such segments, $2m-|T|$ copies of $v$ are consumed from $s_d(a)$, leaving $|T|$ copies. Because the trailing $2t$ occurrences of $\#_d$ in $r_d(f)$ are matched in the remaining portion of $s_d(a)$, we must have $2|T| \geq 2t$, and therefore $a$ satisfies $f_i$ for at least $t$ indices $i$, that is, $a$ satisfies $f$. ■

How long are the strings $r_d(f)$ and $s_d(a)$? Each is a subsequence of $z_d$, and for $m \geq 2$, the length of $z_d$ is bounded by $(10m)^d(3n)$. This is polynomial in $n$ if either $d$ is a fixed constant and $m$

is polynomial in $n$, or if $m$ is a fixed constant and $d = O(\log n)$. In either case, the mapping from $a$ to $s_d(a)$ is computable in polynomial time, and we have the following results.

The first result assumes a polynomial time algorithm to learn shuffle ideals over some fixed alphabet.

**Theorem 10** *Suppose for some positive integer $d$, there exists a polynomial time algorithm to PAC learn shuffle ideals over an alphabet of size $d+2$. Then for any polynomial $p(n)$, there exists a polynomial time algorithm to PAC learn the threshold formulas in $T(n, p(n), d)$.*

The second result assumes a polynomial time algorithm to learn shuffle ideals over an arbitrary finite alphabet, where the dependence on the alphabet size must be at most exponential.

**Theorem 11** *Suppose there exists an algorithm to PAC learn shuffle ideals over arbitrary finite alphabets that runs in time polynomial in $n$ and $C^s$, where $n$ is a bound on the length of examples, $s$ is the alphabet size and $C$ is a fixed constant. Then for any constant $K$, there exists a polynomial time algorithm to PAC learn the threshold formulas in $T(n, 2, K\log n)$.*

### 5.1 Example of the Construction of $r_d(f)$ and $s_d(a)$

We illustrate the construction for the formula

$$f = (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \wedge x_3)$$

from $T(3,2,2)$ and the assignment $a = 001$. To avoid subscripted delimiters, let #, \$, and % stand for $\#_0$, $\#_1$ and $\#_2$ respectively. For the base case we have the following.

$$\begin{aligned} r_0(x_1) &= 1\#\#\#, \\ r_0(\bar{x}_1) &= 0\#\#\#, \\ r_0(\bar{x}_2) &= \#0\#\#, \\ r_0(x_3) &= \#\#1\#, \\ z_0 &= 01\#01\#01\#. \end{aligned}$$

The two subformulas of $f$ have thresholds of 1 and 2 respectively.

$$\begin{aligned} r_1(x_1 \vee \bar{x}_2) &= 1\#\#\#\$1\#\#\#\$\#0\#\#\$\#0\#\#\$\$\$, \\ r_1(\bar{x}_1 \wedge x_3) &= 0\#\#\#\$0\#\#\#\$\#\#1\#\$\#\#1\#\$\$\$\$\$, \\ z_1 &= (01\#01\#01\#\$)^{10}. \end{aligned}$$

For $f$ the threshold is 2.

$$\begin{aligned} r_2(f) &= ((1\#\#\#\$)^2(\#0\#\#\$)^2)\$\$\%)^2((0\#\#\#\$)^2(\#\#1\#\$)^2\$\$\$\$\%)^2\%\%\%\%, \\ z_2 &= ((01\#01\#01\#\$)^{10}\%)^{10}. \end{aligned}$$

The assignment strings for the assignment $a = 001$ are as follows.

$$\begin{aligned} s_0(a) &= 0\#0\#1\#, \\ s_1(a) &= (0\#0\#1\#\$01\#01\#01\#\$)^4, \\ s_2(a) &= ((0\#0\#1\#\$01\#01\#01\#\$)^4\%(01\#01\#01\#\$)^{10}\%)^4. \end{aligned}$$

Assignment $a$ satisfies $f$ and $r_2(f)$ is a subsequence of $s_2(a)$.

## 6. Discussion

We have shown that the class of shuffle ideals is not efficiently properly PAC learnable if RP $\neq$ NP, and is not efficiently improperly PAC learnable under certain cryptographic assumptions. On the other hand, even with classification noise, efficient proper PAC learning of shuffle ideals is possible under the uniform distribution. One technical question that remains is whether the results in Section 5 can be proved for an alphabet of constant size (independent of $d$.) Another is whether PAC learning shuffle ideals is as hard as PAC learning deterministic finite acceptors. Much remains to be understood about the learnability of subclasses of the regular languages.

## Acknowledgments

A preliminary version of this paper appears in the ALT proceedings (Angluin et al., 2012). We thank the anonymous reviewers of ALT 2012 and JMLR for their helpful comments, and Dongqu Chen of Yale University for pointing out an error in the proof of Lemma 3. Improvements in the current version of the paper include a corrected proof of Lemma 3, reducing the minimum alphabet size for Theorem 7 from 3 to 2, and all of the results in Section 5.

The research of Dana Angluin and James Aspnes was supported by the National Science Foundation under grant CCF-0916389, that of Sarah Eisenstat by the T-Party Project (a joint research program between MIT and Quanta Computer Inc., Taiwan) and that of Aryeh Kontorovich by the Israel Science Foundation under grant No. 1141/12.

## References

Dana Angluin. On the complexity of minimum inference of regular sets. *Information and Control*, 39(3):337–350, 1978. ISSN 0019-9958. doi: 10.1016/S0019-9958(78)90683-6.

Dana Angluin. Inference of reversible languages. *Journal of the ACM*, 29(3):741–765, July 1982. ISSN 0004-5411. doi: 10.1145/322326.322334.

Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, November 1987. ISSN 0890-5401. doi: 10.1016/0890-5401(87)90052-6.

Dana Angluin, James Aspnes, and Aryeh Kontorovich. On the learnability of shuffle ideals. In *Proceedings of the 23rd International Conference on Algorithmic Learning Theory*, ALT '12, pages 111–123, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-34105-2. doi: 10.1007/978-3-642-34106-9_12.

Nader H. Bshouty. Exact learning of formulas in parallel. *Machine Learning*, 26(1):25–41, January 1997. ISSN 0885-6125. doi: 10.1023/A:1007320031970.

Nader H. Bshouty, Jeffrey C. Jackson, and Christino Tamon. Exploring learnability between exact and PAC. *Journal of Computer and System Sciences*, 70(4):471–484, June 2005. ISSN 0022-0000. doi: 10.1016/j.jcss.2004.10.002.

Alexander Clark and Franck Thollard. PAC-learnability of probabilistic deterministic finite state automata. *Journal of Machine Learning Research*, 5:473–497, December 2004. ISSN 1532-4435.

Corinna Cortes, Leonid (Aryeh) Kontorovich, and Mehryar Mohri. Learning languages with rational kernels. In *Proceedings of the 20th Annual Conference on Learning Theory*, COLT '07, pages 349–364, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-72925-9.

Colin de la Higuera. A bibliographical study of grammatical inference. *Pattern Recognition*, 38(9): 1332–1348, September 2005. ISSN 0031-3203. doi: 10.1016/j.patcog.2005.01.003.

Samuel Eilenberg and Saunders Mac Lane. On the groups of $H(\Pi, n)$, I. *Annals of Mathematics. Second Series*, 58:55–106, July 1953. ISSN 0003-486X.

Funda Ergün, S. Ravi Kumar, and Ronitt Rubinfeld. On learning bounded-width branching programs. In *Proceedings of the Eighth Annual Conference on Computational Learning Theory*, COLT '95, pages 361–368, New York, NY, USA, 1995. ACM. ISBN 0-89791-723-5. doi: 10.1145/225298.225342.

E. Mark Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1978.

Yoshiyasu Ishigami and Sei'ichi Tani. VC-dimensions of finite automata and commutative finite automata with $k$ letters and $n$ states. *Discrete Applied Mathematics*, 74(3):229–240, May 1997. ISSN 0166-218X. doi: 10.1016/S0166-218X(96)00050-9.

Michael Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45 (6):983–1006, November 1998. ISSN 0004-5411. doi: 10.1145/293347.293351.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, January 1994. ISSN 0004-5411. doi: 10.1145/174644.174647.

Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, USA, 1994. ISBN 0-262-11193-4.

Ondřej Klíma and Libor Polák. Hierarchies of piecewise testable languages. In *Proceedings of the 12th International Conference on Developments in Language Theory*, DLT '08, pages 479–490, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85779-2. doi: 10.1007/978-3-540-85780-8_38.

Leonid (Aryeh) Kontorovich and Boaz Nadler. Universal kernel-based learning with applications to regular languages. *Journal of Machine Learning Research*, 10:1095–1129, June 2009. ISSN 1532-4435.

Leonid (Aryeh) Kontorovich, Dana Ron, and Yoram Singer. A Markov model for the acquisition of morphological structure. *Technical Report CMU-CS-03-147*, June 2003.

Leonid (Aryeh) Kontorovich, Corinna Cortes, and Mehryar Mohri. Learning linearly separable languages. In *Proceedings of the 17th International Conference on Algorithmic Learning Theory*, ALT '06, pages 288–303, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-46649-5, 978-3-540-46649-9. doi: 10.1007/11894841_24.

Leonid (Aryeh) Kontorovich, Corinna Cortes, and Mehryar Mohri. Kernel methods for learning languages. *Theoretical Computer Science*, 405(3):223–236, October 2008. ISSN 0304-3975. doi: 10.1016/j.tcs.2008.06.037.

Kimmo Koskenniemi. Two-level model for morphological analysis. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI '83, pages 683–685, San Francisco, CA, USA, 1983. Morgan Kaufmann Publishers Inc.

M. Lothaire. *Combinatorics on Words*, volume 17 of *Encyclopedia of Mathematics and Its Applications*. Addison-Wesley, 1983.

Mehryar Mohri. On some applications of finite-state automata theory to natural language processing. *Natural Language Engineering*, 2(1):61–80, March 1996. ISSN 1351-3249. doi: 10.1017/S135132499600126X.

Mehryar Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, June 1997. ISSN 0891-2017.

Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002. ISSN 0885-2308. doi: 10.1006/csla.2001.0184.

Mehryar Mohri, Pedro J. Moreno, and Eugene Weinstein. Efficient and robust music identification with weighted finite-state transducers. *IEEE Transactions on Audio, Speech & Language Processing*, 18(1):197–207, January 2010. ISSN 1063-6676. doi: 10.1109/TASL.2009.2023170.

José Oncina and Pedro García. Identifying regular languages in polynomial time. In *Advances in Structural and Syntactic Pattern Recognition*, volume 5 of *Machine Perception and Artificial Intelligence*, pages 99–108. World Scientific Publishing, 1992.

Nick Palmer and Paul W. Goldberg. PAC-learnability of probabilistic deterministic finite state automata in terms of variation distance. *Theoretical Compututer Science*, 387(1):18–31, November 2007. ISSN 0304-3975. doi: 10.1016/j.tcs.2007.07.023.

Rajesh Parekh and Vasant G. Honavar. Learning DFA from simple examples. *Machine Learning*, 44(1-2):9–35, July 2001. ISSN 0885-6125. doi: 10.1023/A:1010822518073.

Gheorghe Păun, editor. *Mathematical Aspects of Natural and Formal Languages*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1994. ISBN 9-8102-1914-8.

Leonard Pitt and Manfred K. Warmuth. Prediction-preserving reducibility. *Journal of Computer and System Sciences*, 41(3):430–467, December 1990. ISSN 0022-0000. doi: 10.1016/0022-0000(90)90028-J.

Leonard Pitt and Manfred K. Warmuth. The minimum consistent DFA problem cannot be approximated within any polynomial. *Journal of the ACM*, 40(1):95–142, January 1993. ISSN 0004-5411. doi: 10.1145/138027.138042.

Owen Rambow, Srinivas Bangalore, Tahir Butt, Alexis Nasr, and Richard Sproat. Creating a finite-state parser with application semantics. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 2*, COLING '02, pages 1–5, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1071884.1071910.

Dana Ron, Yoram Singer, and Naftali Tishby. On the learnability and usage of acyclic probabilistic finite automata. *Journal of Computer and System Sciences*, 56(2):133–152, 1998. ISSN 0022-0000. doi: 10.1006/jcss.1997.1555.

Imre Simon. Piecewise testable events. In *Proceedings of the 2nd GI Conference on Automata Theory and Formal Languages*, pages 214–222, London, UK, 1975. Springer-Verlag. ISBN 3-540-07407-4.

Richard Sproat, William Gale, Chilin Shih, and Nancy Chang. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, 22(3):377–404, September 1996. ISSN 0891-2017.

Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984. ISSN 0001-0782.

Journal of Machine Learning Research 14 (2013) 1533-1561 Submitted 4/12; Revised 1/13; Published 6/13

# Fast Generalized Subset Scan for Anomalous Pattern Detection

**Edward McFowland III** MCFOWLAND@CMU.EDU
**Skyler Speakman** SPEAKMAN@CMU.EDU
**Daniel B. Neill** NEILL@CS.CMU.EDU
*Event and Pattern Detection Laboratory*
*H.J. Heinz III College*
*Carnegie Mellon University*
*Pittsburgh, PA 15213 USA*

**Editor:** Tony Jebara

## Abstract

We propose Fast Generalized Subset Scan (FGSS), a new method for detecting anomalous patterns in general categorical data sets. We frame the pattern detection problem as a search over subsets of data records and attributes, maximizing a nonparametric scan statistic over all such subsets. We prove that the nonparametric scan statistics possess a novel property that allows for efficient optimization over the exponentially many subsets of the data without an exhaustive search, enabling FGSS to scale to massive and high-dimensional data sets. We evaluate the performance of FGSS in three real-world application domains (customs monitoring, disease surveillance, and network intrusion detection), and demonstrate that FGSS can successfully detect and characterize relevant patterns in each domain. As compared to three other recently proposed detection algorithms, FGSS substantially decreased run time and improved detection power for massive multivariate data sets.

**Keywords:** pattern detection, anomaly detection, knowledge discovery, Bayesian networks, scan statistics

## 1. Introduction

We focus on the task of detecting anomalous patterns in massive, multivariate data sets. The anomalous pattern detection task arises in many domains: customs monitoring, where we attempt to discover patterns of illicit container shipments; disease surveillance, where we must detect emerging outbreaks of disease in the very early stages; network intrusion detection, where we attempt to identify patterns of suspicious network activities; and various others. The underlying assumption of anomalous pattern detection is that the majority of the data is generated according to the same distribution representing the (typically unknown and possibly complex) normal *behavior* of the system, and thus we wish to detect groups of records which are unexpected given the typical data distribution. Most existing anomaly detection methods focus on the discovery of single anomalous data records, for example, detecting a fraudulent transaction in financial data. However, an intelligent fraudster will attempt to disguise their activity so that it closely resembles legitimate transactions. In such a case, each individual fraudulent transaction may only be slightly anomalous, and thus it is only by detecting groups of such transactions that we can discover the pattern of fraud.

Alternatively, customs officials who are tasked with detecting smuggling efforts must decide which of the many containers entering the country daily should be opened for inspection. If a smuggler has discovered an effective method for concealing contraband, he may make similar il-

©2013 Edward McFowland III, Skyler Speakman and Daniel B. Neill.

licit shipments—for example, through the same shipping company, to the same port, and/or with the same declared contents—in the future. By searching for groups of these similar and slightly anomalous shipments, we can detect the presence of the subtle underlying pattern of smuggling. As a concrete example, in §4.1 we analyze a data set of real-world container shipments, with attributes including country of origin, commodity, size, weight, and value. Our approach, described in §2, can identify self-similar subsets of data records (container shipments) for which any subset of these attributes are anomalous, for example, shipments of pineapple from the same country, each with elevated weights as a result of the fruits being hollowed out and filled with drugs. Similarly, in the disease surveillance domain, health officials may wish to ignore a single hospital Emergency Department (ED) having an increased number of patient visits. This could be due to noise or associated with a completely different process that does not reflect an actual disease outbreak. However, health officials are very interested when a group of hospital locations, perhaps within a close proximity to each other, all have an increase in the number of ED visits. As a concrete example, in §4.2 we analyze a data set of real-world Emergency Department visits from Allegheny County, PA, with attributes including hospital id, prodrome, age decile, gender, and zip code. Our approach can identify subsets of data records (ED visits) for which any subset of attributes are anomalous, enabling early and accurate outbreak detection.

### 1.1 The Anomalous Pattern Detection Problem

Here we focus on the problem of anomalous pattern detection in general data, that is, data sets where data records are described by an arbitrary set of attributes. We describe the anomalous pattern detection problem as detecting groups of anomalous records and characterizing their anomalous features, with the intention of understanding the anomalous process that generated these groups. The anomalous pattern detection task begins with the assumption that there are a set of processes generating records in a data set. The "background" process generates records that are typical and expected; these records are assumed to constitute the majority of the data set. Records that do not correspond to the background data pattern, and therefore represent atypical system behavior, are assumed to have been generated by an anomalous process and follow an alternative data pattern. If these anomalies are generated by a process which is very different from the background process, it may be sufficient to evaluate each individual record in isolation because many of the records' attributes will be atypical, or individual attributes may take on extremely surprising values. However, a subtle anomalous process will generate records that may each be only slightly anomalous and therefore extremely challenging to detect. The key insight is to acknowledge and leverage the group structure of these records, since we expect records generated by the same process to have a high degree of similarity. Therefore, we propose to detect self-similar groups of records, for which some subset of attributes are unexpected given the background data distribution.

While searching over groups of records (rather than individual records) may substantially increase detection power, performing this search for general data sets presents several challenges. Many previously proposed pattern detection methods are optimized to detect patterns in data from a specific domain, such as fraud detection or disease surveillance, but cannot be as easily applied to other domains (Chau et al., 2006; Neill and Cooper, 2010; Neill, 2011). Typically these methods can attribute their success to prior knowledge of the behavior of relevant patterns of anomalies. Conversely, general methods for anomalous pattern detection are most useful for identifying interesting and non-obvious patterns occurring in the data, when there is little knowledge of what patterns to

look for. All of the pattern detection methods considered in this work use background data to learn a null model $M_0$, where $M_0$ captures the joint probability distribution over the set of attributes under the null hypothesis $H_0$ that nothing of interest is occurring. A general anomalous pattern detection method must learn $M_0$ while making few assumptions, as it must maintain the ability to detect previously unknown and relevant patterns across diverse data without prior knowledge of the domain or the true distribution from which the data records are drawn.

We can reduce the challenge of anomalous pattern detection in general data to a sequence of tasks: learning a null model, defining the search space (i.e., which subsets of the data will be considered), choosing a function to score the interestingness or anomalousness of a subset, and optimizing this function over the search space in order to find the highest scoring subsets. Therefore, we briefly summarize several previously proposed methods for anomalous pattern detection in general categorical data sets based on their techniques, and their limitations, for addressing these tasks; more detailed descriptions can be found in §3. Each method discussed here, including our proposed Fast Generalized Subset Scan approach, learns the structure and parameters of a Bayesian network from training data to represent $M_0$, and then searches for subsets of records in test data that are collectively anomalous given $M_0$. The training data can be historical data, background data, or simply a separate data set. However, the methods do assume that the training data set does not contain anomalous patterns. Das and Schneider (2007) present a simple solution to the problem of individual record anomaly detection by computing each record's likelihood given $M_0$, and assuming that the lowest-likelihood records are most anomalous; we refer to this approach as the Bayesian Network Anomaly Detector (BN). Although BN is able to locate highly individually anomalous records very quickly, it will lose power to detect anomalous groups produced by a subtle anomalous process where each record, when considered individually, is only slightly anomalous. Furthermore, BN ignores the group structure of anomalies and thus fails to provide specific details (groups of records, or subsets of attributes for which these records are anomalous) useful for understanding the underlying anomalous processes. Anomaly Pattern Detection (APD) first computes each record's likelihood given $M_0$, assuming that the lowest-likelihood records are individually anomalous, and then finds rules (conjunctions of attribute values) with higher than expected numbers of individually anomalous records (Das et al., 2008). APD improves on BN by allowing for subsets larger than one record. However, like BN, APD loses power to detect subtle anomalies because of its dependency on individual record anomalousness. Also, APD permits records within a group to each be anomalous for different reasons, therefore compromising its ability to differentiate between true examples of anomalous patterns and noise, and making it difficult to characterize why a given subset is anomalous. Anomalous Group Detection (AGD) maximizes a likelihood ratio statistic over subsets of records, where a subset's likelihood under $H_0$ is computed from the null model Bayesian network and the subset's likelihood under $H_1$ is computed from a Bayesian network learned specifically from the subset of interest (Neill et al., 2008; Das, 2009). The search spaces of all possible rules and all possible subsets are too vast to search over exhaustively for APD and AGD respectively. Therefore both approaches reduce their search spaces, by using a set of 2-component rules for APD and a greedy heuristic search for AGD. In both cases, the algorithm may fail to identify the most interesting subset of the data. Therefore the current state of the literature requires anomalies to be found either in isolation or through a reduction in the search space, when searching for groups, which could remove the anomalous groups of interest from consideration. In §2 we propose an algorithm, Fast Generalized Subset Scan (FGSS), that can efficiently maximize a scoring function over all possible subsets of data records and attributes, allowing us to find the most anomalous sub-

set. Furthermore, our algorithm does not depend on the anomalousness of an entire record, but only some subset of its attributes, and is therefore able to provide useful information about the underlying anomalous process by identifying the subset of attributes for which a group is anomalous.

## 2. Fast Generalized Subset Scan

Fast Generalized Subset Scan (FGSS) is a novel method for anomalous pattern detection in general categorical data. Unlike previous methods, we frame the pattern detection problem as a search over subsets of data records *and* subsets of attributes; we therefore search for self-similar groups of records for which some subset of attributes is anomalous. More precisely, we define a set of data records $R_1 \dots R_N$ and attributes $A_1 \dots A_M$. Here we assume that all attributes are categorical, but future work will extend the approach to continuous attributes as well. For each record $R_i$, we assume that we have a value $v_{ij}$ for each attribute $A_j$. We then define the subsets $S$ under consideration to be $S = R \times A$, where $R \subseteq \{R_1 \dots R_N\}$ and $A \subseteq \{A_1 \dots A_M\}$. We wish to find the most anomalous subset

$$S^* = R^* \times A^* = \arg\max_S F(S),$$

where the score function $F(S)$ defines the anomalousness of a subset of records and attributes. We accomplish this by first learning a Bayesian network model, $M_0$, from the training data. For each value $v_{ij}$ (the value of attribute $A_j$ for record $R_i$), FGSS then computes its likelihood $l_{ij}$ given $M_0$. This likelihood represents the conditional probability of the observed value $v_{ij}$ under the null hypothesis $H_0$, given its parent attribute values for record $R_i$. Then the method computes an empirical $p$-value range $p_{ij}$ for each $l_{ij}$, which serves as a measure of how uncommon it is to see a likelihood as low as $l_{ij}$ under $H_0$. More specifically, $p_{ij}$ is computed by ranking the likelihoods $l_{ij}$ for a given attribute $A_j$, with the rankings then scaled to [0,1]. Finally, FGSS searches for subsets that contain an unexpectedly large number of low (significant) empirical $p$-value ranges, as such a subset is more likely to have been generated by an anomalous process.

### 2.1 Learning the Data Probability Distribution

The FGSS algorithm first learns a Bayesian network which models the probability distribution of the data under the assumption of the null hypothesis that no anomalous patterns exist. As in the previously proposed BN, APD, and AGD methods, this Bayesian network is typically learned from a separate "clean" data set of training data assumed to contain no anomalous patterns, but can also be learned from the test data if the proportion of anomalies is assumed to be very small. We use the Optimal Reinsertion algorithm proposed by Moore and Wong (2003) to learn the structure of the Bayesian network, using smoothed maximum likelihoods to estimate the parameters of the conditional probability table. Smoothing provides a means to handle sparsity of the training data, as it is possible that combinations of attribute values which appear in the test data will not appear in the training data. In such cases we would like to assume a low, but non-zero, probability for the corresponding entries in the conditional probability table.

If we consider the node corresponding to attribute $A_j$ in our Bayesian network $M_0$ and let $A_{p(j)}$ represent its parent nodes, then we represent the parameters of the conditional probability tables as follows:

$$\theta_{jmk} = P_{M_0}(A_j = m \,|\, A_{p(j)} = k) \quad \forall j, m, k.$$

To estimate $\theta_{jmk}$, let $N_{jmk}$ correspond to the number of instances in the data where $A_j = m$ and $A_{p(j)} = k$. To apply Laplace smoothing, we define the arity of $A_j$ as $C$ and add $1/C$ to each $N_{jmk}$. Therefore, our smoothed parameter estimates are computed as

$$\hat{\theta}_{jmk} = \frac{N_{jmk} + 1/C}{\sum_{m'}(N_{jm'k} + 1/C)},$$

thus assuming that the total weight of the prior sums to one for each attribute $A_j$ and set of parent values $A_{p(j)}$. After learning a model of the data distribution under the null hypothesis, FGSS then computes

$$l_{ij} = P_{M_0}(A_j = v_{ij} \,|\, A_{p(j)} = v_{i,p(j)}),$$

representing the individual attribute-value likelihoods of each attribute for a given record, conditioned on its parent attribute values for that record. We compute these individual attribute-value likelihoods for all records in the training and test data sets.

### 2.2 Computing Empirical $p$-value Ranges

The calculation of empirical $p$-value ranges in the test data set requires obtaining a ranking of the likelihoods $l_{ij}$ for each attribute $A_j$. To do so, we calculate for each likelihood $l_{ij}$ the quantities

$$N_{\text{beat}}(l_{ij}) = \sum_{R_k \in D_{\text{train}}} I(l_{kj} < l_{ij}),$$
$$N_{\text{tie}}(l_{ij}) = \sum_{R_k \in D_{\text{train}}} I(l_{kj} = l_{ij}).$$

We then define the empirical $p$-value range corresponding to likelihood $l_{ij}$ as

$$\begin{aligned} p_{ij} &= [p_{\min}(p_{ij}), p_{\max}(p_{ij})] \\ &= \left[\frac{N_{\text{beat}}(l_{ij})}{N_{\text{train}}+1}, \frac{N_{\text{beat}}(l_{ij}) + N_{\text{tie}}(l_{ij}) + 1}{N_{\text{train}}+1}\right], \end{aligned} \tag{1}$$

where $N_{\text{train}}$ is the total number of training data records.

To properly interpret the concept of an empirical $p$-value range, we first consider the traditional empirical $p$-value

$$\hat{p}(x) = \frac{1}{n}\sum_{z=1}^{n} I(X_z \leq x)$$

for $n$ data samples, which closely resembles $p_{\max}(p_{ij})$, the upper limit of $p_{ij}$. For an attribute $A_j$, corresponding to a column in the test data set, there is some true distribution of likelihoods $l_{ij}$ under $H_0$. Since the training data is assumed to contain no anomalous patterns, we can estimate the true cumulative distribution function $F_{L_j}(l)$ with an empirical cumulative distribution function

$$\hat{F}_{L_j}(l) = \frac{N_{\text{beat}}(l) + N_{\text{tie}}(l)}{N_{\text{train}}}$$

derived from the training data set. Then the empirical $p$-value corresponding to a given likelihood $l_{ij}$ in the test data set can be defined as $\hat{p}(l_{ij}) = \hat{F}_{L_j}(l_{ij})$. If the null hypothesis is true, then the test data set also has no anomalous patterns, and is generated from the same distribution as the training data

set. In this case, the empirical $p$-values $\hat{p}(l_{ij})$ will be asymptotically distributed as Uniform[0,1] for each attribute $A_j$. Davison and Hinkley (1997) note that the smoothed empirical $p$-value

$$\hat{p}(l) = \hat{F}_{L_j}(l) = \frac{N_{\text{beat}}(l) + N_{\text{tie}}(l) + 1}{N_{\text{train}} + 1}$$

is also asymptotically unbiased, and is a more accurate estimator of the true $p$-value. This definition also guards against obtaining an empirical $p$-value of zero, which is consistent with our knowledge that true $p$-values are non-zero.

Our FGSS algorithm extends the concept of empirical $p$-values to empirical $p$-value ranges in order to appropriately handle ties in likelihoods. In the general data set context, we often see many records with identical attribute-value likelihoods, typically as a result of identical attribute values. Using an empirical $p$-value, where tied likelihoods are treated identically to lower likelihood values, will introduce a bias toward larger $p$-values when ties in likelihood are present. However, under the null hypothesis that the training and test data sets are drawn from the same distribution, if we compute the empirical $p$-value ranges as defined in (1) and then draw an empirical $p$-value uniformly at random from each range $[p_{\min}(p_{ij}), p_{\max}(p_{ij})]$, then the resulting empirical $p$-values will be asymptotically distributed as Uniform[0,1]. As a concrete example, if a given attribute was entirely uninformative (i.e., all training and test data records had identical likelihoods for that attribute), we would obtain an empirical $p$-value range of [0,1] for each test record, while the previous empirical $p$-value approach would set each empirical $p$-value equal to 1.

For a single $p$-value, $p$, we can define an indicator variable $n_\alpha(p)$ representing whether or not that $p$-value is significant at level $\alpha$:

$$n_\alpha(p) = I(p \leq \alpha).$$

This traditional definition of significance can be extended naturally to $p$-value ranges by considering the proportion of each range that is significant at level $\alpha$, or equivalently, the probability that a $p$-value drawn uniformly from $[p_{\min}(p_{ij}), p_{\max}(p_{ij})]$ is less than $\alpha$. The quantity $n_\alpha(p_{ij})$ representing the significance of a $p$-value range is therefore defined as:

$$n_\alpha(p_{ij}) = \begin{cases} 1 & \text{if } p_{\max}(p_{ij}) < \alpha \\ 0 & \text{if } p_{\min}(p_{ij}) > \alpha \\ \frac{\alpha - p_{\min}(p_{ij})}{p_{\max}(p_{ij}) - p_{\min}(p_{ij})} & \text{otherwise.} \end{cases}$$

For a subset $S$, we can then define the quantities

$$N_\alpha(S) = \sum_{v_{ij} \in S} n_\alpha(p_{ij}), \tag{2}$$

$$N(S) = \sum_{v_{ij} \in S} 1 \tag{3}$$

where $N(S)$ represents the total number of empirical $p$-value ranges contained in subset $S$. $N_\alpha(S)$ can informally be described as the number of $p$-value ranges in $S$ which are significant at level $\alpha$, but is more precisely the total probability mass less than $\alpha$ in these $p$-value ranges, since it is possible for a range $p_{ij}$ to have $p_{\min}(p_{ij}) \leq \alpha \leq p_{\max}(p_{ij})$. For a subset $S$ consisting of $N(S)$ empirical

$p$-value ranges, we can compute the expected number of significant $p$-value ranges under the null hypothesis $H_0$:

$$\begin{aligned} E\left[N_\alpha(S)\right] &= E\left[\sum_{v_{ij}\in S} n_\alpha(p_{ij})\right] \\ &= \sum_{v_{ij}\in S} E\left[n_\alpha(p_{ij})\right] \\ &= \sum_{v_{ij}\in S} \alpha \\ &= \alpha N(S). \end{aligned}$$

We note that this equation follows from the property that the empirical $p$-values are identically distributed as Uniform[0,1] under the null hypothesis, and holds regardless of whether the $p$-values are independent. Under the alternative hypothesis, we expect the likelihoods $l_{ij}$ (and therefore the corresponding $p$-value ranges $p_{ij}$) to be lower for the affected subset of records and attributes, resulting in a higher value of $N_\alpha(S)$ for some $\alpha$. Therefore a subset $S$ where $N_\alpha(S) > \alpha N(S)$ (i.e., a subset with a higher than expected number of low, significant $p$-value ranges) is potentially affected by an anomalous process.

### 2.3 Nonparametric Scan Statistic

To determine which subsets of the data are most anomalous, FGSS uses a nonparametric scan statistic (Neill and Lingwall, 2007) to compare the observed and expected number of significantly low $p$-values contained in subset $S$. We define the general form of the nonparametric scan statistic as

$$F(S) = \max_\alpha F_\alpha(S) = \max_\alpha \phi(\alpha, N_\alpha(S), N(S)) \tag{4}$$

where $N_\alpha(S)$ and $N(S)$ are defined as in (2) and (3) respectively. We assume that the function $\phi(\alpha, N_\alpha, N)$ satisfies several intuitive properties that will also allow efficient optimization:

*(A1)* $\phi$ is monotonically ***increasing*** w.r.t. $N_\alpha$.
*(A2)* $\phi$ is monotonically ***decreasing*** w.r.t. $N$ and $\alpha$.
*(A3)* $\phi$ is ***convex***.

These assumptions follow naturally because the ratio of observed to expected number of significant $p$-values $\frac{N_\alpha}{N\alpha}$ increases with the numerator (A1), and decreases with the denominator (A2). Also, a fixed ratio of observed to expected should be more significant when the observed and expected counts are large (A3).

We consider "significance levels" $\alpha$ between 0 and some constant $\alpha_{\max} < 1$. If there is a prior expectation of the subtleness of the anomalous process, $\alpha_{\max}$ can be chosen appropriately. The less subtle the anomalous process, that is, the more individually anomalous the records it generates are expected to be, the lower $\alpha_{\max}$ can be set. We note that maximizing of $F(S)$ over a range of $\alpha$ values, rather than for a single arbitrarily-chosen value of $\alpha$, enables the nonparametric scan statistic to detect a small number of highly anomalous $p$-values, a larger number of subtly anomalous $p$-values, or anything in between.

In this work we explore the use of two functions $\phi(\alpha, N_\alpha, N)$ which satisfy the monotonicity and convexity properties (A1)-(A3) assumed above: the Higher Criticism (HC) statistic (Donoho and

Jin, 2004) and the Berk-Jones (BJ) statistic (Berk and Jones, 1979). The HC statistic is defined as follows:

$$\phi_{\text{HC}}(\alpha, N_\alpha, N) = \frac{N_\alpha - N\alpha}{\sqrt{N\alpha(1-\alpha)}}. \tag{5}$$

Under the null hypothesis of uniformly distributed $p$-value ranges, and the additional simplifying assumption of independence between $p$-value ranges, the number of empirical $p$-value ranges less than $\alpha$ is binomially distributed with parameters $N$ and $\alpha$. Therefore the expected number of $p$-value ranges less than $\alpha$ under $H_0$ is $N\alpha$, with a standard deviation of $\sqrt{N\alpha(1-\alpha)}$. This implies that the HC statistic can be interpreted as the test statistic of a Wald test for the number of significant $p$-value ranges. We note that the assumption of independent $p$-value ranges is not necessarily true in practice, since our method of generating these $p$-value ranges may introduce dependence between the $p$-values for a given record; nevertheless, this assumption results in a simple and efficiently computable score function.

The BJ statistic is defined as:

$$\phi_{\text{BJ}}(\alpha, N_\alpha, N) = NK\left(\frac{N_\alpha}{N}, \alpha\right), \tag{6}$$

where $K$ is the Kullback-Liebler divergence,

$$K(x,y) = x\log\frac{x}{y} + (1-x)\log\frac{1-x}{1-y},$$

between the observed and expected proportions of $p$-values less than $\alpha$. The BJ statistic can be described as the log-likelihood ratio statistic for testing whether the empirical $p$-values are uniformly distributed on [0,1], where the alternative hypothesis assumes a piecewise constant distribution with probability density function

$$f(x) = \begin{cases} f_1 & \text{for } 0 \le x \le \alpha \\ f_2 & \text{for } \alpha \le x \le 1 \end{cases}$$
$$\text{with } f_1 > f_2.$$

Berk and Jones (1979) demonstrated that this test statistic fulfills several optimality properties and has greater power than any weighted Kolmogorov statistic.

We note that the original version of the nonparametric scan statistic, used for spatial data by Neill and Lingwall (2007), considered the HC statistic (5) only, and used empirical $p$-values rather than $p$-value ranges. Our empirical results below demonstrate that the BJ statistic (6) outperforms HC for some real-world anomalous pattern detection tasks, and our use of empirical $p$-value ranges guarantees unbiased scores even when ties in likelihood are present. Furthermore, we present a novel approach for efficient optimization of any nonparametric scan statistic (satisfying the monotonicity and convexity properties (A1)-(A3) assumed above) over subsets of records and attributes, as described below.

### 2.3.1 EFFICIENT NONPARAMETRIC SUBSET SCANNING

Although the nonparametric scan statistic provides a function $F(S)$ to evaluate the anomalousness of subsets in the test data, naively maximizing $F(S)$ over all possible subsets of records and attributes

would be infeasible for even moderately sized data sets, with a computational complexity of $O(2^N \times 2^M)$. However, Neill (2012) defined the linear-time subset scanning (LTSS) property, which allows for efficient and exact maximization of any function satisfying LTSS over all subsets of the data. For a pair of functions $F(S)$ and $G(R_i)$, which represent the "score" of a given subset $S$ and the "priority" of data record $R_i$ respectively, the LTSS property guarantees that the only subsets with the potential to be optimal are those consisting of the top-$k$ highest priority records $\{R_{(1)} \dots R_{(k)}\}$, for some $k$ between 1 and $N$. This property enables us to search only $N$ of the $2^N$ subsets of records, while still guaranteeing that the highest-scoring subset will be found. We demonstrate that the nonparametric scan statistics satisfy the necessary conditions for the linear-time subset scanning property to hold, allowing efficient maximization over subsets of data records (for a given subset of attributes) or over subsets of attributes (for a given subset of records). In the following section, we will show how these efficient optimization steps can be combined to enable efficient joint maximization over all subsets of records and attributes. We begin by restating a theorem from Neill (2012):

**Theorem 1 (Neill, 2012)** *Let $F(S) = F(X, |S|)$ be a function of one additive sufficient statistic of subset $S$, $X(S) = \sum_{R_i \in S} x_i$ (where $x_i$ depends only on record $R_i$), and the cardinality of $S$. Assume that $F(S)$ is monotonically increasing with $X$. Then $F(S)$ satisfies the LTSS property with priority function $G(R_i) = x_i$.*

**Corollary 2** *We consider the general class of nonparametric scan statistics as defined in* (4)*, where the significance level $\alpha$ is allowed to vary from zero to some constant $\alpha_{max}$. For a given value of $\alpha$, and assuming a given subset of attributes $A \subseteq \{A_1 \dots A_M\}$ under consideration, we demonstrate that $F_\alpha(S)$ can be efficiently maximized over all subsets $S = R \times A$, for $R \subseteq \{R_1 \dots R_N\}$. First, we know that every record $R_i$ has the same number $|A|$ of p-value ranges, and thus $N(S) \propto |R|$. Hence we can write*

$$F_\alpha(S) = \phi(\alpha, N_\alpha(R), |R|),$$

*where the additive sufficient statistic $N_\alpha(R)$ is defined as follows:*

$$N_\alpha(R) = \sum_{R_i \in R} \sum_{A_j \in A} n_\alpha(p_{ij}).$$

*Since the nonparametric scan statistic is defined to be monotonically increasing with $N_\alpha$ (A1), we know that $F_\alpha(S)$ satisfies the LTSS property with priority function*

$$G_\alpha(R_i) = \sum_{A_j \in A} n_\alpha(p_{ij}). \tag{7}$$

Therefore the LTSS property holds for each value of $\alpha$, enabling each $F_\alpha(S)$ to be efficiently maximized over subsets of records. Nearly identical reasoning can be used to demonstrate that $F_\alpha(S)$ can be efficiently maximized over all subsets $S = R \times A$, for $A \subseteq \{A_1 \dots A_M\}$, assuming a given set of records $R \subseteq \{R_1 \dots R_N\}$. In this case,

$$F_\alpha(S) = \phi(\alpha, N_\alpha(A), |A|)$$

satisfies the LTSS property with priority function

$$G_\alpha(A_j) = \sum_{R_i \in R} n_\alpha(p_{ij}). \tag{8}$$

Thus the LTSS property enables efficient computation of $\max_S F_\alpha(S)$ for a given value of $\alpha$, but we must still consider how to maximize this function over all values of $\alpha$, for $0 < \alpha \leq \alpha_{\max}$. We demonstrate that only a small set of $\alpha$ levels must be examined, and therefore

$$\max_S F(S) = \max_\alpha \max_S F_\alpha(S)$$

can also be computed efficiently. More precisely, we demonstrate that only the maximum value $p_{\max}(p_{ij})$ of each $p$-value range $p_{ij}$ in the subset $S$ must be considered as a possible value of $\alpha$. We first define some preliminaries:

**Definition 3** *Let $U(S,\alpha_{max})$ be the set of distinct values $\{p_{max}(p_{ij}) : v_{ij} \in S,\ p_{max}(p_{ij}) \leq \alpha_{max}\} \cup \{0, \alpha_{max}\}$.*

**Definition 4** *Let $\alpha_{(k)}$ be the $k^{th}$ smallest value in $U(S,\alpha_{max})$. We will consider the set of intervals $[\alpha_{(k)}, \alpha_{(k+1)}]$, for $k = 1 \ldots |U(S,\alpha_{max})| - 1$.*

**Definition 5** *Let $P(S,\alpha) = \{p_{ij} : v_{ij} \in S, \alpha \in p_{ij}\}$, be the set of p-value ranges $p_{ij}$ in $S$ such that $p_{min}(p_{ij}) \leq \alpha \leq p_{max}(p_{ij})$.*

**Lemma 6** *$N_\alpha(S)$ is a convex function of $\alpha$ over each interval $[\alpha_{(k)}, \alpha_{(k+1)}]$, for $k = 1 \ldots |U(S,\alpha_{max})| - 1$.*

**Proof of Lemma 6** We consider two cases, one of which will hold for any given interval $[\alpha_{(k)}, \alpha_{(k+1)}]$. In either case, we note that no $p_{\max}$ values are contained within the interval, that is, for all values $v_{ij} \in S$ and the corresponding $p$-value ranges $p_{ij}$, $p_{\max}(p_{ij}) \notin (\alpha_{(k)}, \alpha_{(k+1)})$. We begin by observing that

$$\frac{\partial N_\alpha(S)}{\partial \alpha} = \sum_{p_{ij} \in P(S,\alpha)} \frac{1}{p_{\max}(p_{ij}) - p_{\min}(p_{ij})}. \tag{9}$$

Case 1: For all values $v_{ij} \in S$ and the corresponding $p$-value ranges $p_{ij}$, $p_{\min}(p_{ij}) \notin (\alpha_{(k)}, \alpha_{(k+1)})$. In this case, no $p$-value range begins or ends within the interval, and thus $P(S,\alpha)$ is constant over the entire interval $(\alpha_{(k)}, \alpha_{(k+1)})$. Therefore, we know that (9) equals a positive constant over the entire interval, and thus $N_\alpha(S)$ is a linear (and therefore convex) function of $\alpha$.
Case 2: For some value(s) $v_{ij} \in S$ and the corresponding $p$-value ranges $p_{ij}$, $p_{\min}(p_{ij}) \in (\alpha_{(k)}, \alpha_{(k+1)})$. As for Case 1, we know that (9) is piecewise constant, and therefore $N_\alpha(S)$ is a piecewise linear function of $\alpha$. We note additionally that, for a given $p_{\min}(p_{ij})$, the slope described in (9) **increases** by $\frac{1}{p_{\max}(p_{ij}) - p_{\min}(p_{ij})} > 0$ at $\alpha = p_{\min}(p_{ij})$.

From these two cases, we conclude that, as a function of $\alpha$, $N_\alpha(S)$ is piecewise linear with an increasing slope at each value $p_{\min}(p_{ij})$, a decreasing slope at each value $p_{\max}(p_{ij})$, and an otherwise constant slope. Therefore, within each interval defined by $[\alpha_{(k)}, \alpha_{(k+1)}]$ and thus containing no values $p_{\max}(p_{ij})$, we know that $N_\alpha(S)$ is convex. ■

**Theorem 7** *If $F_\alpha(S) = \phi(\alpha, N_\alpha(S), N(S))$ satisfies assumptions (A1)-(A3) given in §2.3, then*

$$\max_\alpha F_\alpha(S) = \max_{\alpha \in U(S,\alpha_{max})} F_\alpha(S).$$

**Proof of Theorem 7** For any point $\alpha_{\text{int}}$ in the open interval $(\alpha_{(k)}, \alpha_{(k+1)})$, we show that:

$$F_{\alpha_{\text{int}}}(S) \leq \max\{F_{\alpha_{(k)}}(S), F_{\alpha_{(k+1)}}(S)\}.$$

To see this, we can write

$$\alpha_{\text{int}} = \lambda\alpha_{(k)} + (1-\lambda)\alpha_{(k+1)}$$
$$\text{where } 0 < \lambda < 1.$$

Then the proof proceeds as follows:

$$\begin{aligned} F_{\alpha_{\text{int}}}(S) &= \phi(\alpha_{\text{int}}, N_{\alpha_{\text{int}}}(S), N(S)) \\ &\leq \phi(\alpha_{\text{int}}, \lambda N_{\alpha_{(k)}}(S) + (1-\lambda)N_{\alpha_{(k+1)}}(S), N(S)) \\ &\leq \max\{F_{\alpha_{(k)}}(S), F_{\alpha_{(k+1)}}(S)\}. \end{aligned}$$

The first inequality follows from the assumption (A1) that $\phi(\alpha, N_\alpha, N)$ is monotonically increasing with $N_\alpha$: by Lemma 6, we know that

$$N_{\alpha_{\text{int}}}(S) \leq \lambda N_{\alpha_{(k)}}(S) + (1-\lambda)N_{\alpha_{(k+1)}}(S).$$

The second inequality follows from the assumption (A3) that $\phi(\alpha, N_\alpha, N)$ is convex. ■

We can conclude that, when computing $\max_\alpha F_\alpha(S)$, only values of $\alpha \in U(S, \alpha_{\max})$ (Definition 3) must be considered, as there will not be any local maxima of the function outside of this set. This fact, combined with Theorem 1, demonstrates that

$$\begin{aligned} \max_S F(S) &= \max_\alpha \max_S F_\alpha(S) \\ &= \max_{\alpha \in U(S,\alpha_{\max})} \max_S F_\alpha(S) \end{aligned} \tag{10}$$

can be efficiently and exactly computed over all subsets $S = R \times A$, where $R \subseteq \{R_1 \ldots R_N\}$, for a given subset of attributes $A$. To do so, we consider the set of distinct $\alpha$ values $U = U(\{R_1 \ldots R_N\} \times A, \alpha_{\max})$. For each $\alpha \in U$, we employ the same logic as described in Corollary 2 to optimize $F_\alpha(S)$: compute the priority $G_\alpha(R_i)$ for each record as in (7), sort the records from highest to lowest priority, and evaluate subsets $S = \{R_{(1)} \ldots R_{(k)}\} \times A$ consisting of the top-$k$ highest priority records, for $k = 1 \ldots N$. For each of the $|U|$ values of $\alpha$ under consideration, the aggregation step requires $O(N|A|) = O(NM)$ time, sorting the records by priority requires $O(N \log N)$ time, and evaluation of the $N$ subsets requires $O(N)$ time, giving a total complexity of $O(|U|N(M + \log N))$ for this optimization step. In the unconstrained case (as opposed to the similarity-constrained FGSS approach described in §2.5), $|U|$ tends to grow linearly with $N$. However, even though we must consider $N|A|$ $p$-values, we note that $|U|$ is upper bounded by the number of distinct likelihood values $l_{ij}$ with corresponding $p_{\max}(p_{ij}) \leq \alpha_{\max}$ in the conditional probability tables of the Bayesian network learned in the first step of the FGSS algorithm, and thus tends to be much smaller than $N|A|$ in practice.

Similarly, (10) can be efficiently and exactly computed over all subsets $S = R \times A$, where $A \subseteq \{A_1 \ldots A_M\}$, for a given subset of records $R$. In this case, we consider the set of distinct $\alpha$ values $U = U(R \times \{A_1 \ldots A_M\}, \alpha_{\max})$. For each $\alpha \in U$, we again employ the same logic as described in Corollary 2 to optimize $F_\alpha(S)$: compute the priority $G_\alpha(A_j)$ for each attribute as in (8),

sort the attributes from highest to lowest priority, and evaluate subsets $S = R \times \{A_{(1)} \ldots A_{(k)}\}$ consisting of the top-$k$ highest priority attributes, for $k = 1 \ldots M$. For each of the $|U|$ values of $\alpha$ under consideration, the aggregation step requires $O(M|R|) = O(MN)$ time, sorting the attributes by priority requires $O(M \log M)$ time, and evaluation of the $M$ subsets requires $O(M)$ time, giving a total complexity of $O(|U|M(N + \log M))$ for this optimization step.

### 2.4 Search Procedure

Given the two efficient optimization steps described above (optimizing over all subsets of attributes for a given subset of records, and optimizing over all subsets of records for a given subset of attributes), we propose two different search procedures for maximizing the score function $F(S)$ over all subsets of records and attributes. The first approach, which we call "exhaustive FGSS", performs an efficient search over records separately for each of the $2^M$ subsets of attributes. This approach is computationally efficient when the number of attributes is small, and is guaranteed to find the globally optimal subset of records and attributes. However, its run time scales exponentially with the number of attributes (Table 1), with a total complexity of $O(2^M|U|N(M + \log N))$, and thus the exhaustive FGSS approach is not feasible for data sets with a large number of attributes.

Thus we propose the FGSS search procedure that scales well with both $N$ and $M$, using LTSS to efficiently maximize over subsets of records and subsets of attributes. To do so, we first choose a subset of attributes $A \subseteq \{A_1 \ldots A_M\}$ uniformly at random. We then iterate between the two efficient optimization steps described above. We first maximize $F(S)$ over all subsets of records for the current subset of attributes $A$, and set the current set of records as follows:

$$R = \arg\max_{R \subseteq \{R_1 \ldots R_N\}} F(R \times A). \tag{11}$$

We then maximize $F(S)$ over all subsets of attributes for the current subset of records $R$, and set the current set of attributes as follows:

$$A = \arg\max_{A \subseteq \{A_1 \ldots A_M\}} F(R \times A). \tag{12}$$

We continue iterating between (11) and (12) until convergence, at which point we have reached a conditional maximum of the score function ($R$ is conditionally optimal given $A$, and $A$ is conditionally optimal given $R$). This ordinal ascent approach is not guaranteed to converge to the joint optimum

$$\arg\max_{R \subseteq \{R_1 \ldots R_N\}, A \subseteq \{A_1 \ldots A_M\}} F(R \times A),$$

but multiple random restarts can be used to approach the global optimum. We show in §4 that with 50 random restarts, FGSS will locate a near globally optimal subset with high probability. Moreover, if $N$ and $M$ are both large, this iterative search is much faster than an exhaustive search approach, making it computationally feasible to detect anomalous subsets of records and attributes in data sets that are both large and high-dimensional. Each iteration (optimization over records, followed by optimization over attributes) has a complexity of $O(|U|(NM + N \log N + M \log M))$, where $|U|$ is the average number of $\alpha$ thresholds considered. In this expression, the $O(NM)$ term results from aggregating over records and attributes, while the $O(N \log N)$ and $O(M \log M)$ terms result from sorting the records and attributes by priority respectively. Thus the FGSS search procedure has a total complexity of $O(YZ|U|(NM + N \log N + M \log M))$, where $Y$ is the number of random restarts and $Z$ is the average number of iterations required for convergence (Table 1). Since each iteration

| Search Procedure | # of Steps | Optimizing Records | Optimizing Attributes | Aggregating Records and Attributes |
|---|---|---|---|---|
| Exhaustive | $2^M\|U\|$ | $O(N\log N)$ | - | $O(NM)$ |
| Efficient | $YZ\|U\|$ | $O(N\log N)$ | $O(M\log M)$ | $O(NM)$ |
| Exhaustive w/ Similarity Constraints | $2^M\|U\|N$ | $O(k\log k)$ | - | $O(kM)$ |
| Efficient w/ Similarity Constraints | $YZ\|U\|N$ | $O(k\log k)$ | $O(M\log M)$ | $O(kM)$ |

Table 1: Outline of the computational complexity of each FGSS search procedure. From left to right the columns describe: the particular search procedure, the number of optimization steps required, the complexity of sorting over records per optimization step, the complexity of sorting over attributes per optimization step, and the complexity of aggregating over records and attributes per optimization step.

Variable Definitions:

| | |
|---|---|
| $\|U\|$ | is the average number of $\alpha$ thresholds considered. |
| $Y$ | is the number of random restarts (efficient methods only). |
| $Z$ | is the average number of iterations required for convergence (efficient methods only). |
| $M$ | is the number of attributes. |
| $N$ | is the number of records. |
| $k$ | is the average neighborhood size corresponding to distance threshold $r$ (similarity-constrained methods only). |

step optimizes over all subsets of records (given the current subset of attributes) and all subsets of attributes (given the current subset of records), convergence is extremely fast, with average values of $Z$ less than 3.0 for all of our experiments described below.

### 2.5 Incorporating Similarity Constraints

The search approaches described above exploit the linear-time subset scanning property to efficiently identify the unconstrained subset of records and attributes that maximizes the score function $F(S)$. However, the unconstrained optimal subset may contain unrelated records, while records generated by the same anomalous process are expected to be similar to each other. The self-similarity of the detected subsets can be ensured by enforcing a similarity constraint. We augment the FGSS search procedure by defining the "local neighborhood" of each record in the test data set, and then performing an unconstrained FGSS search for each neighborhood, where $F(S)$ is maximized over all subsets of attributes and over all subsets of records contained within that neighborhood. Given a metric $d(R_i,R_j)$ which defines the distance between any two data records, we define the local neighborhood of $R_i$ as $\{R_j : d(R_i,R_j) \leq r\}$, where $r$ is some predefined distance threshold. We then find the maximum score over all similarity-constrained subsets. The FGSS constrained search procedure

has a complexity of $O(YZ|U|N(kM + k\log k + M\log M))$, where $k$ is the average neighborhood size (number of records) corresponding to distance threshold $r$ (Table 1).

In the constrained case, the value of $|U|$ tends to be small, and we observed $|U| < 20$ for all of our experiments described below. For small numbers of attributes, $|U|$ is upper bounded by the number of distinct likelihood values $l_{ij}$ with corresponding $p_{\max}(p_{ij}) \leq \alpha_{\max}$ in the conditional probability tables of the Bayesian network learned in the first step of the FGSS algorithm, as in the unconstrained case. For larger numbers of attributes, the neighborhood size $k$ tends to decrease, and since all of the records in the neighborhood differ in at most $r$ attributes, we note that $|U|$ is upper bounded by $M + (k-1)r$. This is because the center record could have $M$ distinct values of $p_{\max}$, while each other record in the neighborhood could only have $r$ distinct values of $p_{\max}$ not contained in the center record. In practice, we expect $|U|$ to be far lower than this because of duplicates and because many attribute values have $p_{\max}(p_{ij})$ greater than $\alpha_{\max}$.

### 2.6 Statistical Significance Testing

The FGSS algorithm is designed to detect and report the most anomalous subsets of a large test data set. However, by scanning over many different subsets of records and attributes, and computing the maximum of these scores, we may see "large" scores simply due to chance. FGSS can avoid this problem, commonly referred to as *multiple hypothesis testing*, by simply reporting the highest scoring subsets without drawing conclusions as to whether or not their scores are high enough to be considered significant. Alternatively, we can correct for multiple testing by randomization, and then only report the statistically significant subsets. To perform randomization testing, we create a large number $T$ of "replica" data sets under the null hypothesis, perform the same scan (maximization of $F(S)$ over self-similar subsets of records and attributes) for each replica data set, and compare the maximum subset score for the original data to the distribution of maximum subset scores for the replica data sets. More precisely, we create each replica data set, containing the same number of records as the original test data set, by sampling uniformly at random from the training data or by generating random records according to our Bayesian network representing $H_0$. We then use the previously described steps of the FGSS algorithm to find the score of the most anomalous subset $F^* = \max_S F(S)$ of each replica. We can then determine the statistical significance of each subset $S$ detected in the original test data set by comparing $F(S)$ to the distribution of $F^*$. The $p$-value of subset $S$ can be computed as $\frac{T_{\text{beat}}+1}{T+1}$, where $T_{\text{beat}}$ is the number of replicas with $F^*$ greater than $F(S)$ and $T$ is the total number of replica data sets. If this $p$-value is less than our significance level $fpr$, we conclude that the subset is significant. An important benefit of this randomization testing approach is that the overall false positive rate (i.e., the probability of reporting any subsets as significant if the null hypothesis $H_0$ is true) is guaranteed to be less than or equal to the chosen significance level $fpr$. However, a disadvantage of randomization testing is its computational expense, which increases run time proportionally to the number of replications performed. Our results discussed in §4 directly compare the scores of "clean" and anomalous data sets, and thus do not require the use of randomization testing.

### 2.7 FGSS Algorithm

Inputs: test data set, training data set, $\alpha_{\max}$, $r$, $Y$.

1. Learn a Bayesian network (structure and parameters) from the training data set.

2. For each data record $R_i$ and each attribute $A_j$, in both training and test data sets, compute the likelihood $l_{ij}$ given the Bayesian network.

3. Compute the $p$-value range $p_{ij} = [p_{\min}(p_{ij}), p_{\max}(p_{ij})]$ corresponding to each likelihood $l_{ij}$ in the test data set.

4. For each (non-duplicate) data record $R_i$ in the test data set, define the local neighborhood $S_i$ to consist of $R_i$ and all other data records $R_j$ where $d(R_i, R_j) \leq r$.

5. For each local neighborhood $S_i$, iterate the following steps $Y$ times. Record the maximum value $F^*$ of $F(S)$, and the corresponding subsets of records $R^*$ and attributes $A^*$ over all such iterations:

   (a) Initialize $A \leftarrow$ random subset of attributes.

   (b) Repeat until convergence:

      i. Maximize $F(S) = \max_{\alpha \leq \alpha_{\max}} F_\alpha(R \times A)$ over subsets of records $R \subseteq S_i$ in the local neighborhood, for the current subset of attributes $A$, and set $R \leftarrow \arg\max_{R \subseteq S_i} F(R \times A)$.

      ii. Maximize $F(S) = \max_{\alpha \leq \alpha_{\max}} F_\alpha(R \times A)$ over all subsets of attributes $A$, for the current subset of records $R$, and set $A \leftarrow \arg\max_{A \subseteq \{A_1 \ldots A_M\}} F(R \times A)$.

6. Output $S^* = R^* \times A^*$.

7. Optionally, perform randomization testing, and report the $p$-value of $S^*$.

## 3. Related Work

In this section, we briefly contrast the theoretical contributions of this work with the previous work of Neill (2011, 2012) as well as describe two other recently proposed methods for anomalous pattern detection in general categorical data sets: Anomaly Pattern Detection (APD) and Anomalous Group Detection (AGD). In §4, we directly compare the detection performance of our new FGSS method to APD and AGD along with the simple Bayesian network anomaly detection method defined above in the domains of customs monitoring, disease surveillance, and network intrusion detection.

### 3.1 Anomaly Pattern Detection

Anomaly Pattern Detection (APD) (Das et al., 2008) attempts to solve the problem of finding anomalous records in a categorical data set through a two-step approach. The first step is to evaluate the anomalousness of each individual record using a local anomaly detector. Local anomaly detectors are typically simple methods that use characteristics of the individual data record to determine its anomalousness. Das et al. (2008) defined two local anomaly detectors for use within the APD framework. Here we focus on the BN method, which defines the anomalousness of a record as inversely proportional to the likelihood of that record given the Bayesian network learned from the training data; all likelihoods below some threshold value are considered anomalous. The second step evaluates a set of candidate rules, each consisting of a conjunction of attribute values. For example, in Emergency Department data, one possible rule could be "hospital id = 5 AND prodrome = respiratory". Each rule is scored by comparing the observed and expected numbers of individually

anomalous records with the given attribute values, using Fisher's Exact Test. When the number of individually anomalous records is significantly higher than expected, that rule is considered anomalous.

Some of the limitations of APD stem from its dependency on searching over "rules": more specifically, APD enforces a stringent constraint which allows records to be grouped together if and only if they share certain attribute values. All records that share these attribute values will be evaluated together, but it is conceivable that the true anomalies only make up a small fraction of the records that satisfy a given rule. Second, considering all conjunctions of attribute values would be computationally infeasible, and thus only rules containing no more than two attributes are considered. Although this reduction of the search space reduces the run time of APD, it can also adversely affect detection ability. Many of the relevant and interesting patterns we wish to detect may affect more than two attributes, and APD will likely lose power to detect such patterns. Also, APD bases the score of a rule on the number of (perceived) individually anomalous records that satisfy it. Thus we do not expect it to perform well in cases where each individual record is not highly anomalous, and the anomalous pattern is only visible when the records are considered as a group. Finally, APD, unlike FGSS, lacks the ability to provide accurate insight into the subset of attributes or relationships for which a given group of records is anomalous. The patterns returned by APD are simply constraints used to group records for the purpose of searching; in §4.5 we show that these do not correspond well to the true anomalous subset of attributes.

### 3.2 Anomalous Group Detection

Anomalous Group Detection (AGD) (Neill et al., 2008; Das, 2009) is a method designed to find the most anomalous groups of records in a categorical data set. AGD attempts to solve this problem in a loosely constrained manner, improving on a limitation of APD, such that any arbitrary group of anomalous records can be detected and reported. AGD identifies subsets of records $S$ that maximize the likelihood ratio statistic $F(S) = \frac{P(Data_S \mid H_1(S))}{P(Data_S \mid H_0)}$. In this expression, the null hypothesis is represented as a "global" Bayesian network with structure and parameters learned from the training data. Each alternative hypothesis $H_1(S)$ is represented by a "local" Bayesian network, which maintains the same structure as the global Bayesian network but learns parameter values using only the subset of records $S$. A subset that has a large $F(S)$ is one whose records are mutually very likely given the local Bayesian network (self-similar) but are dissimilar to the records outside of subset $S$. The self-similarity metric used by AGD addresses some limitations of APD's rule-based metric, allowing for less stringent constraints in the formation of groups of records. However, with this approach, it is still computationally infeasible to maximize over all possible subsets of records. Therefore, AGD relies on a greedy search heuristic to reduce the search space, with no guarantee that it will find the subset of records which maximizes $F(S)$. Furthermore, for the subsets it does return, AGD does not provide any additional information useful for characterizing the underlying anomalous process, such as the affected subset of attributes.

### 3.3 Fast Subset Scan and Fast Subset Sums

The previous work of Neill (2011, 2012), like the present work, presents new methods for efficient detection of anomalous patterns. However, both previous approaches focus on the domain of spatial event detection, where one or more count data streams are monitored across a collection of spatial locations and over time, with the goal of identifying space-time regions with significantly higher

than expected counts. Our work builds on Neill (2012), which defines and lays the theoretical foundations for the LTSS property, proves that many parametric, univariate spatial and space-time scan statistics satisfy LTSS, and shows how this property can be used for "fast subset scanning" over proximity-constrained subsets of locations. We extend LTSS to detect self-similar, anomalous subsets of records and attributes in general multivariate data, where many of the traditional parametric assumptions found in space-time detection fail to hold. Thus we demonstrate that a general class of nonparametric scan statistics satisfy the necessary conditions of LTSS, and provide an algorithmic framework for optimizing these statistics over subsets of records and attributes. Neill (2011) describes a methodological approach which is very different from both Neill (2012) and this work. It optimizes the Bayesian framework of Neill and Cooper (2010) for integrating prior information and observations from multiple data streams, assuming a known set of event types to be detected. Unlike Neill (2012) and the present work, this "fast subset sums" approach does not identify a most anomalous subset of the data, but instead efficiently computes the posterior probability that each event type has affected each monitored location.

## 4. Evaluation

In this section, we compare the performance of FGSS to the previously proposed AGD, APD, and BN approaches. We consider data sets from three distinct application domains (customs monitoring, disease surveillance, and network intrusion detection) in order to evaluate each method's ability to detect anomalous patterns. These data sets are described in §4.1-§4.3 respectively, along with the evaluation results for each domain. In §4.4, we consider the scalability and evaluate the run times of the competing methods, and in §4.5 we compare the methods' ability to accurately characterize the detected patterns.

We define two metrics for our evaluation of detection power: area under the precision/recall (PR) curve, which measures how well each method can distinguish between anomalous and normal records, and area under the receiver operating characteristic (ROC) curve, which measures how well each method can distinguish between data sets which contain anomalous patterns and those in which no anomalous patterns are present. In each case, a higher area under the curve (AUC) corresponds to better detection performance.

To precisely define these two metrics, we first note that three different types of data sets are used in our evaluation. The *training data set* only contains records representing typical system behavior (i.e., no anomalous patterns are present) and is used to learn the null model. Each *test data set* is composed of records that represent typical system behavior as well as anomalous groups, while each *normal data set* has the same number of records as the test data sets but does not contain any anomalous groups.

For the PR curves, each method assigns a score to each record in each test data set, where a higher score indicates that the record is believed to be more anomalous, and we measure how well the method ranks true anomalies above non-anomalous records. The list of record scores returned by a method are sorted and iterated through: at each step, we use the score of the current record as a threshold for classifying anomalies, and calculate the method's precision (number of correctly identified anomalies divided by the total number of predicted anomalies) and recall (number of correctly identified anomalies divided by the total number of true anomalies). For each method, the area under the PR curve is computed for each of the 50 test data sets, and its average AUC and standard error are reported.

For the ROC curves, each method assigns a score to each test and normal data set, where a higher score indicates that the data set is believed to be more anomalous, and we measure how well the method ranks the test data sets (which contain anomalous groups) above the normal data sets (which do not contain anomalous groups). For each method, the algorithm is run on an equal number of data sets containing and not containing anomalies. The list of data set scores returned by a method are sorted and iterated through: at each step, we compute the true positive rate (fraction of the 50 test data sets correctly identified as anomalous) and false positive rate (fraction of the 50 normal data sets incorrectly identified as anomalous). The area under the ROC curve is computed for each method along with its standard error.

To compute the PR and ROC curves, each method must return a score for every record in each data set, representing the anomalousness of that record, as well as a score for the entire data set. For the BN method, the score of a record $R_i$ is the negative log-likelihood of that record given the Bayesian network learned from training data, and the score of a data set is the average negative log-likelihood of the individual records it contains. For the AGD method, the score of a record $R_i$ is the score of the highest scoring group of which that record is a member: $\text{Score}(R_i) = \max_{S\,:\,R_i \in S} F(S)$. Similarly, the score of a data set is the score of its highest scoring group (Neill et al., 2008; Das, 2009). For the APD method, all records that belong to a significant pattern are ranked above all records that do not belong to a significant pattern; within each of these subsets of records, the individual records are ranked using the individual anomaly detector (BN method). Similarly, a data set's score is the score of the most individually anomalous record it contains, with all data sets containing significant patterns ranked above all data sets which do not contain significant patterns (Das et al., 2008).

In our FGSS method, we find the top-$k$ highest scoring disjoint subsets $S$, by iteratively finding the optimal $S$ in our current test data set and then removing all of the records that belong to this group; we repeat this process until we have $k$ groups or have grouped all the test data records. In this framework, a record $R_i$ can only belong to one group, and thus the score of each record $R_i$ is the score of the group of which it is a member. All records that do not belong to a top-$k$ group are grouped together in the $(k+1)^{th}$ group. Within each group, records are sorted from most to least anomalous, that is, from the lowest to the highest record likelihood given the Bayesian network learned from training data. For all of the FGSS results described in this paper, unless otherwise specified, we use the similarity-constrained FGSS search with a top-$k$ of 20, a maximum radius of $r = 1$, and an $\alpha_{\max}$ of 0.1. The score of a data set is defined as the average group score of all grouped records, $\frac{\sum F_i N_i}{\sum N_i}$, where $F_i$ is the score of group $i$ and $N_i$ is the number of records in group $i$.

### 4.1 PIERS Container Shipment Data

This real-world data set contains records of scanned containers imported into the U.S. from various ports in Asia. Customs and border patrol officials wish to examine such data sets in order to identify patterns of shipments which may represent smuggling or other illicit activities so that these containers can be flagged for further inspection. In our data set, each record is described by 10 features, 7 categorical and 3 continuous. The categorical features include the container's country of origin, departing and arriving ports, shipping line, shipper's name, vessel name, and the commodity being shipped. The continuous features, which we discretize into five equal-width bins, include the size, weight, and value of the container. As this data set has no labeled anomalies which could be used

| $N$ | $k_{inj}$ | $s_{inj}$ | $m_{inj}$ | $FGSS-BJ$ | $FGSS-HC$ | $AGD$ | $APD$ | $BN$ |
|---|---|---|---|---|---|---|---|---|
| 1000 | 1 | 10 | 1 | **76.9±3.9** | 52.3±4.7 | 62.2±4.2 | 47.7±4.3 | 18.8±2.7 |
| 1000 | 1 | 10 | 2 | **80.9±3.2** | 67.6±4.0 | 64.9±4.1 | 65.5±4.1 | 38.9±3.7 |
| 1000 | 4 | 25 | 1 | **94.2±1.0** | 61.7±2.9 | **93.0±1.2** | 52.9±2.0 | 43.5±2.2 |
| 1000 | 4 | 25 | 2 | **97.3±1.0** | 87.3±1.6 | 94.3±0.7 | 77.3±1.5 | 73.1±1.8 |
| 1000 | 10 | 10 | 1 | **90.8±1.2** | 62.0±1.9 | 80.4±1.5 | 52.9±1.6 | 39.6±1.5 |
| 1000 | 10 | 10 | 2 | **91.5±0.8** | 85.4±1.0 | 83.5±1.0 | 75.7±1.3 | 71.4±1.2 |
| 10,000 | 4 | 25 | 1 | **79.7±2.5** | 40.2±3.5 | These | 41.8±2.9 | 8.0±1.0 |
| 10,000 | 4 | 25 | 2 | **71.2±1.8** | 65.2±2.5 | runs | 64.2±2.6 | 26.4±1.9 |
| 10,000 | 10 | 10 | 1 | **54.3±2.4** | 41.5±2.7 | did not | 16.2±1.4 | 6.8±0.8 |
| 10,000 | 10 | 10 | 2 | 51.6±2.0 | **65.4±1.7** | complete | 40.2±2.3 | 26.7±1.4 |

Table 2: PIERS Container Shipment Data: Average area (in percent) under the PR curve, with standard errors. For each row, the method which demonstrates the best performance, and those methods with performance not significantly different at significance level $\alpha = 0.05$, are bolded.

| $N$ | $k_{inj}$ | $s_{inj}$ | $m_{inj}$ | $FGSS-BJ$ | $FGSS-HC$ | $AGD$ | $APD$ | $BN$ |
|---|---|---|---|---|---|---|---|---|
| 1000 | 1 | 10 | 1 | **94.2±2.7** | 87.1±3.5 | 76.6±3.8 | 77.2±4.4 | 66.6±4.1 |
| 1000 | 1 | 10 | 2 | **97.8±1.8** | **95.6±2.2** | 78.8±3.4 | 82.3±4.1 | 71.3±3.5 |
| 1000 | 4 | 25 | 1 | **1±0** | **1±0** | **1±0** | **1±0** | **98.3±1.0** |
| 1000 | 4 | 25 | 2 | **1±0** | **1±0** | **1±0** | **1±0** | **1±0** |
| 1000 | 10 | 10 | 1 | **1±0** | **1±0** | **1±0** | **1±0** | **99.7±0.2** |
| 1000 | 10 | 10 | 2 | **1±0** | **1±0** | **1±0** | **1±0** | **1±0** |
| 10,000 | 4 | 25 | 1 | **99.9±0.1** | **97.4±1.7** | These | **99.3±0.6** | 78.6±4.0 |
| 10,000 | 4 | 25 | 2 | **1±0** | **1±0** | runs | **1±0** | 97.4±1.1 |
| 10,000 | 10 | 10 | 1 | **99.9±0.1** | **1±0** | did not | 94.1±2.8 | 82.8±3.8 |
| 10,000 | 10 | 10 | 2 | **1±0** | **1±0** | complete | **99.8±0.2** | 96.3±1 |

Table 3: PIERS Container Shipment Data: Average area (in percent) under the ROC curve, with standard errors. For each row, the method which demonstrates the best performance, and those methods with performance not significantly different at significance level $\alpha = 0.05$, are bolded.

as a gold standard, our evaluation approach is to inject synthetic anomalous groups into the test data sets.

To create a group of anomalies, we first make $s_{inj}$ identical copies of a randomly chosen record. A subset of attributes $A_{inj}$ is then chosen at random; each of the identical records in the group is then modified by randomly redrawing its values for this subset of attributes. The new value for each attribute is drawn from the marginal distribution of that attribute in the training data set. The records within the injected group are self-similar, as each pair of records differs by at most $m_{inj} = |A_{inj}|$ attributes. Each record in the injected group may be subtly anomalous, since randomly changing an attribute value breaks the relationship of that attribute with the remaining attributes. One possible

real world scenario where such an anomalous group might occur is when a smuggler attempts to ship contraband using methods which have proved successful in the past, thus creating a group of similar, subtly anomalous container shipments.

We performed ten different experiments which differed in four parameters: the number of records $N$ in the test data sets, the number of injected groups $k_{inj}$, the number of records per injected group $s_{inj}$, and the number of randomly altered attributes $m_{inj}$. In each case, 50 test data sets were created, along with an additional 50 normal data sets (containing the same number of records as the test data sets, but with no injected anomalies). A separate training data set containing 100,000 records was generated for each experiment; the training and normal data sets are assumed to contain only "normal" shipping patterns with no anomalous patterns of interest. Results of these experiments are summarized in Tables 2 and 3.

Table 2 compares each method's average area under the PR curve across the various PIERS scenarios, thus evaluating the methods' ability to distinguish between anomalous and normal records in the test data sets. We observe that FGSS-BJ (using the Berk-Jones nonparametric scan statistic) demonstrated significantly higher AUC than all other methods in nine of the ten experiments, while FGSS-HC (using the Higher Criticism nonparametric scan statistic) demonstrated significantly higher AUC than all other methods in the remaining experiment. Both FGSS-BJ and FGSS-HC consistently outperformed APD and BN; FGSS-BJ outperformed AGD in all experiments, while FGSS-HC underperformed AGD when only a single attribute was affected. All methods tended to have improved performance when the proportion of anomalies $k_{inj}s_{inj}/N$ was larger, when the group size $s_{inj}$ was larger, and when the records were more individually anomalous (corresponding to a larger number of randomly changed attributes $m_{inj}$). However, several differences between methods were noted. FGSS-BJ and AGD both experienced only slight improvements in performance when the number of randomly changed attributes $m_{inj}$ was increased from 1 to 2, while FGSS-HC, APD, and BN experienced large improvements in performance for $m_{inj} = 2$. This suggests that FGSS-HC, APD, and BN rely more heavily on the individual anomalousness of data records, while FGSS-BJ and AGD rely more heavily on the self-similarity of a group of records, each of which may only be subtly anomalous. AGD performed almost as well as FGSS-BJ when the proportion of anomalies and the group size were large, but its performance degraded for a small (1%) proportion of anomalies. Moreover, we were unable to compute results for AGD on data sets containing $10,000$ records, as each run of AGD (on a single test data set) required approximately one week to complete.

Table 3 compares each method's average area under the ROC curve across the various PIERS scenarios, thus evaluating the methods' ability to distinguish between the test data sets (which contain anomalous patterns) and the equally-sized normal data sets (in which no anomalies are present). For the two experiments with 1000 records and 1% anomalies, the two FGSS methods significantly outperformed AGD, APD, and BN. For 1000 records and 10% anomalies, all methods performed extremely well. For 10,000 records and 1% anomalies, the FGSS methods and APD performed well, while BN exhibited significantly reduced performance and (as noted above) the AGD runs did not complete.

### 4.2 Emergency Department Data

This real-world data set represents visits to hospital Emergency Departments in Allegheny County, Pennsylvania during the year 2004. Each record represents a patient visit characterized by five categorical attributes: hospital id, prodrome, age decile, gender, and patient home zip code. As in

| Method | PR | ROC |
|---|---|---|
| *FGSS − BJ* | 63.8±2.5 | **95.4±1.7** |
| *FGSS − HC* | 49.7±2.1 | 89.1±3.3 |
| *AGD* | **74.3±2.4** | 93.2±2.5 |
| *APD* | 51.5±1.9 | 91.6±2.2 |
| *BN* | 47.6±2.0 | 84.8±4.2 |

Table 4: Emergency Department Data: Average area (in percent) under the PR curve and ROC curve, with standard errors. For each column, the method which demonstrates the best performance, and those methods with performance not significantly different at significance level $\alpha = 0.05$, are bolded.

Das (2009), we inject simulated respiratory cases resembling an anthrax outbreak. The simulated cases of anthrax were produced by a state-of-the-art simulator (Hogan et al., 2007) that implements a realistic simulation model of the effects of an airborne anthrax release on the number and spatial distribution of respiratory ED cases. We treat the first two days of the attack as the test data, thus evaluating a method's ability to detect anthrax attacks within two days of the appearance of symptoms. It is important for a method to detect the outbreak within these first two days, as early detection and characterization have the potential to significantly decrease morbidity and mortality. Early outbreak detection is difficult, however, as there are typically a small number of observed cases, resulting in only an extremely weak signal. We acknowledge that the challenge of discovering the presence of a subtle, emerging event in space-time data is better addressed by spatial event detection methods (Kulldorff and Nagarwalla, 1995; Kulldorff, 1997; Neill et al., 2005; Neill, 2009) rather than these general methods, but only the general pattern detection approaches are considered in this work. We train the methods on the previous 90 days of data, and evaluate how well each method can detect the signal of an outbreak. Though the simulator provides a detailed model for the effects of an anthrax release, none of the methods are given any information from it. This allows us to test each method's ability to recognize a realistic, but previously unknown, disease outbreak.

Table 4 compares each method's average area under the PR and ROC curves. We observe that AGD demonstrates the best performance for identifying which records are anomalous (as measured by area under the PR curve). In our disease surveillance scenario, this corresponds to best identifying which ED visits correspond to anthrax cases in the event of an attack. However, FGSS-BJ demonstrates the best performance for identifying which data sets are anomalous (as measured by area under the ROC curve). In our disease surveillance scenario, this corresponds to detecting that an anthrax attack has occurred. FGSS-HC, APD, and BN all perform poorly compared to FGSS-BJ and AGD. These results support our understanding of the various detection methods, as the data records corresponding to anthrax-related ED visits are not extremely individually anomalous, but there are a large number of similar cases.

### 4.3 KDD Cup Network Intrusion Data

The KDD Cup Challenge of (1999) was designed as a supervised learning competition for network intrusion detection. Contestants were provided a data set where each record represents a single

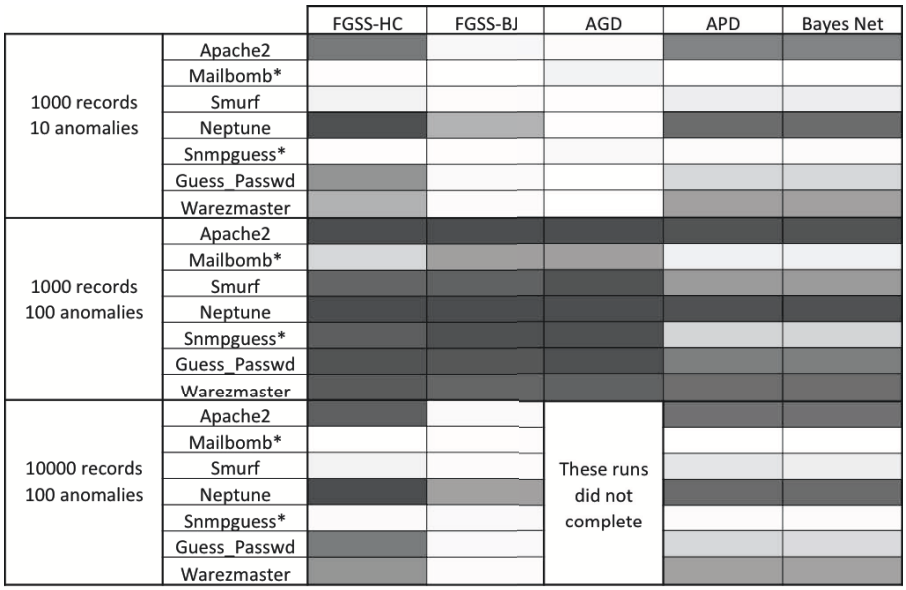



Figure 1: KDD Network Intrusion Data: Heat map of the average area under the PR curve (measuring performance for distinguishing affected vs. unaffected data records). Darker shades correspond to higher areas under the curve (i.e., better performance). (*) indicates experiments for which the FGSS input parameters were adjusted, as discussed in the text.

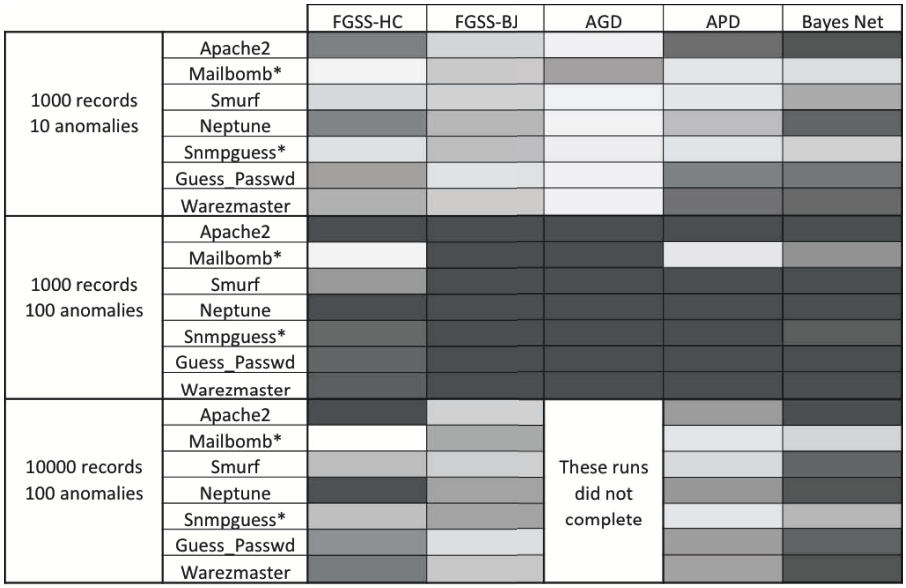



Figure 2: KDD Network Intrusion Data: Heat map of the average area under the ROC curve (measuring performance for distinguishing affected vs. unaffected data sets). Darker shades correspond to higher areas under the curve (i.e., better performance). (*) indicates experiments for which the FGSS input parameters were adjusted, as discussed in the text.

connection to a simulated military network environment. Each record was labeled as belonging to normal network activity or one of a variety of known network attacks. The 41 features of a record, most of which are continuous, represent various pieces of information extracted from the raw data of the connection. As a result of the provided labels, we can generate new, randomly sampled data sets either containing only normal network activity or normal activity injected with examples of a particular intrusion type. The anomalies from a given intrusion type are likely to be both self-similar and different from normal activity, as they are generated by the same underlying anomalous process. These facts should make it possible to detect intrusions by identifying anomalous patterns of network activity, without requiring labeled training examples of each intrusion type. Das (2009) notes that using all 41 features makes the anomalies very individually anomalous, such that any individual record anomaly detection method could easily distinguish these records from normal network activity. In this case, methods that search for groups of anomalies also achieve high performance, but the differences between methods are not substantial. Thus, following Das (2009), we use a subset of 22 features that provide only basic information for the connection, making the anomalies less obvious and the task of detecting them more difficult. We also use the same seven common attack types as described by Das (2009), and discretize all continuous attributes to five equal-width bins.

In Figure 1 and Figure 2 respectively, we compare the areas under the PR and ROC curves for the different methods, for each injection scenario and intrusion type. We observe very different results for the cases of 1% and 10% injected anomalies. For 1% anomalies, FGSS-HC tends to have highest area under the PR curve, indicating that it is best able to distinguish between anomalous and normal records; FGSS-HC and BN tend to have highest area under the ROC curve, indicating that these methods are best able to distinguish between normal data sets and those containing anomalous patterns. These results are consistent with what we understand about the data and the various methods, since records generated by most of the attack types are individually highly anomalous, and FGSS-HC tends to detect smaller subsets of more individually anomalous records. When the proportion of anomalies is increased to 10%, all methods tend to demonstrate higher performance, as measured by area under the PR and ROC curves. However, now FGSS-BJ and AGD achieve the highest detection performance, with near-perfect ability to distinguish between normal and attack scenarios. These results, while suggesting that the optimal choice of detection method is highly dependent on the type and severity of the network attack, demonstrate that FGSS can successfully detect intrusions across multiple scenarios given appropriate choices of the scan statistic (BJ versus HC) and parameters.

We use alternate values of the FGSS parameters for two of the attack types, Mailbomb and Snmpguess. We separate these two attacks from the others because of a trait that they alone share. For our subset of 22 attributes, all of the records injected by the Mailbomb attack are identical to each other; after the discretization of continuous attributes, the records injected by the Snmpguess attack are also identical to each other and to many normal records. This atypical case, where all the records of interest are identical, rewards the AGD method, which requires large groups of similar records in order to achieve high detection power. More precisely, AGD attempts to maximize the likelihood ratio statistic $F(S) = \frac{P(Data_S \mid H_1(S))}{P(Data_S \mid H_0)}$. The numerator of this expression becomes large when the injected records are identical, regardless of whether or not the pattern is anomalous, and thus AGD achieves high detection power for these attacks while FGSS (using the standard parameter settings) and other methods perform poorly. However, we demonstrate that the similarity-constrained FGSS-BJ method with adjusted parameter settings of maximum radius $r = 0$ and $\alpha_{\max} = 0.3$ is also able to achieve high performance comparable to AGD, and much better than the other methods which rely

on the individual anomalousness of the records of interest. We acknowledge that it is not typical to know the appropriate degree of self-similarity or the appropriate value of $\alpha_{\max}$ a priori, though these values could easily be learned by cross-validation given labeled training data for a particular attack type.

### 4.4 Computational Considerations and Scalability

As noted above, both the AGD and APD methods reduce the search space to maintain computational tractability, which may also harm detection power. Naively maximizing a score function over all possible subsets of $N$ records is $O(2^N)$, and thus AGD uses a greedy search over subsets of records. Das (2009) describes the complexity of AGD as $O(GCN^2)$ where $G$ is the maximum allowable group size (provided as an input to the algorithm) and $C$ is the number of non-zero values of $N_{mjk}$ in $S$. While the greedy search reduces run time, it may find a suboptimal subset of records, and is still computationally expensive. Similarly, naively maximizing a score function over all possible subsets of $M$ attributes is $O(2^M)$, and thus APD reduces its search space to rules consisting of at most two attributes.

Our search procedure can be used to efficiently maximize the score function over subsets of records while exhaustively searching over subsets of attributes (Exhaustive FGSS) or to efficiently optimize over both subsets of records and subsets of attributes using an iterative search procedure (FGSS). Also, we can enforce similarity constraints on the anomalous groups returned, or perform an unconstrained search over all subsets of records and attributes. Figure 3 compares the run times of FGSS and AGD for varying numbers of records $N$ and attributes $M$. For each $N$ and $M$, run times were averaged over 100 data sets, each randomly sampled from the KDD Cup normal network activity data. We only use the BJ scoring function for these experiments, since BJ and HC run times were nearly identical. Also, the BN and APD methods were omitted from the graphs; both methods had extremely fast run times, never requiring more than twelve seconds for any scenario. Though BN and APD are consistently faster as a result of their severely reduced search spaces, this reduction in run time comes at the expense of detection ability, as demonstrated above.

As shown in Figure 3, all four variants of FGSS scaled approximately linearly with the number of records. Both variants of (non-exhaustive) FGSS scaled approximately linearly with the number of attributes $M$, while exhaustive FGSS scaled exponentially with $M$. We note that the run-time overhead associated with the iterative maximization approach used by FGSS does not typically yield speedups over exhaustive FGSS until $M \geq 12$. Finally, we note that constrained FGSS is much more computationally expensive than unconstrained FGSS when $M$ is small, but is very similar in run time for larger $M$.

In addition to comparing the run times of the efficient and exhaustive versions of FGSS, we can also measure how often the efficient version of FGSS finds the globally optimal subset. We define the *approximation ratio* as the largest value $p$ such that the efficient FGSS method achieves a score within $(100-p)\%$ of the global maximum score (computed by exhaustive FGSS) at least $p\%$ of the time. For example, an approximation ratio of 95% would signify that FGSS achieves a score within 5% of the global maximum with 95% probability. Results were computed for all values of $N$ shown in Figure 3, and for $M \leq 16$ attributes; for larger values of $M$, it was computationally infeasible to run the exhaustive FGSS method to completion. For each scenario, the FGSS unconstrained search achieved an approximation ratio of 98% or better, while the FGSS constrained search, the procedure used by our main FGSS algorithm, achieved a approximation ratio of 100% (finding

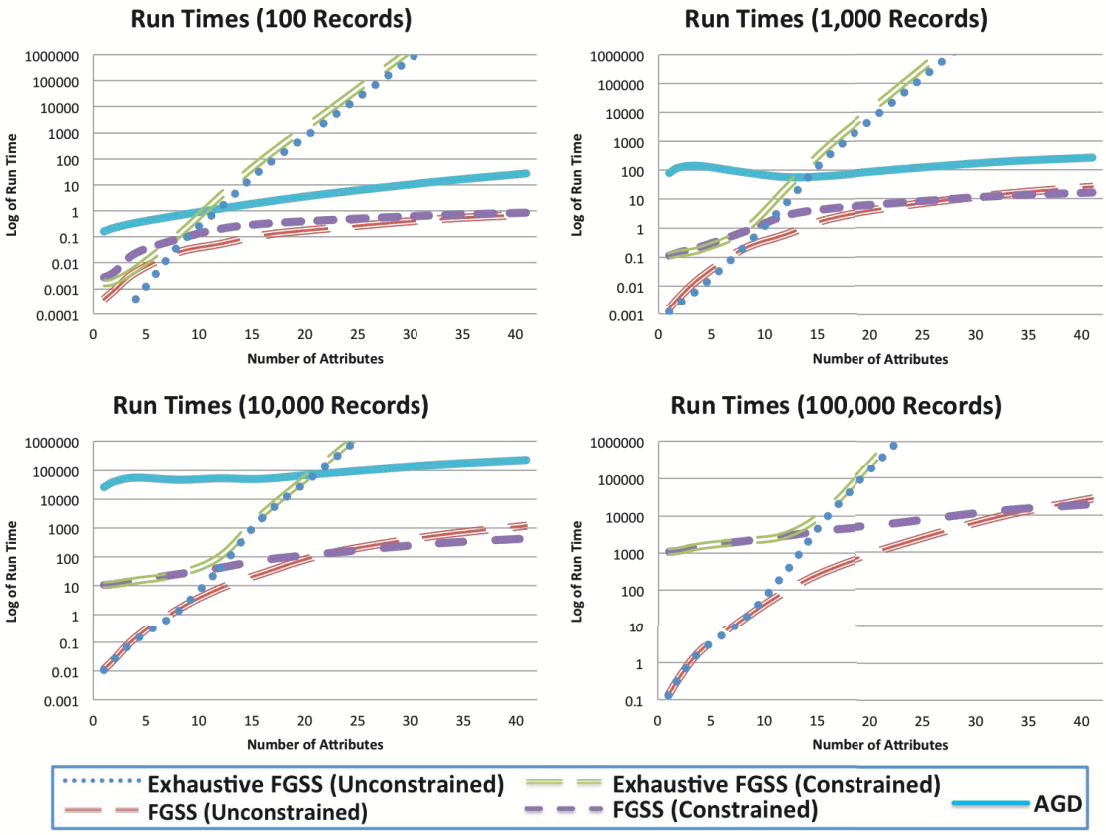



Figure 3: Average run times in seconds of the FGSS and AGD methods, as a function of the numbers of records and attributes.

the exact global optimum for each of the 100 data sets we evaluated). These results empirically demonstrate that very little, if any, detection ability is lost when using the efficient FGSS algorithm to iteratively maximize over subsets of records and attributes.

As can be seen from Figure 3, AGD is considerably slower than our efficient FGSS algorithms, and for small numbers of attributes it is also slower than exhaustive FGSS. The run-time disparity between the various FGSS algorithms and AGD grows with the number of records; we were unable to quantify the difference for data sets of 100,000 records, because AGD required in excess of 24 days to evaluate a single data set with $N = 100,000$ and $M = 1$. In sections §4.1 and §4.3, we were unable to evaluate AGD on data sets of even 10,000 records due to excessive run times. This difference can be attributed to the existence of anomalies in the test data sets used for evaluation: the greedy search procedure used by AGD will continue to grow a group until the maximum group size is reached or the inclusion of the next record does not sufficiently increase the group's score. When the data set contains anomalies, unlike the "normal" data used in this scalability experiment, AGD will find more anomalous records to group together and thus forms larger groups, substantially increasing its run time.

### 4.5 Pattern Characterization

Anomalous pattern detection can be described as a form of knowledge discovery, where the knowledge of interest includes not only the subset of data records affected by an anomalous process, but also which subset of attributes are anomalous for these records. Accurately describing which facets of a subset of data are anomalous can be crucial, particularly when discovering a previously unknown pattern type. In addition to identifying the subset of records affected by an anomalous process, our FGSS method also identifies the subset of attributes for which these records are anomalous. The previously proposed APD method also characterizes anomalous patterns by identifying "rules" which correspond to a higher than expected number of individually anomalous records. However, we hypothesize that the identified attributes may not correspond well to the subset of attributes which are anomalous. To test this hypothesis, we compare the pattern characterization ability of FGSS and APD using the semi-synthetic PIERS data. Recall that to generate an anomalous group in the PIERS data, we selected a subset of attributes at random, and regenerated these attribute values for each affected record. The affected subset of attributes for each record is used as the ground truth to which we can compare the subset of attributes identified as anomalous by a given method. We measure each method's attribute overlap coefficient, defined as

$$\text{Overlap} = \frac{|\text{Predicted Attributes} \cap \text{True Attributes}|}{|\text{Predicted Attributes} \cup \text{True Attributes}|},$$

for each of the PIERS injection scenarios.

However, it is also important to take into consideration the structure of the Bayesian network used to determine the anomalous patterns. The structure of the network is important in characterizing the pattern, as it represents the conditional dependencies between attributes. When an attribute has an anomalous value, either its corresponding likelihood or the likelihoods of its children given the Bayesian network structure will be low. Therefore, our evaluation framework gives a method credit for identifying either the affected attribute or at least one of its children. To do so, prior to computing the overlap coefficient, each method's set of predicted attributes is redefined according to the following logic. Given the set of predicted attributes $A$ and the set of true (affected) attributes $B$, if there exists an attribute $A_i \in A$ with a parent $A_p \in B$, then $A = A \cup \{A_p\}$, that is, the parent attribute is counted as correctly predicted. If it is also the case that $A_i \notin B$, then $A = A \setminus \{A_i\}$, that is, the method is not penalized for identifying the child attribute.

Figure 4 shows a comparison of each method's overlap coefficient for the PIERS data, averaged across the different numbers of groups and attributes injected. We also include a simple $p$-value characterization method in our comparison: this approach evaluates each record in isolation and predicts any attribute whose entire $p$-value range is less than or equal to $\alpha_{\max}$. We feel that this is a more appropriate "straw man" than APD to demonstrate the improvements in characterization ability provided by FGSS. FGSS places all ungrouped records in a group together, and thus we also use the $p$-value characterization method to identify a subset of attributes for each ungrouped record. We allow APD to use all detected patterns in order to identify a subset of attributes for each record, using the most significant pattern for which that record is a member. We observe that FGSS-BJ and FGSS-HC consistently demonstrate significantly better performance than the $p$-value method, and the $p$-value method consistently outperforms APD by a large margin. These results support our hypothesis that the grouping of records that are self-similar and anomalous for some subset of attributes in our FGSS framework results in substantially improved pattern characterization ability.

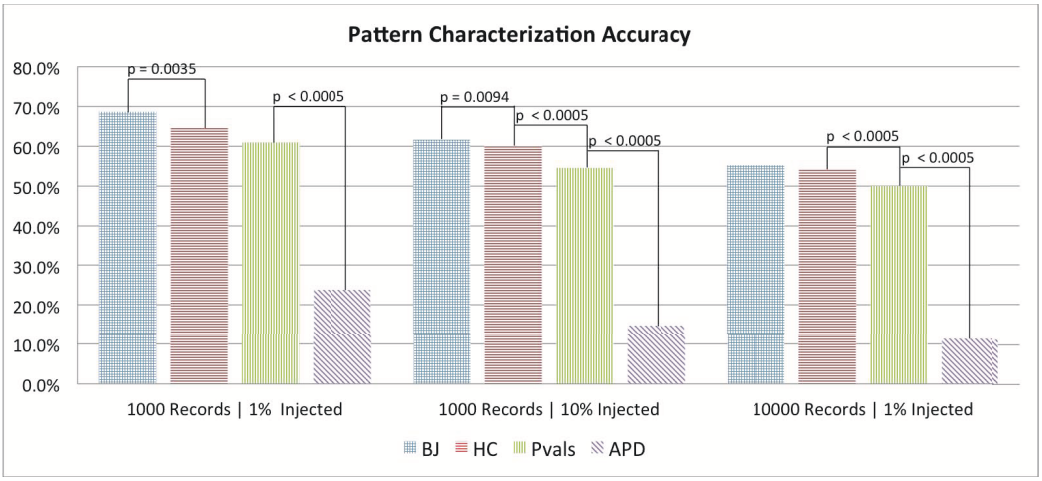



Figure 4: Pattern characterization accuracy for each method, evaluated on the PIERS data set, and averaged across different numbers of groups and attributes injected.

## 5. Conclusions

This paper has presented several contributions to the literature on anomalous pattern detection. We formalize the pattern detection problem as a search over subsets of data records and attributes, and present the Fast Generalized Subset Scan (FGSS) algorithm, which efficiently detects anomalous patterns in general categorical data sets. The FGSS algorithm provides a systematic procedure to map data set values to an unbiased measure of anomalousness, empirical $p$-value ranges. The algorithm then uses the distribution of these empirical $p$-value ranges under the null hypothesis in order to find subsets of data records and attributes that *as a group* significantly deviate from their expectation as measured by a nonparametric scan statistic. We demonstrate that a general class of nonparametric scan statistics satisfy the linear-time subset scanning property (LTSS). This property allows us to search efficiently and exactly over all subsets of data records or attributes while evaluating only a linear number of subsets. These efficient optimization steps are then incorporated into an iterative procedure which jointly maximizes over subsets of records and attributes. Additionally, similarity constraints can be easily incorporated into our FGSS framework, allowing for the detection of self-similar subsets of records which have anomalous values for some subset of attributes.

We provide an extensive comparison between FGSS and other recently proposed pattern detection (AGD, APD) and individual record anomaly detection (BN) methods for general categorical data on semi-synthetic and real-world data sets. Our results indicate that FGSS consistently outperforms the other methods. FGSS excels in scenarios when there is a strong self-similarity among the records generated by an anomalous process, with each individual record only emitting a subtle anomalous signal. FGSS demonstrates improved scalability as compared to AGD, the method with the most comparable detection ability. Furthermore, we empirically demonstrate that, even as FGSS scales to high-dimensional data, it finds the globally optimal subset of records and attributes with high probability. This optimization task can be performed exactly when the number of attributes

is small (e.g., twelve attributes or fewer) using the "exhaustive FGSS" approach described above. When the number of attributes is large, FGSS converges to a conditional maximum (for which the subset of records is optimal given the subset of attributes and vice-versa), and multiple restarts are used to approach the joint optimum. Finally, FGSS not only achieves high detection power but is also able to accurately characterize the subset of attributes for which each identified subset of records is anomalous.

In future work, we plan to extend FGSS in three main directions. Currently FGSS can only handle categorical attributes, which forces it to discretize continuous attributes when evaluating mixed data sets. This constraint only exists because our current method for obtaining record-attribute likelihoods, modeling the conditional probability distribution between attributes with a Bayesian network and using Optimal Reinsertion (Moore and Wong, 2003) to learn the network structure, can only handle categorical attributes. By discretizing continuous attributes, we may lose vital information that would make the task of detecting anomalous patterns easier. Therefore we are currently investigating extensions of FGSS which better exploit the information contained in continuous attributes. We believe that augmenting a Bayesian network, learned only from the categorical attributes, with a regression tree for each continuous attribute will increase the power of FGSS to detect patterns. Second, we are investigating other variants of the nonparametric scan statistic which take into account the dependence between $p$-values for a given record and correctly adjust for the multiplicity of tests. Such statistics might increase detection power as compared to the simpler HC and BJ statistics used here, but it is not clear whether they can be optimized efficiently over subsets of the data. Finally, we are also concerned with FGSS being able to better detect novel patterns of interest. Currently, FGSS only maintains a model $M_0$ describing the distribution of the data when no anomalous patterns are present, but we plan to extend this approach to maintain models for multiple, known pattern types. We will detect subsets of records and attributes that are unlikely given each known pattern model as well as $M_0$, thus enabling FGSS to discover previously unknown pattern types given the current set of known patterns.

## Acknowledgments

The authors would like to thank Sriram Somanchi for fruitful discussions and valuable feedback. This work was partially supported by the National Science Foundation, grants IIS-0916345, IIS-0911032, and IIS-0953330. Additionally, Edward McFowland III was supported by an NSF Graduate Research Fellowship (NSF GRFP-0946825) and an AT&T Labs Fellowship.

## References

KDD Cup, 1999. URL `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.

R. H. Berk and D. H. Jones. Goodness-of-fit test statistics that dominate the Kolmogorov statistics. *Z. Wahrsch. Verw. Gebiete*, 47:47–59, 1979.

D. H. Chau, S. Pandit, and C. Faloutsos. Detecting fraudulent personalities in networks of online auctioneers. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 103–114, 2006.

K. Das. Detecting patterns of anomalies. Technical Report CMU-ML-09-101, PhD thesis, Carnegie Mellon University, Department of Machine Learning, 2009.

K. Das and J. Schneider. Detecting anomalous records in categorical datasets. In *Proceedings of the 13th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 220–229, 2007.

K. Das, J. Schneider, and D. B. Neill. Anomaly pattern detection in categorical datasets. In *Proceedings of the 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2008.

A. C. Davison and D. V. Hinkley. *Bootstrap Methods and Their Applications*. Cambridge University Press, 1997.

D. Donoho and J. Jin. Higher criticism for detecting sparse heterogeneous mixtures. *Annals of Statistics*, 32(3):962–994, 2004.

W. R. Hogan, G. F. Cooper, G. L. Wallstrom, M. M. Wagner, and J.-M. Depinay. The bayesian aerosol release detector: An algorithm for detecting and characterizing outbreaks caused by an atmospheric release of bacillus anthracis. *Statistics in Medicine*, 26:5225–5252, 2007.

M. Kulldorff. A spatial scan statistic. *Communications in Statistics: Theory and Methods*, 26(6): 1481–1496, 1997.

M. Kulldorff and N. Nagarwalla. Spatial disease clusters: detection and inference. *Statistics in Medicine*, 14:799–810, 1995.

A. W. Moore and W.-K. Wong. Optimal reinsertion: A new search operator for accelerated and more accurate bayesian network structure learning. In *Proceedings of the 20th International Conference on Machine Learning*, pages 552–559. AAAI Press, 2003.

D. B. Neill. Expectation-based scan statistics for monitoring spatial time series data. *International Journal of Forecasting*, 25:498–517, 2009.

D. B. Neill. Fast Bayesian scan statistics for multivariate event detection and visualization. *Statistics in Medicine*, 30(5):455–469, 2011.

D. B. Neill. Fast subset scan for spatial pattern detection. *Journal of the Royal Statistical Society (Series B: Statistical Methodology)*, 74(2):337–360, 2012.

D. B. Neill and G. F. Cooper. A multivariate Bayesian scan statistic for early event detection and characterization. *Machine Learning*, 79:261–282, 2010.

D. B. Neill and J. Lingwall. A nonparametric scan statistic for multivariate disease surveillance. *Advances in Disease Surveillance*, 4:106, 2007.

D. B. Neill, A. W. Moore, M. R. Sabhnani, and K. Daniel. Detection of emerging space-time clusters. In *Proceedings of the 11th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2005.

D. B. Neill, G. F. Cooper, K. Das, X. Jiang, and J. Schneider. Bayesian network scan statistics for multivariate pattern detection. In J. Glaz, V. Pozdnyakov, and S. Wallenstein, editors, *Scan Statistics: Methods and Applications*, 2008.

Journal of Machine Learning Research 14 (2013) 1563-1603 Submitted 11/10; Revised 9/12; Published 6/13

# Sub-Local Constraint-Based Learning of Bayesian Networks Using A Joint Dependence Criterion

**Rami Mahdi** RAMIMAHDI@YAHOO.COM
*Department of Genetic Medicine*
*Weill Cornell Medical College*
*New York, NY, 10065, USA*

**Jason Mezey** JGM45@CORNELL.EDU
*Department of Biological Statistics and Computational Biology*
*Cornell University*
*Ithaca, NY, 14853, USA*

**Editor:** Peter Spirtes

## Abstract

Constraint-based learning of Bayesian networks (BN) from limited data can lead to multiple testing problems when recovering dense areas of the skeleton and to conflicting results in the orientation of edges. In this paper, we present a new constraint-based algorithm, light mutual min (LMM) for improved accuracy of BN learning from small sample data. LMM improves the assessment of candidate edges by using a ranking criterion that considers conditional independence on neighboring variables at both sides of an edge simultaneously. The algorithm also employs an adaptive relaxation of constraints that, selectively, allows some nodes not to condition on some neighbors. This relaxation aims at reducing the incorrect rejection of true edges connecting high degree nodes due to multiple testing. LMM additionally incorporates a new criterion for ranking v-structures that is used to recover the completed partially directed acyclic graph (CPDAG) and to resolve conflicting v-structures, a common problem in small sample constraint-based learning. Using simulated data, each of these components of LMM is shown to significantly improve network inference compared to commonly applied methods when learning from limited data, including more accurate recovery of skeletons and CPDAGs compared to the PC, MaxMin, and MaxMin hill climbing algorithms. A proof of asymptotic correctness is also provided for LMM for recovering the correct skeleton and CPDAG.

**Keywords:** Bayesian networks, skeleton, constraint-based learning, mutual min

## 1. Introduction

Learning a Bayesian network (BN) from observational data is a reverse engineering process that can provide insight into the direct relations between observed variables and can be used to establish causation (Pearl, 1988; Spirtes et al., 2001). BNs have been used in learning applications in a number of fields including systems biology (Friedman et al., 2000; Friedman, 2004), medicine (Cowell et al., 1999), and artificial intelligence (Russell and Norvig, 2009) where emerging applications and the growing availability of complex data sets containing thousands of variables are requiring faster, more scalable, and more accurate methods.

Unconstrained learning of a BN is a search for a network that fits the observational data with the highest posterior probability. However, due to the large number of all possible networks, which

©2013 Rami Mahdi and Jason Mezey.

is super exponential in the number of variables being modeled, an exhaustive search is not possible for more than a few tens of variables (Chickering et al., 2004) while heuristic search methods tend to converge to suboptimal solutions. This scalability issue is a known limitation of score-based methods, where a scoring criterion such as the Bayesian information criterion (BIC) (Cooper and Herskovits, 1992) or the minimum description length criterion (MDL) (Lam and Bacchus, 1994) is used to rank candidate networks.

An alternative learning approach to score-based search is the use of conditional independence testing, also referred to as constraint-based learning. Methods in this class such as the IC algorithm (inductive causation) (Pearl, 1988), PC algorithm (Spirtes et al., 2001), and TPDA algorithm (three-phase dependency analysis) (Cheng et al., 2002), first recover the skeleton of the network and edges are oriented afterward. The learning is performed in a way to ensure that the resulting network is consistent with the conditional independences/dependencies entailed by the observations. Under the faithfulness assumption (see Section 3 for definition), constraint-based methods were shown to recover the correct network as the size of the observed samples approaches infinity (Zhang and Spirtes, 2003; Kalisch and Bühlmann, 2007). Moreover, their computational complexity has a polynomial order when the maximum number of connections per node is bounded. Constraint-based methods have also been used in combination with score-based search in what typically is referred to as hybrid learning. Hybrid methods such as the SC algorithm (sparse candidate) (Friedman et al., 1999), MMHC algorithm (MaxMin hill climbing) (Tsamardinos et al., 2006), and the COS algorithm (constrained optimal search) (Perrier et al., 2008) first recover a super structure of the skeleton using a constraint-based approach. Afterward, a constrained score-based search is used to find an optimal network where the search is restricted only to edges existing in the super structure. This strategy can reduce the size of the search space considerably and can lead to higher score solutions.

Although constraint-based learning can, under appropriate assumptions, recover the correct graph in the asymptotic limit, its performance in real applications depends heavily on the accuracy of independence testing, which in turn is sensitive to noise and sample size. As will be discussed in this paper, when learning from small sample data, the use of conditional independence testing can lead to multiple testing problems that deteriorate the accuracy of skeleton recovery and to conflicting results in the orientation of edges. In addition, errors in the first phase of recovering the skeleton can also deteriorate the precision of orienting the edges, resulting in a propagated error.

## 2. Contribution

We present a new constraint-based algorithm, light mutual min (LMM), for learning Bayesian networks that has properties well suited for learning from limited sample data. For skeleton recovery, LMM improves the assessment of candidate edges by using a new mutual dependence criterion that considers conditional independence on subsets of neighboring variables at both sides of an edge simultaneously. In addition, we implement an adaptive relaxation of independence constraints in dense areas of the graph by, selectively, allowing only one node of a connected pair to be aware of the edge. This relaxation is only performed whenever asymmetric evidence of conditional independence is found between a pair of connected variables, where the aim is to reduce the accidental rejection of true edges connecting high degree nodes due to the multiple testing problem in the case of limited training samples. Another consequence of this relaxation is that sets of the recognized neighbors that are used for conditional independence testing remain small, leading to a considerable reduction in the number of required independence tests.

Moreover, we present a new approach to recover v-structures in a given skeleton based on the level of induced dependence caused by common neighbors, and LMM is extended to recover the completed partially directed acyclic graph (CPDAG) of the equivalence class. The name light mutual min is motivated by the fact that the algorithm is fast and uses reduced sets of neighbors (light) in addition to ranking edges using a measure that combines the minimal dependence from both sides of an edge (mutual min). Also, we refer to LMM as a sub-local approach due to using reduced sets of neighbors for independence testing as compared to local algorithms such as the PC and MaxMin algorithms which use subsets of all neighbors for conditional independence testing.

For completeness, a proof of asymptotic correctness is provided to show that the proposed approaches will recover the correct skeleton and CPDAG when the number of observations grows sufficiently large. Also, to empirically assess performance, we compare LMM to the PC and MaxMin algorithms, two of the most popular and computationally efficient BN learning methods (Spirtes et al., 2001; Tsamardinos et al., 2006; Kalisch et al., 2010). Based on empirical evaluation using simulated data, LMM is shown to significantly and consistently outperform both the PC and MaxMin algorithms in recovering the skeleton of the graph in terms of both accuracy and speed when learning from limited sample data. In addition, the extended LMM is found to recover more accurate CPDAGs than the PC algorithm, while being competitive with the MMHC algorithm in the small network case and more accurate in the large network case.

The rest of this paper is organized as follows: Section 3 presents necessary definitions. Section 4 presents related work. Section 5 presents a discussion of the limitations of existing methods. Section 6 presents our proposed methods. Section 7 presents experimental results and comparison to other methods. A proof of asymptotic correctness for the presented methods is provided in the Appendix for the recovery of both the skeleton and the CPDAG.

*Availability:* Our implementation of all presented methods in addition to a supplementary material of further discussion and illustrations are made available with this publication and they also can be accessed at `http://www.mloss.org/software/view/460/` or alternatively at `http://mezeysoftware.bscb.cornell.edu/index.php/LMM`.

## 3. Definitions and Preliminaries

In this section, we present necessary definitions and notations following Pearl (1988) and Neapolitan (2004) with slight variations.

**Definition 1 (Directed Graph)** *A directed graph $G = (V,E)$ consists of a set of nodes representing variables $V = \{1,...,p\}$ and a set of edges $E \subseteq V \times V$ where an edge $(i,j) \in E$ implies that $E(i,j)$ is an edge pointing from $V_i$ (parent) to $V_j$ (child). Two variables $i$ and $j$ are adjacent in $G$ if and only if $(i,j) \in E$ or $(j,i) \in E$.*

In this paper, the notation $CP_i^*$ is used to refer to the true set of child and parent variables of the variable $i$ while the notation $CP_i$ is used to refer to the set of child and parent variables of $i$ that are being inferred by the algorithm. The set of child and parent variables are also sometimes referred to as the neighbor set of $i$.

**Definition 2 (Directed Acyclic Graph (DAG))** *A directed graph $G = (V,E)$ is said to be acyclic if and only if for every node $V_i$ in the graph, there does not exist a path of connected and directed edges such that starting from node $V_i$ and following the direction of edges can lead back to the same node $V_i$.*

**Definition 3 (Skeleton)** *The skeleton of a directed graph $G = (V,E)$ is a graph that contains all the nodes and edges of G such that all the edges have no directions. Every undirected edge in the skeleton represents a parent-child relation without informing who is the parent and who is the child.*

**Definition 4 (Conditional Independence)** *Two variables x and y are conditionally independent given a set of variables $Z_{\setminus x,y}$ w.r.t a probability distribution P, denoted as $x \perp\!\!\!\perp y \mid Z$, if and only if $P(x,y \mid Z = z) = P(x \mid Z = z) \times P(y \mid Z = z)$, $\forall z$ where $P(Z = z) > 0$.*

The notation $Z_{\setminus x,y}$ means $x$ and $y$ are excluded from the conditioning set $Z$ which is always the case for all methods presented in this paper even if it is not explicitly stated. Examples of methods to determine conditional independence are the use of statistical tests of partial correlation and the G-squared measure (Neapolitan, 2004).

**Definition 5 (Bayesian Network)** *A directed acyclic graph $G = (V,E)$ is said to be a Bayesian network w.r.t a probability distribution P if it satisfies the Markov condition (local Markov property): Every variable $x \in V$ is independent of any subset of its non-descendant variables conditioned on the set of its parents.*

**Definition 6 (V-Structure)** *An ordered triplet of nodes $(x,w,y)$ forms a v-structure in a DAG if and only if x and y meet head to head at w ($x \rightarrow w \leftarrow y$) while x and y are not directly connected in the graph.*

**Definition 7 (Faithfulness)** *A graph G and a probability distribution P are said to satisfy the faithfulness condition (or to be faithful to one another) if and only if, based on the Markov condition, G entails all and only the conditional independence relations in P.*

Many researchers have suggested some constraints on BN inference that would facilitate finding sound solutions. The most used of these constraints is the faithfulness property, and it has been argued that, in most cases, the true BN will have such a property (Spirtes et al., 1993). As a consequence of the faithfulness assumption (Spirtes et al., 2001), an edge between a node $x$ and a node $y$ exists in $G$ if and only if there does not exist a set $Z_{\setminus x,y}$ such that $x$ and $y$ are independent when conditioned on $Z_{\setminus x,y}$: $(\nexists Z \subseteq V_{\setminus x,y}\ s.t.\ x \perp\!\!\!\perp y \mid Z, w.r.t\ P)$.

**Definition 8 (Blocked Path)** *In a directed graph $G = (V,E)$, a path Pa of connected edges between two distinct nodes $x,y \in V$ is said to be blocked by a set of nodes $Z \subseteq V_{\setminus x,y}$ if one of the following holds:*

1. *There is a node $w \in Z$ on the path Pa where the edges incident to w on Pa meet head-to-tail at w ($.. \rightarrow w \rightarrow ..$ or $.. \leftarrow w \leftarrow ..$ ).*

2. *There is a node $w \in Z$ on the path Pa where the edges incident to w on Pa meet tail-to-tail at w ($.. \leftarrow w \rightarrow ..$).*

3. *There is a node w on the path Pa, such that w and all of w's descendants are not in Z, and the edges incident to w on Pa meet head-to-head at w ($.. \rightarrow w \leftarrow ..$).*

**Definition 9 (D-Separation)** *In a DAG $G = (V,E)$, two distinct nodes $x,y \in V$ are said to be d-separated by $Z \subseteq V_{\setminus x,y}$, denoted by $Dsep_G(x,y \mid Z)$, if every path between $x$ and $y$ is blocked by $Z$. Moreover, two disjoint sets of nodes $X,Y \subset V$ are said to be d-separated by $Z \subseteq V - (X \cup Y)$ if and only if every $x \in X$ and every $y \in Y$ are d-separated by $Z$.*

**Theorem 10 (Pearl, 1988, D-Separation $\Leftrightarrow$ Conditional Independence)** *Given a faithful BN of a DAG $G$ and a probability distribution $P$, every d-separation in $G$ entails a conditional independence relation in $P$ and every conditional independence relation in $P$ is represented by a d-separation in $G$:*

$$Dsep_G(x,y \mid Z) \Leftrightarrow x \perp\!\!\!\perp y \mid Z \ (w.r.t\ P).$$

**Definition 11 (Markov Equivalence)** *Two DAGs $G_1 = (V,E_1)$ and $G_2 = (V,E_2)$ are called Markov equivalent if for every three mutually disjoint subsets $X,Y,Z \subset V$, $X$ and $Y$ are d-separated by $Z$ in $G_1$ if and only if $X$ and $Y$ are also d-separated by $Z$ in $G_2$:*

$$Dsep_{G_1}(X,Y \mid Z) \Leftrightarrow Dsep_{G_2}(X,Y \mid Z).$$

Based on Theorem 10, two Markov equivalent DAGs entail the same set of conditional independence relations. Also, when given the same observational data, it is possible that there exist multiple equivalent DAGs that are equally likely to have generated the same observations. With the absence of any external information, this equivalence bounds our inference ability to learning the set of equivalent DAGs as opposed to learning a single causal DAG. In spite of this limitation, the structural characteristics shared by equivalent DAGs (Theorem 12), can still be very informative about the underlying causal relations.

**Theorem 12 (Verma and Pearl, 1990, Equivalence Class of DAGs)** *Two DAGs $G^{(1)}$ and $G^{(2)}$ are equivalent if and only if they have the same skeleton and contain the same set of v-structures.*

Typically, the class of equivalent DAGs is represented by the completed partially directed acyclic graph (CPDAG). A partially directed acyclic graph (PDAG) is a graph where some edges are directed and some are undirected. A PDAG is said to be complete if (1) every directed edge exists also in every DAG in the equivalence class of the DAG and (2) for every undirected edge $i - j$, there exists a DAG with $i \rightarrow j$ and a DAG with $i \leftarrow j$ in the same equivalence class.

## 4. Local Constraint-Based Algorithms

Several constraint-based methods were developed to recover the skeleton of BNs (Pearl, 1988; Spirtes et al., 2001; Cheng et al., 2002; Tsamardinos et al., 2006) and all these methods share common properties in that, a local search is typically performed to identify possible marginal or conditional independence between pairs of variables using a statistical test such as the G-squared test or the partial correlation test (Neapolitan, 2004). In this paper, we restrict the discussion and the comparison to two representative methods: the PC and MaxMin algorithms, which are two of the most popular methods. A detailed comparison among several methods, including the PC and MaxMin algorithms, can be found in the work of Tsamardinos et al. (2006).

The PC algorithm (Spirtes et al., 2001) starts with a fully connected graph where unnecessary edges get iteratively deleted one at a time. For every node $V_i$, the conditional independences are tested along the existing edges by conditioning on all subsets of the current neighbors and edges are deleted whenever dependences are found to be insignificant. The extended version of the PC algorithm uses the recovered skeleton to recover the CPDAG of the equivalence class by identifying v-structures and using an additional set of CPDAG orientation rules (Meek, 1995).

In contrast to the PC algorithm, the MaxMin algorithm (Tsamardinos et al., 2006) starts with an empty graph. Afterward, for every node in the graph, the algorithm performs a forward selection of neighbors followed by a backward elimination regardless of which is a parent and which is a child. In either phase, the algorithm tests for independence by conditioning on all subsets of recognized neighbors. In addition, a post-processing step is performed in which edges are deleted if the dependence between two nodes does not appear to be significant from both sides simultaneously. The extended version of the algorithm, MaxMin hill climbing (MMHC), is a hybrid algorithm that uses a score-based search constrained by the recovered skeleton to recover the CPDAG of the equivalence class.

## 5. Difficulties when Learning from Small Sample Data

Using independence testing statistics to learn a BN from small sample data gives rise to a number of issues that can deteriorate the accuracy of both the recovery of the skeleton and the orientation of the edges. In this paper, we will focus on the PC and the MaxMin algorithms as case studies. We note that these issues are a consequence of sampling and do not contradict that these algorithms were proven to recover the correct network in the asymptotic limit.

### 5.1 Unused Conditional Independence Testing Information

In local constraint-based learning algorithms, an edge $E_{xy}$ is usually excluded from the skeleton if either node, $x$ or $y$, finds at least one subset of their neighboring variables to induce complete conditional independence between $x$ and $y$. This approach is analogous to searching for two subsets of variables $Z_x \subseteq CP_x$ and $Z_y \subseteq CP_y$ that induce conditional independence between $x$ and $y$ with the highest statistical confidence based on the observational data. If the maximum of the confidence about conditional independence on either $Z_x$ or $Z_y$ is found to be greater than a threshold $\alpha$, $E_{xy}$ gets excluded from the skeleton and included otherwise. Although this approach is sufficient to recover the correct skeleton in the infinite sample case, its use in learning from limited sample data ignores information about how probable we are to be correct in rejecting the conditional independence hypothesis with the lower confidence. For example, when an edge $E_{xy}$ is being evaluated using two tests of conditional independence with p-values of 0.04 and 0.03, we are more likely to be incorrect to include $E_{xy}$ in the skeleton than if the p-values of the two tests were 0.04 and 0.01. Though we are equally likely to be correct in rejecting the first null hypothesis of conditional independence in both cases, we are more likely to be correct in rejecting the second conditional independence hypothesis in the second case. Therefore, to improve the accuracy of constraint-based learning, candidate edges should be ranked based on a joint confidence criterion that combines the outcome of conditional independence tests at both sides of the edge simultaneously as complementary sources of information.

### 5.2 Increased Type II Error in Dense Regions of the Graph: $P(reject\,E_{xy}|true\,E_{xy})$

The type II error in this context refers to the main null hypothesis that the edge does not exist ($\bar{H}_0 : E_{xy} \notin Skeleton$). In constraint based learning, an edge $E_{xy}$ is typically rejected if at least one conditional independence hypothesis gets accepted when conditioning on all subsets of neighbors ($CP_x$ and $CP_y$). However, as the number of recovered connections to a certain node $x$ increases, the probability of incorrectly inferring conditional independence between $x$ and other variables tends to increase. There are two reasons for this behavior:

1. *Multiple Testing*: In order to recover a correct edge $E_{xy}$, the number of conditional independence tests that must be correctly rejected grows fast with the number of current neighbors ($CP_x$ or $CP_y$). However, due to limited training samples, the probability of incorrectly accepting an individual independence hypothesis is greater than zero. Therefore, the chance of rejecting a correct edge increases as the number of independence tests increases (Tsamardinos and Brown, 2008).

2. *Vanishing Dependence Coefficients*: Conditioning on a larger set of parent and/or child variables of a variable $x$ can, in many cases, lead to smaller dependence coefficients with other parent and child variables (see supplementary material for a poof of multiple cases). For example, conditioning on a larger set of child variables of $x$ can result in smaller partial correlations with its parent or other child variables. This in turn increases the chance of incorrectly deciding that the true partial correlation is zero and hence incorrectly rejecting the edge when learning from small sample data.

The increased type II error is a consequence of the fact that connectivity density varies across the network where some nodes are connected to more neighbors than others. Therefore, a fixed threshold for accepting or rejecting the individual null hypothesis of conditional independence does not take this into account.

### 5.3 Conflicting Results in the Recovery of the CPDAG

The constraint-based method for identifying the direction of BN edges relies on the identification of v-structures based on the concept of separation sets (Meek, 1995; Spirtes et al., 2001). Using this method, it is possible to recover two or more conflicting v-structures (i.e., $x \rightarrow w \leftarrow y$ and $w \rightarrow y \leftarrow u$). The PC algorithm which is the most widely used constraint-based CPDAG recovery algorithm does not offer any resolution for such conflicts. Also, it was not until recently that an algorithm, called Edge-Opt (Fast, 2010), was proposed to resolve conflicting CPDAG results. Edge-Opt resolves conflicts by performing a heuristic search for a DAG that maximizes the number of satisfied d-separation constraints as implied by the observed data in addition to performing a tie-breaking based on a score criterion. However, Edge-Opt considers all d-separation constraints equally significant which, for limited sample data, does not take into account that marginal and conditional independences are inferred from the data with different confidence levels.

In this paper, we propose a fundamentally different approach than Edge-Opt, where we take advantage of the confidences at which the constraints are inferred to rank v-structures and to resolve conflicts. The proposed approach, in addition to being easy to implement, does not require a heuristic search or any scoring criterion of the global network.

## 6. Methods: Light Mutual Min Algorithm (LMM)

In the proposed algorithm, we attempt to address the above issues related to skeleton recovery using two main techniques. First, candidate edges are ranked using a new measure that combines independence tests when conditioning on all subsets of neighboring variables of the first and second node simultaneously. The proposed ranking criterion is an estimate of the joint conditional posterior probability that the dependence between the corresponding two variables cannot be explained away by subsets of neighboring variables of either the first or the second variable. Second, to ease the multiple testing problem, a new method is presented to relax independence constraints in dense areas of the graph. Moreover, to address the conflicting results issue in orienting edges, we propose a new criterion to rank all candidate v-structures. This ranking offers a method to simultaneously identify v-structures and resolve conflicts. All methods presented are illustrated for the multivariate Gaussian case. However, the same approach can also be extended to other cases with necessary modifications.

### 6.1 Joint Criterion of Conditional Dependence for All Conditioning Sets

In the multivariate Gaussian case, two variables $i$ and $j$ are considered independent when conditioning on the set $Z_{\setminus i,j}$ if and only if they have a zero partial correlation ($\rho_{ij|Z} = 0 \Leftrightarrow i \perp\!\!\!\perp j|Z$). Since the true partial correlation is unknown, a statistical test is typically used to test whether the sample partial correlation $\hat{\rho}_{ij|Z}$ is significant or not (Neapolitan, 2004). In constraint-based learning, an edge $E_{ij}$ is rejected from the graph if and only if at least one independence test is accepted when conditioning on all subsets of the neighbor sets $CP_i$ or $CP_j$ separately. In contrast, in our method, we suggest using a joint dependence criterion that combines independence tests from both sides of an edge simultaneously. To do so, we first follow the approach of Schäfer and Strimmer (2005) to compute an estimate of the posterior probability that the true partial correlation is not zero when given the sample partial correlation $\hat{\rho}_{ij|Z}$.

From Hotelling (1953), when the true partial correlation is zero ($H_0 : \rho = 0$), the sample partial correlation has the following null distribution:

$$f_0(\rho) = (1-\rho^2)^{(\kappa-3)/2} \frac{\Gamma(\kappa/2)}{\pi^{1/2}\Gamma[(\kappa-1)/2]}. \tag{1}$$

In (1), $\Gamma$ is a gamma function with $\kappa$ degrees of freedom, which in this case should be set to $(N - |Z| - 1)$ where $N$ is the number of samples and $|Z|$ is the size of the conditioning set. In contrast, for the alternative hypothesis ($H_A : \rho \neq 0$), $\rho$ can have any value in the range [-1,1], and unless we possess prior information about its distribution, for simplicity, we assume it follows a uniform distribution ($f_A(\rho) = 0.5,\ \forall \rho \in [-1,1]$ and 0 otherwise).

Using the sample partial correlation and the distributions of $\rho$ under both the null and the alternative hypothesis, the posterior probability of the true partial correlation being non-zero can be computed as:

$$P(\rho_{ij|Z} \neq 0 \mid \hat{\rho}_{ij|Z}) = \frac{\pi_A \times f_A(\hat{\rho}_{ij|Z})}{\pi_A \times f_A(\hat{\rho}_{ij|Z}) + \pi_0 \times f_0(\hat{\rho}_{ij|Z})}. \tag{2}$$

In (2), $\pi_0$ and $\pi_A$ are the prior probabilities of the null and the alternative hypothesis respectively where $\pi_0 + \pi_A = 1$. although these priors are generally not known *a priori*, they can be approximated by the user or empirically estimated using likelihood maximization (see Section 6.1.1 for details).

Assuming the faithfulness property, when given the correct neighbor set of a variable $i$ ($CP_i^*$), the dependence of $i$ on another variable $j$, with a true neighbor set ($CP_j^*$) that is neither a child nor a parent of $i$, can sufficiently be explained away by at least one subset of either $CP_i^*$ or $CP_j^*$. Therefore, when learning from limited observation data $D$ and $CP_i$ is our current best estimate of the neighbor set of $i$, an estimate of a one-sided conditional posterior probability that the edge $E_{ij}$ is part of the correct graph can be computed as follows:

$$P(E_{ij}|D)_{[CP_i]} = \min_{Z \subseteq CP_i,\, j \notin Z} P(\rho_{ij|Z} \neq 0 \mid \hat{\rho}_{ij|Z}). \tag{3}$$

Similarly, given the estimated neighbor set of $j$ ($CP_j$), one can also compute the one-sided conditional posterior $P(E_{ij}|D)_{[CP_j]}$. Here, we refer to $P(E_{ij}|D)_{[CP_i]}$ as one-sided because it ignores the information about independence tests when conditioning on subsets of the neighbor set of the other variable. We also call it conditional because we restrict the conditioning tests to subsets of $CP_i$ as if it contains all child and parent variables of $i$.

In order to take advantage of the mutual dependence between a parent and a child variable and the asymptotic property that this dependence cannot be explained away by any set of other variables, in the proposed approach, we rank edges using an estimate of the joint conditional posterior probability that none of the subsets of either of the two neighbor sets can make the two variables conditionally independent as follows:

$$P(E_{ij}|D)_{[CP_i,CP_j]} = P(E_{ij}|D)_{[CP_i]} \times P(E_{ij}|D)_{[CP_j]}. \tag{4}$$

Equation (4) can be interpreted as an estimate of the joint conditional posterior probability that the minimal partial correlations from conditioning on subsets of neighboring variables of $i$ and $j$ separately are both significant. In Section A of the Appendix, ranking edges using Equation (4) is shown to be sufficient for recovering the correct skeleton in the asymptotic limit. For the rest of this paper, we use the acronyms one-sided CPPD and joint CPPD to refer to the quantities measured by Equations (3) and (4) respectively, where CPPD is short for conditional posterior probability of dependence (for all conditioning sets) between the two corresponding variables.

What distinguishes ranking edges using Equation (4) is that it combines the two one-sided CPPDs to make one decision about each edge. In contrast, other constraint-based methods such as the PC and MaxMin algorithms reject the edge if a single subset of either $CP_i$ or $CP_j$ renders $i$ and $j$ conditionally independent. This is equivalent to ranking edges using the minimum of the two one-sided conditional posteriors in the right hand side of Equation (4) and ignoring how larger than the minimum the other one-sided conditional posterior was in comparison. While either the minimum or the product can work perfectly when given a very large number of samples, combining two sources of information about the edge as in Equation (4) is anticipated to improve the estimation of the posterior that the edge is part of the correct graph in limited sample problems. In this work, we empirically compare the two ranking criterion and show that the proposed criterion provides a consistent and significant improvement in the accuracy of skeleton recovery.

However, we should note that, for simplification, the factorization in the right hand side in (4) does ignore a possible correlation between the minimal partial correlations from both sides of the edge and it also ignores the possibility that $CP_i$ and $CP_j$ might be overlapping. In Section B of the Appendix, we elaborate on the effect of overlapping neighbor sets on ranking edges and use examples to illustrate why the overlap of neighbor sets is not likely to have a significant negative effect on the accuracy of ranking edges.

Though all methods presented in this paper are restricted to the multivariate normal case, the proposed ranking criterion (Equation 4) can be extended to incorporate other types of independence testing statistics. For example, the method proposed by Margaritis and Thrun (2001) for computing a posterior of conditional independence for the non-linear case, based on multi-resolution discretization, can be used in Equation (2) to compute the posterior of conditional dependence. Other conditional independence testing methods (i.e., G2 statistic, Neapolitan, 2004) can also be used by incorporating the correct null and alternative distributions. These extensions, however, require further research and empirical analysis that goes beyond the scope of the current work.

#### 6.1.1 ESTIMATING PRIORS

Although the priors $\pi_0$ and $\pi_A$ in Equation (2) are not known *a priori*, one can anticipate that BN learning is typically applied to sparse networks ($\pi_0 >> \pi_A$). In addition, likelihood maximization can be used to empirically estimate these priors. For example, in the case of zero order partial correlations (conditioning on the empty set: $\emptyset$), $\pi_A$ can be selected to maximize the log of the likelihood of the sample partial correlations generated from a mixture of the null and the alternative distributions as follows:

$$\hat{\pi}_A = arg \max_{0<\pi_A<1} log \prod_{i,j} [\pi_A \times f_A(\hat{\rho}_{ij}) + (1-\pi_A) \times f_0(\hat{\rho}_{ij})]. \tag{5}$$

Although $\pi_A$ and $\pi_0$ are expected to vary between the zero order and higher order partial correlations, the $\hat{\pi}_A$ estimated in (5) can be considered as an upper bound for the alternative prior for the purpose of computing Equation (2). This is because as we take the least significant of all partial correlations when conditioning on more than one set, including the empty set, dependence becomes more likely to be explained away. Therefore, in all experiments reported in this paper, as a heuristic and for simplicity, using the $\hat{\pi}_A$ estimated in (5), we set $\pi_A$ and $\pi_0$ in Equation (2) to $\hat{\pi}_A/2$ and $(1-\hat{\pi}_A/2)$ respectively for all partial correlations. To maximize (5), we used a line search where $\pi_A$ was restricted to [0.001, 0.2].

### 6.2 Adaptive Reduction of Independence Testing

While in theory, a locally adaptive threshold can resolve the increased type II error problem described in Section 5.2, estimating an accurately variable threshold is a nontrivial problem. As an alternative, we employ a heuristic procedure to relax independence testing across the network, such that dense regions in the graph get the most relaxation.

When inferring a skeleton from small sample data, pairs of nodes are expected to show asymmetric one-sided CPPD (Equation 3). This is because every node in the graph has its own neighbor set, and parameter estimation from small sample data is not perfect. In addition, due to the multiple testing problem, nodes with many recognized neighbors will tend to show smaller CPPD with other variables. The proposed approach takes advantage of this asymmetry to identify which node of every pair to be connected might be suffering from multiple testing. Afterward, candidate edges connecting to the identified node become candidates for relaxation of constraints. This relaxation is performed whenever a new edge $E_{ij}$ is added to the skeleton by selectively updating the neighbor sets of $i$ and $j$ using the following rules:

1. Both neighbor sets of $i$ and $j$ are updated ($j \in CP_i$, $i \in CP_j$) if the one-sided CPPD between $i$ and $j$ is symmetric or almost symmetric ($|P(E_{ij}|D)_{[CP_i]} - P(E_{ij}|D)_{[CP_j]}| < \omega$, for small and constant $\omega \in (0,1]$).

2. Only the neighbor set of $i$ is updated ($j \in CP_i$, $i \notin CP_j$) if $i$ has the higher one-sided CPPD ($P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]} + \omega$).

3. Only the neighbor set of $j$ is updated ($j \notin CP_i$, $i \in CP_j$) if $j$ has the higher one-sided CPPD ($P(E_{ij}|D)_{[CP_j]} > P(E_{ij}|D)_{[CP_i]} + \omega$).

In this paper, when a node $i$ is added to the neighbor set of a node $j$, $j$ is said to be aware of the connection $E_{ij}$. The process of occasionally making only one of a connected pair of nodes aware of a recovered edge will result in a reduction in the number of tests used to evaluate edges connecting to the other node.

In the given rules, $\omega$ is a threshold to decide whether dependence is significantly asymmetric between two nodes. In the limited sample case, when $\omega = 1$, the selective reduction of independence testing will not be performed, whereas when $\omega$ is set to a small positive value ($\omega \in (0,1)$), the reduction of independence testing will be applied whenever the difference of the one-sided CPPD (Equation 3) is greater than $\omega$. Though any value of $\omega$ in the range $(0,1]$ can be used, in this paper, we limit the comparison to two extreme cases ($\omega = 1$ or $10^{-6}$). Also, in the Appendix, we provide a proof that the proposed algorithm recovers the correct skeleton in the asymptotic limit for all $\omega > 0$.

Note that our approach relies mainly on selective reduction of independence testing to mitigate the effect of multiple testing. The proposed approach can also be integrated with other heuristics to control for multiple testing. For example, the false discovery rate control methods proposed by Tsamardinos and Brown (2008) or Li and Wang (2009) can potentially be integrated in Equation (3) for even further inference accuracy. This integration however is beyond the scope of the current work.

#### 6.2.1 MOTIVATION FOR THE SELECTIVE REDUCTION OF INDEPENDENCE TESTING

In the limited sample case, the proposed method of selectively updating neighbor sets serves as an adaptive reduction/relaxation of the independence testing where highly connected nodes in the graph get the most relaxation due to their rapidly growing independence from other variables as a result of the increased type II error. As a consequence of this relaxation, the set of neighbors used in conditional independence testing remains small and the multiple testing issue is less likely to contribute to the rejection of true edges. Furthermore, the smaller neighbor sets lead to a dramatic decrease in the number of conditional independence tests and thus much faster learning.

Although making a node not aware of some of its neighbors might enable it to accept a new edge without taking into consideration all its neighbors, leading to a possible false positive identification, this is not expected to be a common behavior due to the following reasons:

1. Due to ranking edges by the product of the one-sided CPPDs from both sides of an edge, for an edge to be incorrectly selected by the algorithm, the dependence has to appear incorrectly significant from both sides simultaneously.

2. Assigning information about edges to the nodes that show higher one-sided CPPD serves to assign neighbor sets of minimal size in a way that provides a maximal mutual information

with the nodes to which they are assigned. For example, if in a new edge $E_{ij}$, $i$ shows higher one-sided CPPD than $j$ ($P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]} + \omega$), this means the set $CP_j$ contains variables that partially contain the information that node $i$ provides about $j$. Therefore, by exclusively assigning the information about the edge to node $i$, less information is lost than if the information about the edge was exclusively assigned to node $j$. Also, since node $j$ has low one-sided CPPD on $i$, it is less likely that node $i$ is needed to block the dependence of $j$ on non-true neighbors.

In addition, as the number of training samples increases, fewer pairs of variables will show asymmetric dependence leading to less relaxation of independence testing. As a result, in the case of a large number of training samples, every node will become aware of all of its neighbors, and hence the algorithm still retains its correctness in recovering the true skeleton in the asymptotic limit (see Appendix for proof). Note that, this approach is heuristic and the algorithm behavior will vary depending on the number of observational samples and the local density of edges in the graph. Section 6.3.1 gives an example that illustrates the behavior of the proposed relaxation of independence testing during skeleton recovery. Moreover, the second part of the supplementary material gives further illustrations with further discussion of the motivation for the proposed approach.

### 6.3 LMM Algorithm for Skeleton Recovery

The simplified version of LMM (Algorithm 1) has two major phases: forward selection and backward elimination. In the forward selection, the algorithm incrementally adds new edges with the highest joint CPPD (Equation 4) until a certain number of edges are added or a threshold is reached. In the backward elimination phase, the algorithm incrementally eliminates edges with the least joint CPPD. As edges are added to the skeleton, sets of neighbors are updated using the rules stated in Section 6.2.

As the forward selection phase of LMM proceeds, the joint CPPD (Equation 4) along a previously recovered edge $E_{ij}$ might become less significant as more edges are added due to the recovery of new neighbors, such as common parent variables that explain the observed dependence between $i$ and $j$. Therefore, to improve the efficiency of the algorithm (Aliferis et al., 2010), the recovered edge $E_{ij}$ with the least joint CPPD becomes a candidate for early deletion. Also, as the network grows in the first phase, the node with the higher one-sided CPPD among a pair of connected nodes might change. For example, when an edge $E_{ij}$ is first added, $i$ has a higher one-sided CPPD on $j$ and hence only the neighbor set of $i$ is updated to include $j$. However, as more edges are added that connect to $i$, $j$ might become the node with the higher one-sided CPPD and hence, $j$ should be made aware of the edge ($CP_j \leftarrow CP_j \bigcup i$). To address these issues, every three iterations of forward selection, we perform the following three operations:

1. OPT1: Delete the edge $E_{ij}$ with the least joint CPPD if it is less than the joint CPPD of any yet unrecovered edge.

2. OPT2: Set ($CP_j \leftarrow CP_j \cup i$) if $E_{ij}$ is the recovered edge with the highest joint CPPD where $j$ is more or almost equally dependent on $i$ ($P(E_{ij}|D)_{[CP_j]} > P(E_{ij}|D)_{[CP_i]} - \omega$) but it is not aware of the edge ($i \notin CP_j$).

3. OPT3: Set ($CP_j \leftarrow CP_j - i$) if $E_{ij}$ is the recovered edge with the least joint CPPD where $j$ is less dependent on $i$ ($P(E_{ij}|D)_{[CP_j]} < P(E_{ij}|D)_{[CP_i]} - \omega$) but it is aware of the edge ($i \in CP_j$).

**Algorithm 1** : LMM_Simplified(D, $\omega$, $\gamma$, [$MinSize$], [$MaxSize$])

**Input:** Data D, Relaxation Parameter $\omega \in (0,1]$, Joint CPPD Threshold $\gamma \in (0,1]$
**Input:** [Optional]: Maximum # of Edges to Add in Forward Selection: $MaxSize$
**Input:** [Optional]: Minimum # of Edges to Leave in Backward Elimination: $MinSize$
**Output:** Skeleton: $\hat{G}_S$

$\hat{G}_S \leftarrow \emptyset$ //$\emptyset$ is the empty set
$CP_i \leftarrow \emptyset, \forall i$
**%Phase I:Forward Selection**
**repeat**
  $E_{ij} \leftarrow arg \max_{E_{xy} \notin \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]}$
  $addEdge(i, j)$
**until** $|\hat{G}_S| = MaxSize$ or $\max_{E_{xy} \notin \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]} < \gamma$
**%Phase II:Backward Elimination**
**repeat**
  $E_{ij} \leftarrow arg \min_{E_{xy} \in \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]}$
  $deleteEdge(i, j)$
**until** $|\hat{G}_S| = MinSize$ or $\min_{E_{xy} \in \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]} > \gamma$
**return** $\hat{G}_S$

**Procedure addEdge(i, j)**
$\hat{G}_S \leftarrow \hat{G}_S \cup E_{ij}$
**if** $P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]} - \omega$ **then**
  $CP_i \leftarrow CP_i \cup j$
**end if**
**if** $P(E_{ij}|D)_{[CP_j]} > P(E_{ij}|D)_{[CP_i]} - \omega$ **then**
  $CP_j \leftarrow CP_j \cup i$
**end if**
**End Procedure**

**Procedure deleteEdge(i, j)**
$\hat{G}_S \leftarrow \hat{G}_S - E_{ij}$
$CP_i \leftarrow CP_i - j$
$CP_j \leftarrow CP_j - i$
**End Procedure**

Note that the OPT1 operation tries to ensure that at every stage of the forward selection, the recovered edges are the edges that thus far showed the most significant mutual dependence that has not yet been explained away by any of the recovered relations. On the other hand, OPT2 and OPT3 serve to ensure that the selective reduction of independence testing complies with the rules stated in Section 6.2. The selection of the edge with the highest joint CPPD in OPT2 and the edge with the least joint CPPD in OPT3 is consistent with creating neighbor sets of maximum mutual information

with the nodes to which they are assigned. Algorithm 2 details the changes to the forward selection of the simplified LMM (Algorithm 1: Lines 5-6).

**Algorithm 2** : LMM(D, ω, γ)

**% Update Algorithm 1 by adding the following lines between Lines 5 and 6.**
**5: ....**
**if** iterationNumber is multiple of 3 **then**
**% OPT1**
$E_{ij} \leftarrow arg \min_{E_{xy} \in \hat{G}_S} P(E_{xy} \mid D)_{[CP_x, CP_y]}$
**if** $P(E_{ij} \mid D)_{[CP_i, CP_j]} < \max_{E_{xy} \notin \hat{G}_S} P(E_{xy} \mid D)_{[CP_x, CP_y]}$ **then**
deleteEdge$(i, j)$
**end if**
**% OPT2**
$E_Q \leftarrow \{E_{ij} : E_{ij} \in \hat{G}_S, i \notin CP_j, j \in CP_i, P(E_{ij} \mid D)_{[CP_j]} > P(E_{ij} \mid D)_{[CP_i]} - \omega\}$
**if** $E_Q \neq \emptyset$ **then**
$E_{ij} \leftarrow arg \max_{E_{xy} \in E_Q} P(E_{xy} \mid D)_{[CP_y]}$
$CP_j \leftarrow CP_j \cup i$
**end if**
**% OPT3**
$E_Q \leftarrow \{E_{ij} : E_{ij} \in \hat{G}_S, i \in CP_j, j \in CP_i, P(E_{ij} \mid D)_{[CP_j]} < P(E_{ij} \mid D)_{[CP_i]} - \omega\}$
**if** $E_Q \neq \emptyset$ **then**
$E_{ij} \leftarrow arg \min_{E_{xy} \in E_Q} P(E_{xy} \mid D)_{[CP_y]}$
$CP_j \leftarrow CP_j - i$
**end if**
**end if**
**6: ....**

Note, LMM algorithm is presented here in high level language. Details on how we implemented and optimized LMM are presented in the supplementary material and also can be seen in the source code which we are making available with this publication. In the case of applying OPT1-3, to avoid closed loops of adding and deleting the same edges or updating the same sets of neighbors, in our implementation, edges added in the last ten iterations are not considered for either of OPT1, OPT2, or OPT3 operations.

### 6.3.1 SIMULATED EXAMPLE

To illustrate the behavior of LMM, we simulated the network in Figure 1, and generated 1,000 samples as described in Section 7.1 below. We used a relatively large number of observations to ease the replication of our results and to test whether or how well LMM can recover the correct network given the proposed method of reducing independence testing when given a large set of samples.

Figure 2 (a) shows a network of 13 edges recovered by LMM forward selection. A solid connection point (circle) indicates that the node is aware of the connection. For example, in Figure 2 (a), node $F$ is connected to all of $H$, $K$, and $J$, but it is only aware of its neighbor $J$: $CP_F = \{J\}$. Such

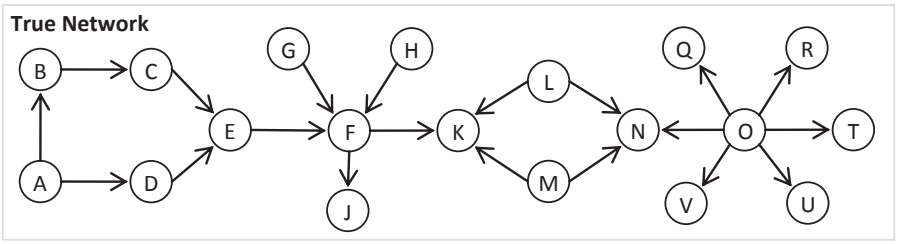



Figure 1: A simulated network of 19 nodes and 20 edges

a result is expected, since once a node starts conditioning on child variables it can develop smaller partial correlations with its parent variables (see Supplement). On the other hand, node $J$ has only one parent and therefore, it will always be aware of it as a neighbor and hence its dependence on grandparent variables (nodes $E$, $G$, and $H$) or sibling variables (node $K$) will always be successfully blocked.

Figure 2 (b) shows a network of 25 edges recovered by continuing the forward selection for the same network and the same run of LMM. False edges are represented by red dotted lines. In this case, although the resulting connectivity information is not complete (distributed), it is sufficient to block candidate false edges. For example, all child variables of node $O$ are aware of $O$ as a neighbor and therefore, their mutual dependence is blocked (e.g., $P(E_{QR}|D)_{[CP_Q]} \cong 0$). On the other hand, node $O$ is only aware of three of its child variables and that does not have any effect of increasing the chances of recovering false edges among its child variables. Also, note that all the false edges recovered in Figure 2 (b) were only selected due to forcing LMM to recover more than 20 edges and the dependence along these edges is already blocked (e.g., $P(E_{JG}|D)_{[CP_J]} \cong 0$) and they will be eliminated in the backward elimination phase as shown in Figure 2 (c).

The sub-network of node $O$ and its child variables in Figure 2 is a good example of why LMM can outperform other algorithms such as the PC or MaxMin in limited sample problems. For example, in PC algorithm, to correctly recover the edge $E_{ON}$, all independence tests of conditioning on all subsets of the other 5 neighbors must be rejected while in LMM, node $O$ was aware of three neighbors only and therefore the chances of rejecting the edge due to multiple testing was reduced without increasing the risk of false positive edges among the child variables of $O$. However, this is still a heuristic approach, and it is possible that in more complex examples the dependence along a false edge is only partially blocked (see the supplementary material for more examples and illustrations).

### 6.3.2 RANKING OF SKELETON EDGES

The LMM algorithm for skeleton recovery as given in Algorithm 1 can be used to recover skeletons in any of the following three approaches:

1. By using a threshold value ($\gamma \in (0,1]$) for the joint CPPD to decide which edges to be accepted or rejected.

2. By specifying the number of edges to be added in the forward selection and the number of edges to remain after backward elimination ($MaxSize > MinSize > 0$, and $\gamma = 0$).

3. By allowing the algorithm to add many edges in the forward selection (large $MaxSize$, and $\gamma = 0$) and letting the backward elimination eliminate all of them one at a time ($MinSize = 0$).

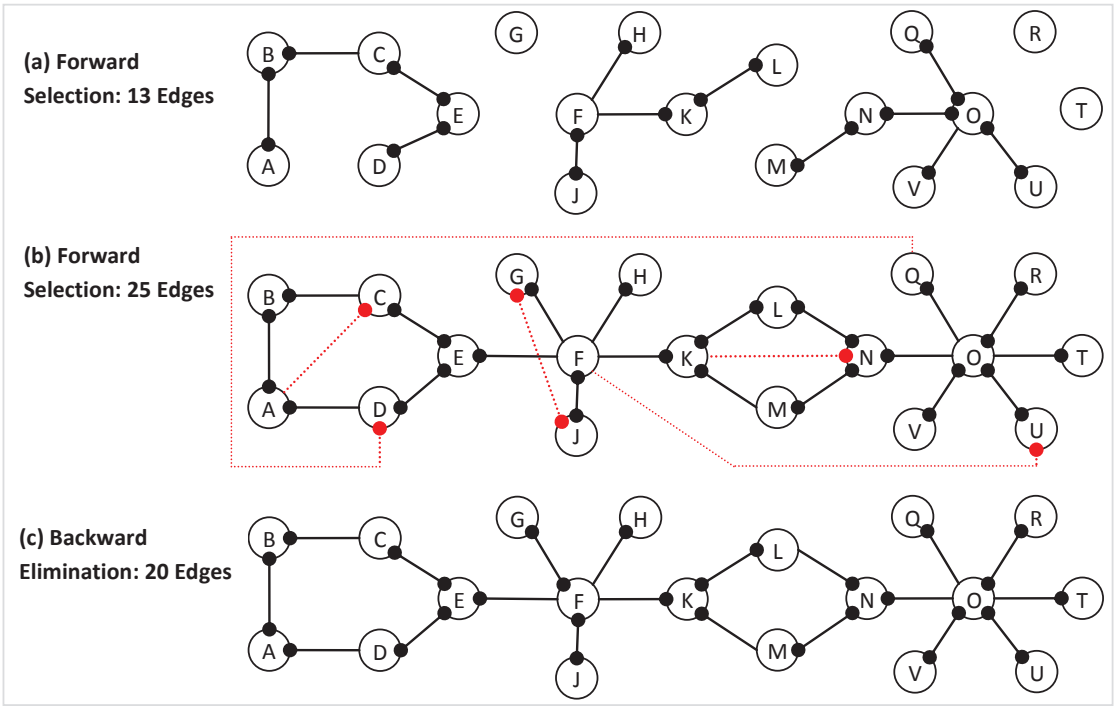



Figure 2: LMM output during forward selection and backward elimination of the same algorithm run at iterations where a) 13 edges have been added in forward selection, b) 25 edges have been added in forward selection, and c) 20 edges have remained in the skeleton after backward elimination has started. Red dotted lines represent false edges. A solid connection point (filled circle) at the end of an edge indicates the node is aware of its neighbor at the other side of the edge.

The order at which edges are deleted in the backward elimination is then used to rank edges where the last deleted edges become the most significant.

Using a threshold for the joint CPPD (Approach 1) makes running LMM similar to running the PC and MaxMin algorithms where a threshold for independence testing is also needed. However, providing the optimal value for any of these thresholds is not trivial for users in limited sample problems. Running LMM by specifying the number of edges to be added and deleted (Approach 2) can be an ideal alternative in cases where the users have a rough estimate of the correct number of edges. For example, if the user thinks the correct network contains about a 1,000 edges, the user can set *MinSize* to 1,000 and *MaxSize* to a slightly larger number (e.g., 1,200). Nonetheless, using LMM to provide a rank of edges (Approach 3) instead of recovering a single skeleton can be more useful, in that, it relieves users from providing a threshold or guessing the number of edges ahead of time. In addition, the rank of edges has a simple interpretation, where the top edges have the highest confidence of being correct while the later edges have less confidence of being correct. The justification for this ranking procedure is that at every iteration of the backward elimination,

the edge with the least joint CPPD is deleted, and thus, the remaining edges are considered to have higher evidence of direct dependence (higher joint CPPD) and are thus more likely to be correct.

To our knowledge, this flexibility property of this last approach has no match in the algorithmic procedure of either the PC or the MaxMin algorithms where one needs to re-run the algorithm many times using different thresholds for independence testing to be able to rank edges. In addition, our approach can be used as a practical alternative to re-sampling and model averaging methods that are typically used to rank edges (Neapolitan, 2004). However, we should note that, ranking skeleton edges is not new to BN learning. For example, Tsamardinos and Brown (2008) ranked edges based on their p-values to control for false discovery rate in skeleton recovery. A similar approach for ranking edges was also proposed by Armen (2011).

In all experiments reported, the skeletons produced by LMM were generated using the proposed ranking procedure (Approach 3). The only exception of this configuration was the experiments of computational complexity comparison where we used a threshold parameter (Approach 1: $\gamma \in (0,1]$) for the joint CPPD in order to make the computational complexity comparison as fair as possible. Also, note that, one can also use a hybrid approach by using a non-zero $\gamma$ in Approaches 2 and 3 above. This later case, however, is not used in any of the presented empirical results.

### 6.4 Complexity Analysis of Skeleton Recovery

The computational complexity of constraint-based methods is a measure of the number of independence tests needed to recover the skeleton. However, approximating the exact number of tests needed to execute LMM or other algorithms is a non-trivial problem. Alternatively and similar to other authors (Kalisch and Bühlmann, 2007; Tsamardinos et al., 2006), we use the worst case as an upper bound on the expected complexity.

In the worst case, the three algorithms, PC, MaxMin, and LMM, have a computational complexity of $O(p^2 \times 2^q)$ where $p$ is the number of nodes and $q$ is the maximum size of all neighbor sets in the recovered network ($q = \max_i |CP_i|$). This is because the worst case assumes all nodes have the same maximum connectivity, and therefore we need to perform $O(p^2 \times 2^q)$ independence tests to evaluate candidate edges. However, when using LMM with limited sample data, the proposed adaptive reduction method of independence testing discussed in Section 6.2 (Algorithm 1: Lines 12-18) forces the algorithm to avoid using some neighbors in conditional independence tests. Therefore, although the node with the maximum connectivity is connected to $q$ edges in the true DAG, during the skeleton recovery, it will only be aware of a fraction of them, leading to a reduced number of independence tests (i.e., $O(p^2 \times 2^{q/\delta}) \, s.t. \, \delta \geq 1$). Unfortunately, estimating this fraction before running the algorithm is not possible. Therefore, in this paper, we mainly rely on the empirical evaluation to compare the time complexity of LMM to the other two algorithms.

One of the popular techniques to speed constraint-based learning is by limiting the size of the conditioning sets. This method can reduce computational complexity and it can be used with the PC, MaxMin or LMM algorithms. In the supplementary material, we elaborate on how we implemented LMM and on the optimization techniques we used to avoid redundant computations while making the search for the next edge to add or delete computationally efficient.

### 6.5 Extending the Skeleton to the Equivalence Class

Learning a directed graph from observational data is typically restricted to learning the completed partially directed acyclic graph (CPDAG). This restriction is motivated by the equivalence nature

of some Bayesian networks, in that, multiple networks with partially different orientations of edges can produce the same observational data. As a consequence, learning the complete directed graph from observational data alone can be impossible in many cases (Meek, 1995; Spirtes et al., 2001).

Once a skeleton is recovered, different approaches can be used to extend it to a CPDAG. Examples of these methods are the score-based constrained hill climbing search algorithm (Tsamardinos et al., 2006) and the constraint-based approaches such as the PC algorithm (Meek, 1995; Spirtes et al., 2001). We stress that our contribution in recovering the skeleton can be used with any other algorithm to recover the CPDAG. However, in this work, we focus on constraint-based learning due to its easy implementation and its computational scalability. In addition, we improve upon the classical constraint-based inference by presenting a new method for resolving conflicting v-structures.

Recovering CPDAGs using constraint-based inference relies on the recovery of the v-structures (see definitions) which are proven to be common among all DAGs of the same equivalence class. From properties of v-structure relations (see Lemma 13 below), a common child variable $w$ in a v-structure relation ($x \to w \leftarrow y$) induces a conditional dependence between the parent variables $x$ and $y$. Moreover, since $x$ and $y$ are not adjacent in the skeleton, $w$ cannot belong to any set $Z_{\setminus x,y}$ that makes $x$ and $y$ conditionally independent. This property is the basic idea in constraint-based methods (Spirtes et al., 2001). However, this approach can produce inconsistent results. For example, in a case of limited training samples, it is possible to recover multiple conflicting v-structures.

To improve the accuracy of constraint-based inference of CPDAG, we present a new approach for ranking v-structures that can be used to resolve conflicts. The proposed method is derived from the following Lemma (see Neapolitan, 2004, Chapter 2):

**Lemma 13** *Suppose we have a DAG $G = (V,E)$ and an uncoupled meeting $x - w - y$ . Then the following are equivalent:*

1. *$x - w - y$ is a head-to-head meeting: $x \to w \leftarrow y$.*
2. *There exists a set not containing $w$ that d-separates $x$ and $y$.*
3. *All sets containing $w$ do not d-separate $x$ and $y$.*

Based on Lemma 13 and using the partial correlation method, when $x \to w \leftarrow y$ is a correct v-structure and we have data with a large sample size $n$, then both of the following statements are correct (see Appendix for proof):

1. $\lim_{n\to\infty} \left[ \min_{Z \subseteq V_{\setminus x,y,w}} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \right] = 0.$
2. $\lim_{n\to\infty} \left[ \min_{Z \subseteq V_{\setminus x,y}, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \right] = 1.$

Although, Lemma 13 was shown to be accurate in true DAGs (Neapolitan, 2004, Chapter 2), it might not always hold when learning DAGs in real applications due to the unguaranteed accuracy of independence testing methods based on small sample data. As a result, it is possible to recover wrong or conflicting results (e.g., conflicting v-structures: $x \to w \leftarrow y$ and $w \to x \leftarrow u$).

In order to identify v-structures in a given skeleton and to resolve conflicts, in the proposed approach, we rank all candidate v-structures based on the confidence that the candidate head of

the v-structure is a common child of the other two variables. To compute this confidence criterion for every candidate v-structure ($x \rightarrow w \leftarrow y$), we compute the joint CPPD (Equation 4) between $x$ and $y$ when $w$ is added to all conditioning sets and the joint CPPD when $w$ is removed from all the conditioning sets. The difference between the two joint CPPDs is then used as a measure of the level of the induced dependence between $x$ and $y$ caused by $w$. If the joint CPPD is found to increase when $w$ is added to all conditioning sets, $w$ is concluded to be a common child of both $x$ and $y$. Also, in a case where two conflicting v-structures get accepted, we take advantage of the amount of increase in the joint CPPD to perform tie-breaking and ignore the v-structure with the smaller induced dependence. The justification for this approach follows directly from Lemma 13, in that, $w$ induces dependence between $x$ and $y$ if and only if it is their common child, and in small sample problems, it becomes intuitive to use the increase in the joint CPPD when $w$ is added to all conditioning sets as a measure of the confidence that $w$ is a common child.

Algorithm 3 presents a summary of the proposed constraint-based orientation of a given skeleton. Note that, testing for induced dependence from common neighbors does not have the multiple testing problem. This is because the number of conditioning sets that contain the common neighbor is equal to the number of conditioning sets that do not contain the common neighbor. Therefore, in the orientation algorithm, we use the complete set of neighbors found in the skeleton. To distinguish these types of sets, we use bar notation on top of the set name (i.e., $\overline{CP_x}$).

The set of additional orientation rules in Algorithm 3 are also used in other constraint based learning methods of the CPDAG such as the PC and TPDA algorithms. Meek (1995) provides a detailed discussion of these rules and their correctness for finding the CPDAG.

As mentioned earlier (Section 5.3), to our knowledge, there is only one other algorithm called Edge-Opt (Fast, 2010) that attempts to resolve conflicts in the constraint-based orientation of BN edges. Our approach is fundamentally different from Edge-Opt, in that, Edge-Opt considers all constraints to be equally significant, while our approach takes advantage of the differences in confidence at which the constraints are inferred and uses these differences to rank candidate v-structures. Moreover, in addition to being easy to implement, our approach does not require any heuristic search or computing a score criterion of the global network to perform tie-breaking. However, we should note that our approach does not solve conflicting edges resulting from the additional orientation rules (R1-4). In our implementation, whenever a conflict is found along a certain edge, it is left undirected.

In the experimental validation, we were not able to empirically compare our approach to Edge-Opt because its publicly available implementation does not support the multivariate Gaussian case. Nonetheless, we compare our implementation to a variant of itself where a random tie-break is used to resolve conflicting v-structures and show that the proposed tie-breaking method provides a significant and consistent improvement in the accuracy of CPDAG orientation. We also compare the proposed approach to the PC and MaxMin hill climbing algorithms. Moreover, in the Appendix, we provide a complete proof that the proposed approach recovers the correct CPDAG in the asymptotic limit.

## 7. Experiments and Comparison

For comparisons, we restrict our experiments to multivariate Gaussian simulated data and the partial correlation method is used for independence testing (or dependence measurement) in all algorithms. All PC algorithm results reported are performed using the publicly available PCAlg tool (Kalisch

**Algorithm 3** : LMM_EO($G_S$, $D$)

**Input:** Skeleton: $G_S$, Data: $D$

$Q \leftarrow \emptyset$ //$\emptyset$ is the empty set
**for all** $\langle x, w, y \rangle$ where $E_{xy} \notin G_S$ and $E_{wy}, E_{wx} \in G_S$ **do**
  $CD_{w+}(x,w,y) \leftarrow \min_{Z \subseteq \overline{CP_x}, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \times \min_{Z \subseteq \overline{CP_y}, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$
  $CD_{w-}(x,w,y) \leftarrow \min_{Z \subseteq \overline{CP_x}, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \times \min_{Z \subseteq \overline{CP_y}, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$
  $CD(x,w,y) = CD_{w+}(x,w,y) - CD_{w-}(x,w,y)$
  **if** $CD(x,w,y) > 0$ **then**
    $Q \leftarrow Q \cup < x, w, y >$
  **end if**
**end for**
**while** $Q$ is not empty **do**
  $< x, w, y > \leftarrow arg \max_{<j,k,l> \in Q} CD(j,k,l)$
  $Q \leftarrow Q - < x, w, y >$
  **if** the edges $x - w$ and $y - w$ are not oriented **then**
    Orient $x - w$ into $x \rightarrow w$
    Orient $y - w$ into $y \rightarrow w$
  **end if**
**end while**

**% Apply the Additional Orientation Rules as Follows:**

**repeat**
  Orient $i - j$ into $i \rightarrow j$ whenever any of the following is correct:
  R1: There exists an arrow $k \rightarrow i$ *s.t.* $k$ and $j$ are not adjacent.
  R2: There exists a directed path from $i$ to $j$ (i.e., $i \rightarrow \{\} \rightarrow j$).
  R3: There exist two chains $i - k \rightarrow j$ and $i - l \rightarrow j$.
  R4: There exist two chains $i - k \rightarrow l$ and $k \rightarrow l \rightarrow j$ *s.t.* $k$ and $l$ are adjacent.
**until** no more orientations are found

et al., 2010). The original MaxMin algorithm tool (Aliferis et al., 2003) only supports the discrete data case. We therefore re-implemented the algorithm to use the partial correlation test to recover the skeleton. For the CPDAG recovery, we used the publicly available toolkit BNlearn (Scutari, 2010) which implements a variant of MaxMin hill climbing that supports the BIC score criterion for continuous data, which we used for all MMHC reported experiments. Also, in all experiments, the hill climbing search was performed with a Tabu search with a list of 100 possible solutions which is the same setting used by the authors of the algorithm (Tsamardinos et al., 2006).

### 7.1 Simulating Data

For every experiment, the true model is randomly generated (Kalisch and Bühlmann, 2007) as follows:

1. Fix an ordering of variables.

2. Fill the adjacency matrix $A$ with zeros.

3. Randomly fill entries in the lower triangle matrix $A$ with ones by independent realizations of Bernoulli random variables with a success probability $s$ where $0 < s < 1$ ($s$ represents the level of sparseness of the network)

4. Replace each entry with a 1 in the adjacency matrix by independent realizations of a uniform random variable in the range $[-1,-0.1] \cup [0.1,1]$.

These steps will result in an adjacency matrix $A$ whose entries are either zero or in the range [-1,-0.1] or [0.1,1]. Afterward, in the corresponding DAG, if $A_{ij}$ is not zero, then node $j$ is a parent of node $i$ with a coefficient $A_{ij}$. Using this randomization setting, in the case of $p$ variables, the expected number of neighbors $CP_i^*$ for a node $i$ can be estimated as: $E(|CP_i^*|) = s \times (p-1)$, while the expected number of all edges can be estimated as: $E(|G|) = s \times (p-1) \times p/2$. Therefore, the density of the graph is linearly proportional to $s$. For the rest of this paper, when the simulated networks are said to contain $X$ edges, this means $s$ was selected so that the expected number of edges was $X$. However, the exact number of edges in each network will vary slightly due to randomization.

In our empirical analysis, we have generated a single set of observational samples from every simulated network. Once the adjacency matrix of the simulated network is fixed, every observational sample is recursively generated as follows:

$$X^{(1)} = \varepsilon^{(1)} \sim N(0, \sigma_1^2).$$

$$\forall i = 2,3...p : X^{(i)} = \varepsilon^{(i)} + \sum_{j=1}^{i-1} A_{ij} \times X^{(j)} \ , s.t.\ \varepsilon^{(i)} \sim N(0, \sigma_i^2).$$

where $\varepsilon^{(1)}, \varepsilon^{(2)}, ....\varepsilon^{(p)}$ are independent random normal variables representing the marginal noise. In all reported experiments, the variances of these noise variables are randomly sampled from an inverse gamma distribution: $\sigma_i^2 \sim InvGamma(\alpha = 2, \beta = 1)$, $\forall i$.

### 7.2 Results and Discussion

To evaluate the proposed algorithm we performed multiple experiments with various settings in an effort to cover a wide variety of possible cases. In every case, we simulated multiple networks and carried out a semi-exhaustive evaluation such that, all methods are compared at different levels of recovery where they are used to recover many solutions of varying sizes and the comparison is illustrated at each level. To finish all the results presented, we used 5 workstations with 8 cores each with a total computation time of about 4 weeks. Moreover, for computational reasons and due to the low statistical power when learning from limited sample data, the individual conditioning sets were restricted not to contain more than four variables for all methods. The only exception was the performance evaluation experiment based on large sample data where we let the conditioning sets contain up to six variables.

For comparison, the receiver operating characteristics curve (ROC) is used to evaluate the accuracy of skeleton recovery while the true positive rate (TPR) plots (TPR against number of retrieved edges) are used to evaluate the accuracy of CPDAG inference. Also, in Section C of the Appendix, we provide additional illustrations of CPDAG inference comparisons using the structural Hamming

distance (SHD) metric. Every method was used to recover multiple skeletons or CPDAGs of varying sizes for the same network and the evaluation plots (ROC, TPR, and SHD curves) of every method were then generated for every single network. For compact presentation, we only present the average of these evaluation curves for every set of networks in addition to bar plots of specified cases to demonstrate the variance of inference accuracy among all networks of the same set. In the case of PC and MaxMin, for every network, each algorithm was run many times with different thresholds ($\alpha = 10^{-30}$, $10^{-10}$, $10^{-6}$, $10^{-4}$, $10^{-3}$, 0.003, 0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.5). We also, in some cases, used additional alpha values greater than 0.5 to recover larger numbers of edges. In contrast, for all experiments except for computational complexity, LMM was run only once for every network using Approach 3 described in Section 6.3.2. In the forward selection, LMM was set to add as many edges as 2.5 times the number of nodes ($MaxSize = 2.5 \times numberofnodes$) and the order at which edges were removed in the backward elimination was used to rank all selected edges ($MinSize = 0$). The choice of 2.5 was mainly to ensure that the number of edges added in the forward selection is large enough to work well for most cases.

### 7.2.1 SKELETON RECOVERY

In the first set of experiments, we evaluate LMM for recovering skeletons using 900 networks divided into 9 groups simulated as follows:

1. Fixed number of nodes $p = 200$.
2. Three levels of connectivity density: $\sim$100, $\sim$200, and $\sim$400 edges.
3. Three levels of number of observations: 30, 100, and 300 samples.
4. For every configuration (same number of edges and same number of observations), a set of 100 different networks were generated with separate observational data for each.

First, to assess the effect of the new joint dependence criterion and the proposed adaptive reduction of independence testing, we compared LMM to two variants of itself LMM-1 and LMM-2. In all experiments presented, LMM was configured to adaptively reduce the independence testing as described in Section 6.2 by setting $\omega$ to very small value ($\omega = 10^{-6}$). In contrast, LMM-1 and LMM-2 were configured not to use the adaptive reduction of independence testing by setting $\omega$ to 1. Also, LMM-1 was set to use the proposed joint CPPD criterion (Equation 4) while LMM-2 was set to use the minimum of the one-sided CPPDs from both sides ($\hat{P}(E_{ij} \mid D)_{[CP_i,CP_j]} = min(P(E_{ij} \mid D)_{[CP_i]}, P(E_{ij} \mid D)_{[CP_j]})$) as a mutual dependence criterion which made it the most similar to the PC and MaxMin algorithms in assessing edges.

Figure 3, shows an averaged plot of the false positive rate (FPR) and the true positive rate (TPR) of the three algorithms in recovering the skeletons of four sets of the simulated networks where the number of edges is $\sim$100 and $\sim$400 while the number of observations is 30 and 100 samples. In addition, Figure 4 shows the bar plots of the area under the ROC (auROC) for the three methods for skeleton recovery of 9 networks sets when the FPR is restricted to [0-0.06]. Note that, because these are partial ROC curves, the auROC is normalized by the maximum possible auROC which is 0.06.

Based on both Figures 3 and 4, the proposed joint CPPD criterion (LMM and LMM-1) is shown to consistently improve the accuracy of skeleton recovery as compared to just taking the minimum of

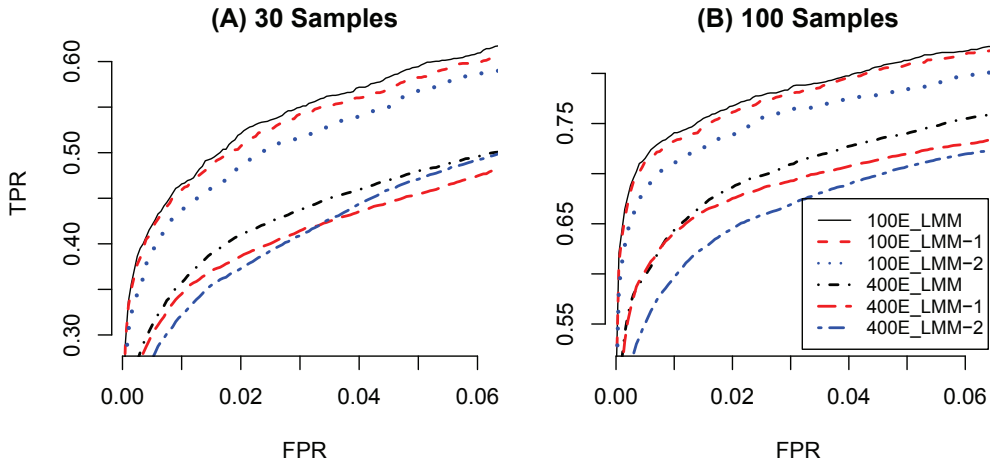



Figure 3: Average ROC of skeleton recovery of LMM, LMM-1, and LMM-2 when the number of true edges is ∼100, and ∼400 and the number of observations is: A) 30 and B) 100 samples. X-Axis is the false positive rate (FPR) and Y-Axis is the average true positive rate (TPR).

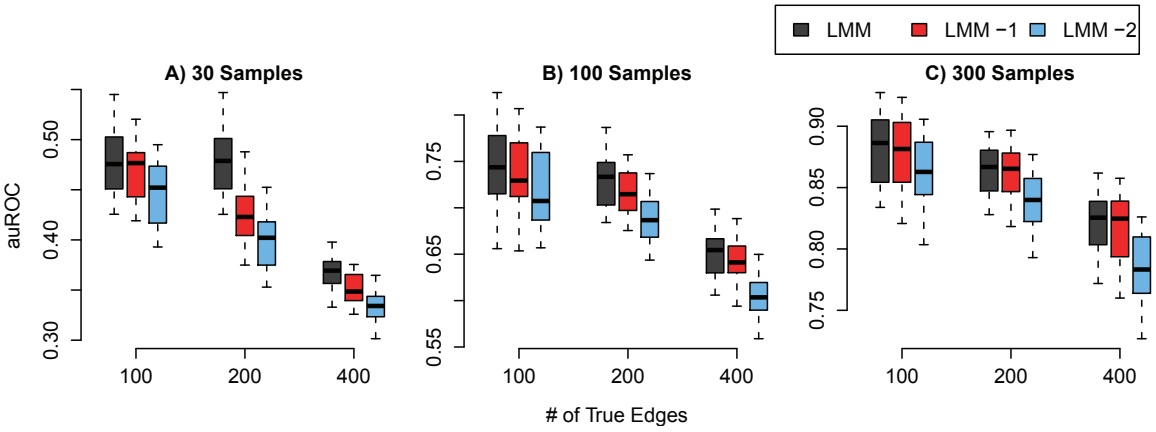



Figure 4: Bar plots of the auROC of skeleton recovery of LMM, LMM-1, and LMM-2 when the number of true edges is ∼100, ∼200, and ∼400 and the number of observations is: A) 30 B) 100 and C) 300 Samples. X-Axis is the correct number of edges in the true networks and Y-Axis is the area under the partial ROC (auROC) where FPR is constrained to $[0, 0.06]$. The plotted auROC is also normalized by the maximum possible area under the partial ROC.

the one-sided CPPDs from both sides of an edge (LMM-2). Also, the proposed adaptive reduction of independence testing (used in LMM only) is shown to improve the accuracy further, especially when the number of observations is small and the true graph is not sparse. This is consistent with multiple testing being a problem when learning from small sample data. Multiple testing is also not expected

to be a significant issue when recovering sparse networks where the variance of connection density is small. We also compared the performance of LMM to the PC and MaxMin algorithms. Figure 5 shows the average partial ROC of the three algorithms in recovering the skeletons of four network sets. Note that FPRs greater than 0.035 are ignored because solutions with more than 0.035 FPR are unlikely to have any practical use and it also is impractical to run the PC and MaxMin algorithms to recover denser skeletons since the number of required independence tests grows exponentially with the size of neighborhood.

In addition, we computed the area under the ROC curve where FPR is bounded to [0-0.035] for every case. Figure 6 shows the bar plots of the auROC for the three methods for the 9 network sets where the auROC is normalized by the maximum possible auROC, which is 0.035.

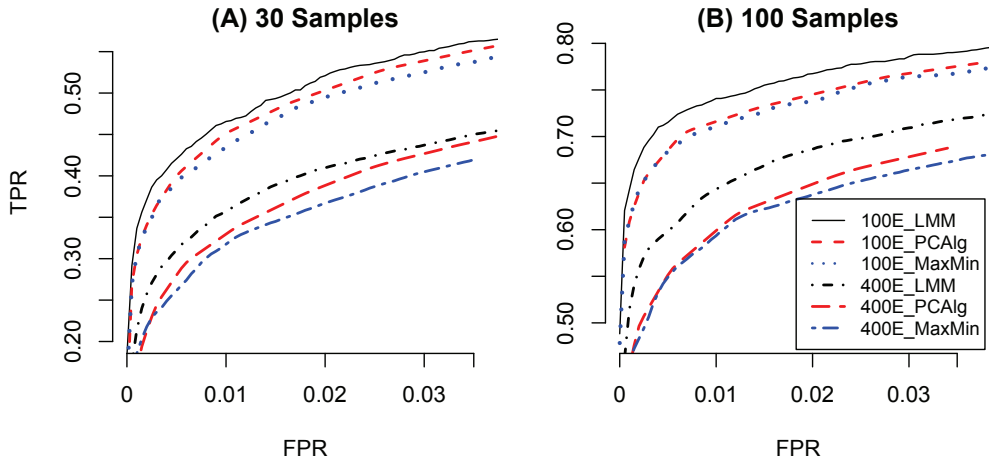



Figure 5: Average ROC of skeleton recovery of LMM, PC, and MaxMin when the number of true edges is ∼100, and ∼400 and the number of observations is: A) 30 and B) 100 samples. X-Axis is the false positive rate (FPR) and Y-Axis is the average true positive rate (TPR).

Based on both Figures 5 and 6, LMM consistently outperforms both algorithms in all cases. Also, similar to first evaluation, the improvement is less significant when the correct graph is generally sparse (i.e., 100 edges), since multiple testing is not an issue in very sparse graphs. The bar plots show the improvement to be consistent and not a result of outliers.

### 7.2.2 COMPUTATIONAL COMPLEXITY OF SKELETON RECOVERY

To experimentally compare the time complexity of the three algorithms (LMM, PC, and MaxMin) for skeleton recovery, we perform two types of comparisons. The first aims at evaluating the computational complexity when we have a large sample size, where the three algorithms can recover skeletons with high accuracy. The second comparison aims at evaluating the computational complexity when the sample size is limited.

For the first evaluation, we simulated 9 sets of networks where every set contains 40 different networks (see Table 1). Afterward, a set of 10,000 observational samples were generated from every network.

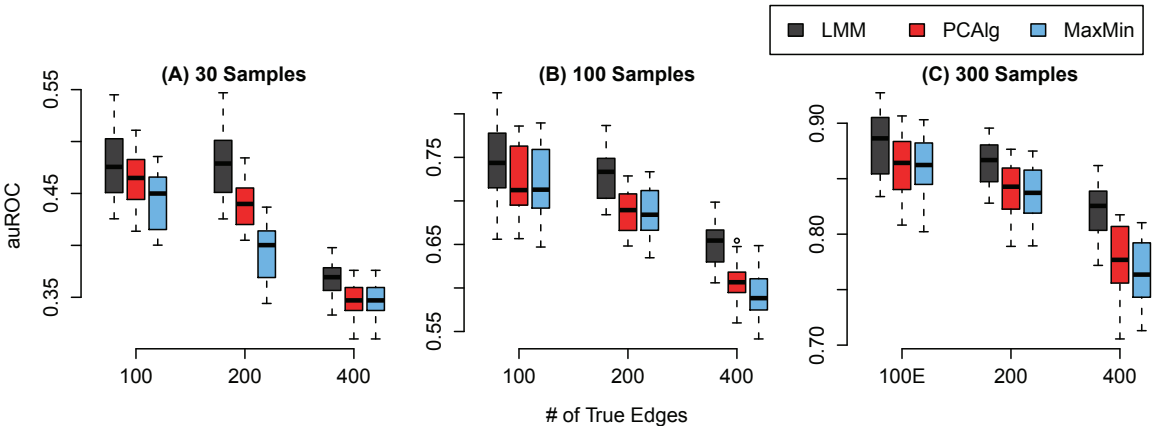



Figure 6: Bar plots of the auROC of skeleton recovery of LMM, PC, and MaxMin when the number of true edges is ∼100, ∼200, and ∼400 and the number of observations is: A) 30 B) 100 and C) 300 samples. X-Axis is the correct number of edges in the true networks and Y-Axis is the area under the partial ROC (auROC) where FPR is constrained to $[0, 0.035]$. The plotted auROC is also normalized by the maximum possible area under the partial ROC.

In order to compare complexity when the three methods recover highly accurate skeletons, we performed a search for the optimal parameters that resulted in the least distance between the recovered skeletons and the true skeletons as measured by structural hamming distance.

In the case of PC and MaxMin, we ran each algorithm to recover every network using different values for the threshold of the independence tests ($\alpha = 10^{-4}, 3\times10^{-4}, 10^{-3}, 3\times10^{-3}, 0.01, 0.03, 0.1$). Similarly, we ran LMM multiple times using Approach 1 described in Section 6.3.2 with different thresholds for the CPPD (Algorithm 1:$\gamma = 0.01, 0.03, 0.1, 0.3$). Afterward, for every method and every network set, the threshold that resulted in the most accurate skeletons was chosen. Note that, here, we define the distance between two skeleton as the number of operations of adding and deleting edges that are needed to transform one skeleton into the other. This distance is a special case of the structural Hamming distance (SHD) between two PDAGs defined by Tsamardinos et al. (2006).

Table 1 shows the average SHD and time in seconds when each method is used with the optimal parameters to recover every set of networks. The table, also, shows the standard deviation of both SHD and time in small font.

Based on Table 1, in terms of accurate recovery, the three algorithms are shown to perform equally well when given a large number of samples. On the other hand, in terms of computational time, both MaxMin and LMM outperform the PC algorithm significantly in all cases. However, LMM seems to be slightly slower than MaxMin especially when the true graph is sparse. It is important to note that the proposed relaxation in LMM only works when the observational sample size is small (see Section 6.2). We should also note that, both LMM and MaxMin used in the experiments presented in this section are implemented in Java while the PC algorithm is implemented in R which is known to be slower than Java. This partially explains why the PC algorithm was the slowest performing algorithm in this experiment.

| | | Average SHD | | | Average Time | | |
|---|---|---|---|---|---|---|---|
| Nodes | Edges | PC | MaxMin | LMM | PC | MaxMin | LMM |
| 100 | 100 | **2**$_{\pm 1.2}$ | 2.4$_{\pm 1.3}$ | 2.1$_{\pm 1.3}$ | 3.7$_{\pm 0.9}$ | **0.3**$_{\pm 0.1}$ | 0.8$_{\pm 0.3}$ |
| | 200 | 25$_{\pm 9.4}$ | **24**$_{\pm 1}$ | 26$_{\pm 9}$ | 32$_{\pm 8}$ | 4.2$_{\pm 2.7}$ | **3.2**$_{\pm 1.1}$ |
| | 300 | 101$_{\pm 22}$ | **92**$_{\pm 21}$ | 95$_{\pm 22}$ | 107$_{\pm 20}$ | 20$_{\pm 5}$ | **18** $_{\pm 7}$ |
| 300 | 300 | 6.6$_{\pm 1.8}$ | 5$_{\pm 1.4}$ | **3.4**$_{\pm 0.7}$ | 24$_{\pm 3.6}$ | **1**$_{\pm 0.3}$ | 3.1$_{\pm 1}$ |
| | 600 | 39$_{\pm 3.5}$ | 38$_{\pm 11}$ | **31**$_{\pm 12}$ | 121$_{\pm 19}$ | 15$_{\pm 4.1}$ | **14**$_{\pm 4.3}$ |
| | 900 | 188$_{\pm 31}$ | **184**$_{\pm 24}$ | 186$_{\pm 30}$ | 917$_{\pm 91}$ | 118$_{\pm 13}$ | **113**$_{\pm 27}$ |
| 1,000 | 1,000 | 19$_{\pm 4}$ | 30$_{\pm 5.3}$ | **15**$_{\pm 3.5}$ | 254$_{\pm 7}$ | **7**$_{\pm 0.5}$ | 18$_{\pm 0.6}$ |
| | 2,000 | 64$_{\pm 11}$ | 64$_{\pm 13}$ | **58**$_{\pm 12}$ | 612$_{\pm 59}$ | **53**$_{\pm 10}$ | 76$_{\pm 14}$ |
| | 3,000 | 295$_{\pm 31}$ | 278$_{\pm 34}$ | **269**$_{\pm 30}$ | 3,235$_{\pm 370}$ | **995**$_{\pm 146}$ | 1,210$_{\pm 201}$ |

Table 1: Average execution time in seconds and average SHD of each algorithm when recovering skeletons of BNs of given sizes. Best results for each network set are in bold face. Each result is presented as mean $\pm$ standard deviation.

Nonetheless, the superior computational advantage of MaxMin is only valid when the sample size is large. As will be shown in the next experiment, the complexity of MaxMin increases fast when the given sample size is small due to the correction step of asymmetric connectivity being deferred to the last phase of the algorithm.

In order to compare the computational complexity of the three algorithms when the sample size is small, we only consider one set of 40 networks of 300 nodes and $\sim$600 edges each. We generated 100 observational samples from every network and every algorithm is used to recover many solutions of different sizes for every network. We also compare to the unrelaxed version of LMM (LMM-1) where we disable the adaptive reduction of independence testing (Algorithm 1: $\omega = 1$). The goal here is to test how computational complexity can grow as we attempt to retrieve various number of edges. Learning BN from small sample data is approximate in nature. While in many cases users are interested in a solution with the least SHD, in other cases, users may also be interested in various levels of recall (true positive rate), which can only be achieved by retrieving various numbers of edges. Here, we compare the four methods (LMM, LMM-1, PC, and MaxMin) when used to retrieve skeletons of different sizes up to 1.8 times the size of the true skeleton. This is done by running every method on every network many times using different thresholds for the independences tests. Figure 7 shows both the average TPR and the average computation time versus the number of retrieved edges.

Based on Figure 7, LMM retrieves more accurate skeletons in much less time than all other methods including LMM-1. By taking advantage of adaptive relaxation of constraints, LMM can ease the multiple testing problem and at the same time reduce the computational complexity. On the other hand, the MaxMin algorithm computational time grew fast as we tried to retrieve denser solutions. To understand this behavior, we debugged the code of MaxMin as it recovers the skeleton from small sample data and found that this dramatic increase in computation time is due to the correction of asymmetric connectivity step being deferred to the last phase of the algorithm (Tsamardinos et al., 2006). MaxMin works by searching for child and parent variables for every node separately and at the end, for an edge to be selected, both nodes have to appear as neighbors to each other. However, in the case of small sample data, the asymmetric connectivity seems to

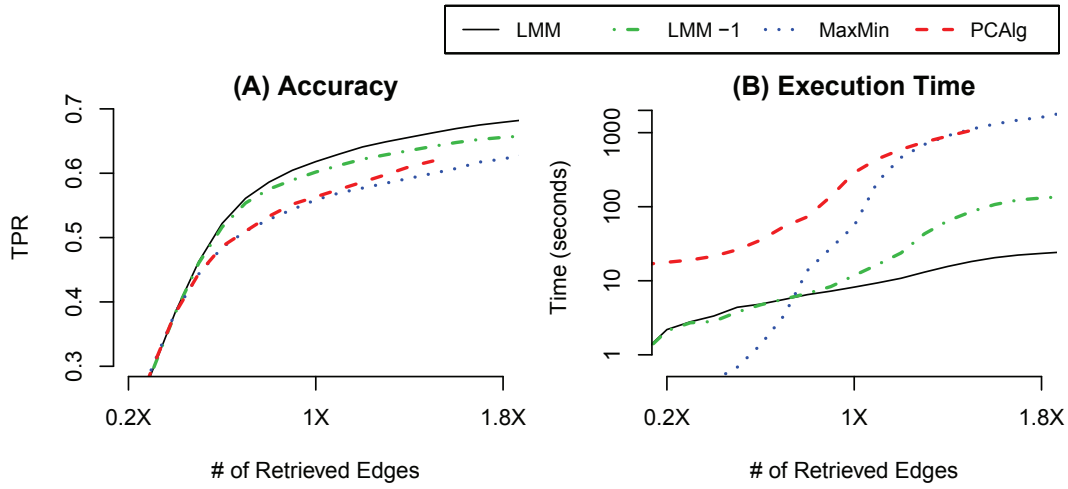



Figure 7: Average TPR (A) and computation time (B) when using LMM, LMM-1, PC, and MaxMin to retrieve varying numbers of edges for 40 networks (each composed of 300 nodes and ∼600 edges) based on limited data (100 samples). X-Axis is the number of retrieved edges normalized by the true number of edges (X = Number of edges in the correct network). Time-Axis is log-scaled.

be common and a lot of edges are rejected leading to a small number of recovered edges, even when every node identified a relatively large number of candidate neighbors. Therefore, to retrieve a larger number of edges, one will need to use a larger $\alpha$ leading to larger neighbor sets in the first phase and hence many more independence tests. A similar finding about the effect of symmetry correction on the computational efficiency of MaxMin has also been reported and discussed by Aliferis et al. (2010). However, unlike the suggestion made by Aliferis et al. (2010) which eliminates the symmetry correction step, all variants of LMM presented in this work avoid the deterioration of computational requirements without sacrificing the asymptotic correctness of the inference by performing early correction of symmetric connectivity. For example, when considering LMM-1, once an edge $E_{xy}$ is added or deleted, both neighbor sets $CP_x$ and $CP_y$ are updated accordingly and immediately. This early joint update of neighbor sets informs nodes not to condition on false neighbors early on as the algorithm proceeds. This explains why LMM-1 is faster than MaxMin for dense solutions, although it does not relax independence testing.

We again note that in addition to this low computational cost, the algorithm offers an ease of use property where users can make the algorithm retrieve a fixed number of edges or a ranked list of edges (see Section 6.3.2).

### 7.2.3 CPDAG RECOVERY

To evaluate the extended LMM for CPDAG recovery, we compared it to the PC and MMHC algorithms by using each method to recover multiple CPDAGs of different sizes for every single network. Afterward, a plot of the true positive rate (TPR) versus the number of retrieved edges is generated for each network. When the curves are plotted in conjunction with a curve representing a hypothetical optimal recovery, the plot becomes a compact representation where one can read the

TPR, false positives (FP), and false negatives (FN) at every point on the curve. In addition, comparing two methods using the curve of TPR versus the number of retrieved edges has the same semantic as a ROC curve where higher always means better, in that, at a fixed number of retrieved edges (X-Axis), higher TPR also means lowers FP and FN. This makes it a more informative representation of results than the structural Hamming distance plot (SHD) used by other authors (Tsamardinos et al., 2006). In the Appendix, we provide an illustrative example comparing the two metrics and we also provide the SHD plots for all experiments.

First, we used all algorithms to recover CPDAGs for two sets of 100 networks each. The first set is simulated so that every network had ∼100 edges while the second set is simulated so that every network had ∼300 edges. Figure 8 shows the average TPR of the CPDAGs recovered by LMM_EO, LMM_EO-1, MMHC, and PC algorithm for each set separately when given the same number of training samples (100 samples). LMM_EO-1, refers to using LMM_EO with a random tie breaking for conflicting v-structures. The figure shows the average TPR of each algorithm at different levels of recovery (number of retrieved edges) up to 30-60% more edges than the true number of edges. In the case of using MMHC for recovering network of ∼300 edges (Figure 8:B), we were not able to get BNLearn to recover CPDAGs with more than ∼10% edges more than the true CPDAG due to the deletion of edges performed by hill climbing search to maximize the score.

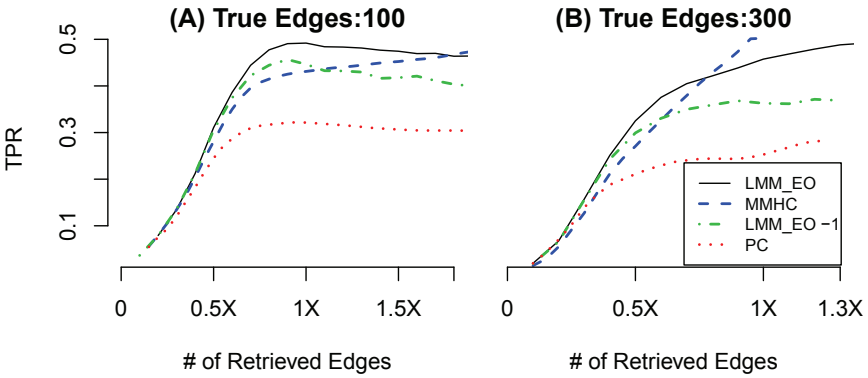



Figure 8: Average TPR when recovering CPDAGs of networks of 100 nodes and A) ∼100 or B) ∼300 true edges when the number of samples is 100. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average true positive rate (TPR).

As illustrated by the comparison of LMM_EO and LMM_EO-1 in Figure 8, the proposed approach for resolving conflicting v-structures consistently improves the accuracy of orienting the edges as compared to random selection among conflicting v-structure. Also LMM is shown to significantly outperform the PC algorithm while being competitive to MMHC in this case. To illustrate the variance of recovery among all recovered networks, Figure 9 shows the bar plots for the TPR when recovering networks containing as many edges as 0.7, 1.0, and 1.3 times the number of true edges in each network. Figure 9 is complementary to Figure 8 where the improvement is shown to be consistent and not a result of outliers.

In the second set of experiments, we used six sets of networks of sizes 100, 500, and 2500 nodes and two different levels of observations: 100 and 300 samples. All networks are simulated so

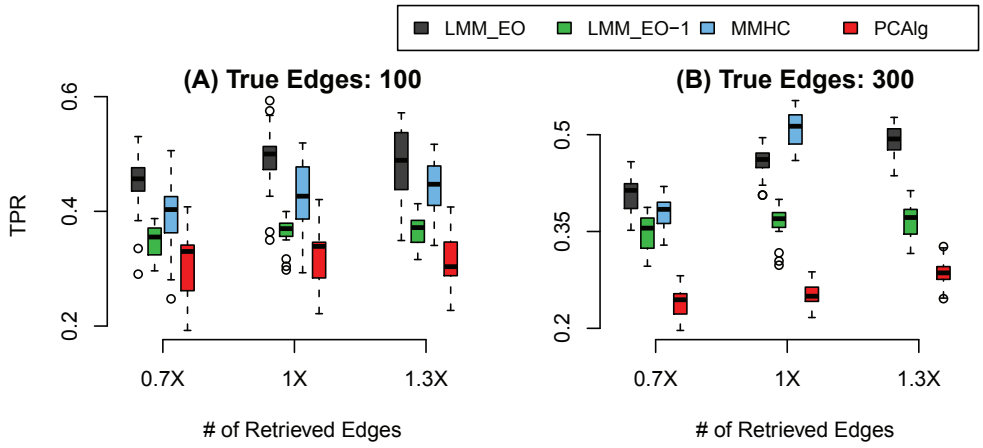



Figure 9: TPR of CPDAGs recovery of networks of 100 nodes and A) ∼100 or B) ∼300 edges. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the true positive rate (TPR).

that the number of edges is two times the number of nodes in every network. Figure 10 shows the average TPR for every method for the recovery of the CPDAGs of every set while Figure 11 shows the corresponding bar plots for the TPR when the number of recovered edges is as many as 0.7 and 1.0 times the number of true edges.

Based on Figures 10 and 11, in terms of accurate recovery, LMM scales well for larger networks, where at the same level of density and number of observations, LMM has almost the same level of TPR. In addition, LMM has far more accurate recovery than the PC algorithm for different network sizes. When compared to MMHC, in the 100 nodes case, LMM seems to have comparable accuracy. However, in the 500 and 2,500 variables cases, LMM is more accurate. The deterioration of MMHC could be a result of the local minimum problem, in that, global optimization techniques, such as the Tabu search with hill climbing, become less effective as the search space grows very large and the solution becomes prone to converging to a bad local minimum.

Finally, to empirically test the correctness of LMM, we used LMM to recover the CPDAGs of medium size networks (500 nodes) with two different levels of density (500 and 1000 edges) as the number of observations becomes large. For every setting, 100 different networks were generated. Figure 12 shows the average TPR when recovering CPDAGs of different sizes at three different levels of observations (100, 1,000, and 10,000 samples).

Figure 12 shows that the recovered CPDAGs by LMM converge steadily towards the true CPDAGs as the number of observations grows large. For example, when the number of observations increased ten-fold, the difference between the recovered CPDAGs of the accurate size (correct number of edges) and the true CPDAGs decreased about four-fold each time.

## 8. Conclusion

In this work, we have presented a new constraint-based algorithm, light mutual min (LMM), for structural learning of Bayesian networks. When the observational data are limited in size, LMM improves the assessment of candidate skeleton edges by ranking them based on an estimate of the

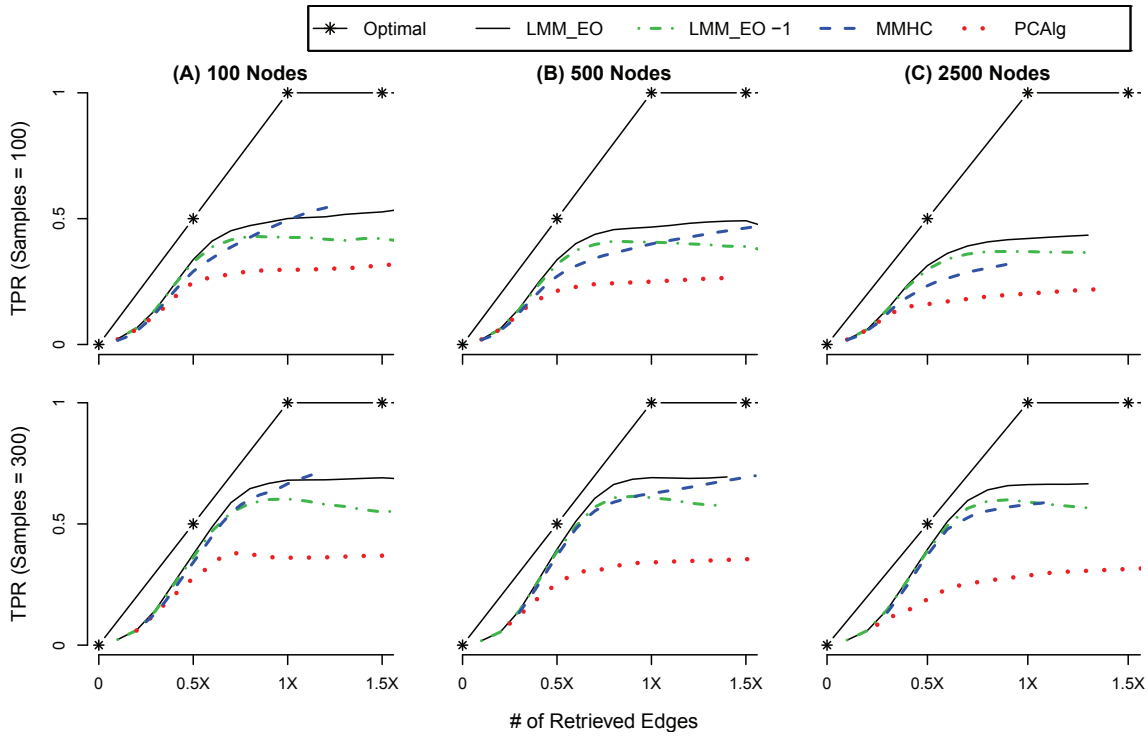



Figure 10: Average TPR when recovering CPDAGs of networks of A) 100, B) 500, and C) 2500 nodes with ∼200, ∼1,000, and ∼5,000 true edges respectively, when the number of samples is 100 (1st row) and 300 (2nd row). X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average true positive rate (TPR)

joint conditional posterior that none of the neighboring variables at either sides of an edge can render the considered two variables conditionally independent. This approach is motivated by the asymptotic property, under the faithfulness assumption, that a parent and a child variable cannot be found independent when conditioning on any neighboring variables on either side of the edge. Therefore, considering conditional independence tests at both sides of the edge simultaneously provides complementary sources of evidence that can improve the assessment of candidate edges. In addition, to ease the multiple testing problem in recovering dense areas of the skeleton, LMM employs an adaptive relaxation of independence testing by, selectively, allowing some nodes not to condition on some of their neighbors. This relaxation is only performed whenever asymmetric evidence of conditional dependence is found between a pair of connected variables, where the aim is to reduce the accidental rejection of true edges connecting high degree nodes due to multiple testing. As the number of observational samples increases, the asymmetric evidence of dependence is expected to become less common and therefore the relaxation will be applied less often. This property makes the proposed technique work adaptively depending on the number of observational samples while at the same time maintains the asymptotic correctness of the algorithm. Moreover, this relaxation re-

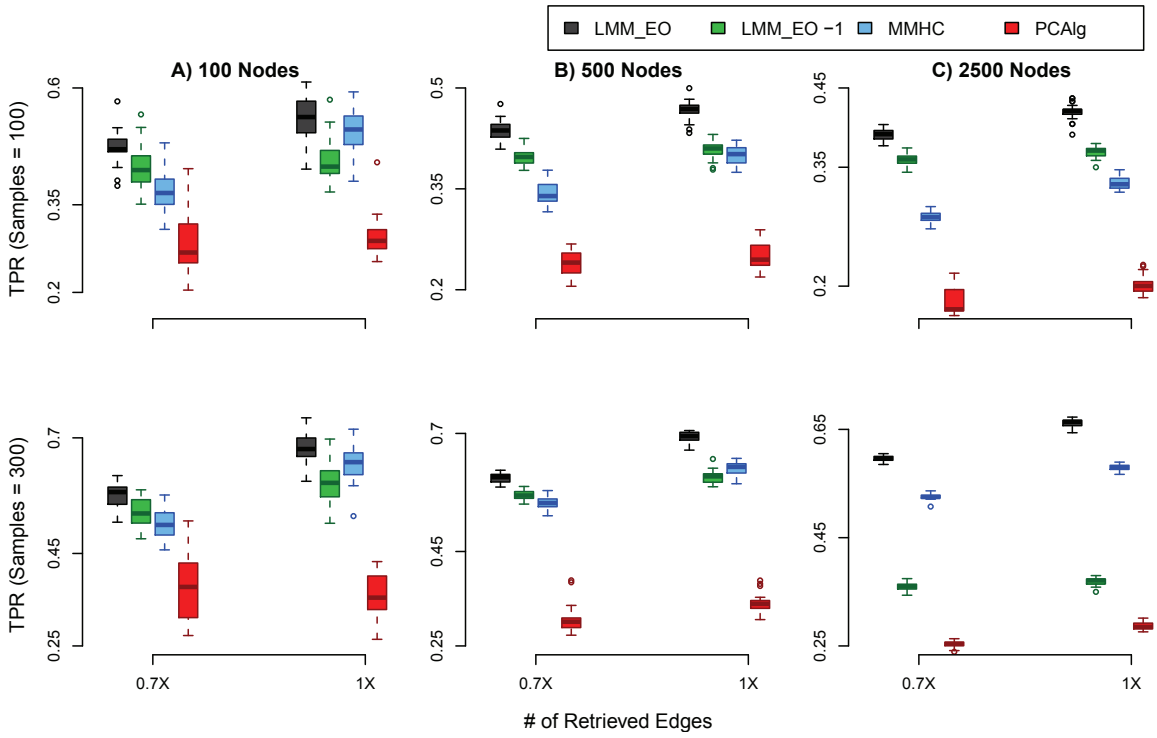



Figure 11: TPR of CPDAGs recovery of networks of size: A) 100, B) 500, and C) 2500 nodes with ∼200, ∼1,000, and ∼5,000 true edges when number of samples is 100 (1st row) and 300 (2nd row). X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the true positive rate (TPR).

sults in a significant reduction in the number of independence tests making LMM a computationally competitive learning tool for high-dimensional graphs and graphs with non-sparse connectivity.

In addition, we proposed a new approach to recover v-structures in a given skeleton based on the confidence about the dependence induced by the addition of the common neighbor to conditioning sets. The advantage of this method is the ability to rank candidate v-structures, providing a tool for resolving conflicts. When LMM is extended to recover the CPDAG of the equivalence class, the proposed conflict resolution method improved the accuracy of the recovered CPDAGs.

## Appendix A. Proof of Correctness

In this section, we provide proofs of correctness of the proposed methods for the recovery of both the correct skeleton and CPDAG when given a very large number of observational samples (asymptotic limit). The provided proofs are similar in style and content to the proofs that were given for the MaxMin (Tsamardinos et al., 2006) and the PC algorithms (Kalisch and Bühlmann, 2007) with necessary modifications. First, we have to make the following necessary assumptions:

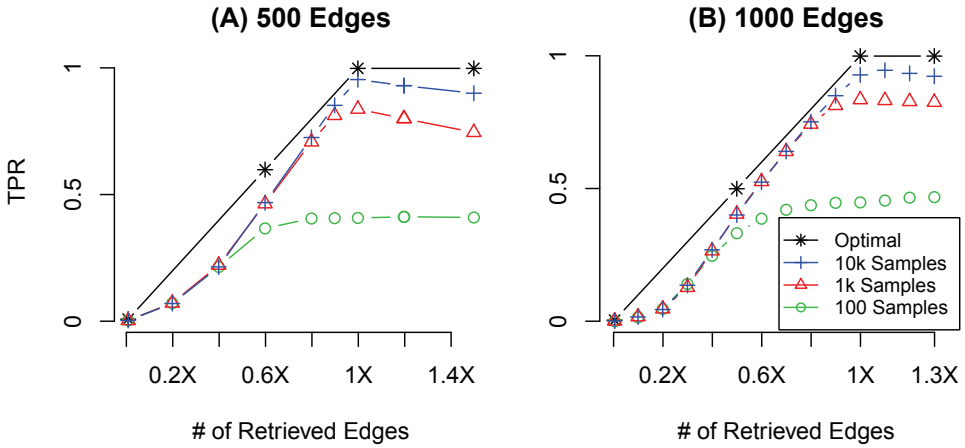



Figure 12: Average TPR of the recovered CPDAGs as the number of observations grows large when recovering networks of 500 nodes and A) ∼500 edges, and B) ∼1000 edges. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the true positive rate (TPR).

A1: The observational samples are realizations of i.i.d random vectors $X_1, X_2, ...X_n$ with $X_i \in \Re^p$ and they are generated from a DAG $G = (V, E)$ with $p$ nodes and a corresponding probability distribution $P_n$.

A2: The distribution $P_n$ is multivariate Gaussian and faithful to the DAG $G$.

A3: The partial correlation $\rho_{ij|Z}$ between two variables $i$ and $j$ when conditioned on a set $Z$ is an indicator of conditional independence under $P_n$ where

$$\rho_{ij|Z} = 0 \; iff \; i \perp\!\!\!\perp j|Z.$$

A3 implies that the partial correlation method is an appropriate way to measure conditional independence from the given data and it is a necessary assumption for any constraint-based approach that uses this method for independence testing to be asymptotically correct. Based on A3 and given that the sample partial correlation $\hat{\rho}_{ij|Z}$ is an asymptotically correct estimator of $\rho_{ij|Z}$ (Hotelling, 1953), the following becomes correct:

$$\lim_{n\to\infty} |\hat{\rho}_{ij|Z}| = 0 \; iff \; i \perp\!\!\!\perp j|Z.$$

As a result and by substitution in Equation (2), for all positive and non-infinitesimal $\pi_0$ and $\pi_A$ where $(\pi_0 + \pi_A = 1)$, the following also becomes correct:

$$\lim_{n\to\infty} P(\rho_{ij|Z} \neq 0|\hat{\rho}_{ij|Z}) = \begin{cases} 0 & iff \; i \perp\!\!\!\perp j|Z; \\ 1 & otherwise. \end{cases}$$

### A.1 Correctness of LMM for Skeleton Recovery

The proof of LMM correctness for skeleton recovery relies on the symmetric dependence between parent and child variables in the asymptotic limit. The proof is given by Lemma 17 which also depends on Lemmas 14, 15 and 16 .

**Lemma 14** *If $E_{ij}$ is a true edge in the skeleton of a DAG $G = (V,E)$, then the one-sided CPPD $P(E_{ij} \mid D)_{[Z]} = 1$, $\forall Z \subseteq V_{\setminus i,j}$.*

**Proof** From assumption A1, $\nexists B \subseteq V_{\setminus i,j}$ s.t. $i \perp\!\!\!\perp j \mid B$. Based on this fact and based on assumption A3, then $P(\rho_{ij|B} \neq 0 | \hat{\rho}_{ij|B}) = 1$, $\forall B \subseteq Z$ and $\forall Z \subseteq V_{\setminus i,j}$. Therefore, by substitution in Equation (3), if $E_{ij}$ is part of the correct skeleton then $P(E_{ij} \mid D)_{[Z]} = 1, \forall Z \subseteq V_{\setminus i,j}$. ■

**Lemma 15** *If $G_S$ is the set of all edges in the skeleton of G, and $\hat{G}_S^f$ is the set of all edges recovered in the forward selection phase of LMM$(D,\omega,\gamma)$, then $G_S \subseteq \hat{G}_S^f, \forall \gamma < 1$.*

**Proof** From Lemma 14, $\forall E_{ij} \in G_S$, $P(E_{ij}|D)_{[Z]} = 1, \forall Z \subseteq V_{\setminus i,j}$. Therefore, by substitution in Equation (4), the joint CPPD $P(E_{ij}|D)_{[CP_i,CP_j]} = 1, \forall CP_i, CP_j \subseteq V$. As a result, the first phase of LMM will not exit before all edges in $G_S$ are recovered and therefore, $G_S \subseteq \hat{G}_S^f$. ■

**Lemma 16** *If $CP_i^*$ is the set of all parent and child variables of node i in the correct skeleton $G_S$, and $CP_i$ is the set of neighbors of node i recovered in the forward selection phase of LMM$(D,\omega,\gamma)$, then $\forall \omega > 0$, $CP_i^* \subseteq CP_i$ (assuming $\gamma < 1$ and $\omega$ is not infinitesimal).*

**Proof** From Lemma 15, all $E_{ij} \in G_S$ are correctly recovered in the forward selection. Also, from Lemma 14, $P(E_{ij} \mid D)_{[CP_i]} = 1$, and $P(E_{ij} \mid D)_{[CP_j]} = 1, \forall i,j \; s.t. \; E_{ij} \in G_S$, and hence:

$$\lim_{n\to\infty} |P(E_{ij} \mid D)_{[CP_i]} - P(E_{ij} \mid D)_{[CP_j]}| < \omega, \; \forall \omega > 0.$$

As a result, lines 14 and 17 of LMM (Algorithm 1) will always get executed whenever a new edges is added when given a very large number of training samples. Therefore, after the forward selection, every node will be aware of all its child and parent variables, and thus $CP_i^* \subseteq CP_i$, $\forall i$.

Note that Lemma 16 emphasizes the property that the proposed relaxation of independence testing is adaptive, such that as the sample size of the data increases, the asymmetric one-sided CPPD among pairs of nodes connected in the true DAG occur less often and absent altogether when the sample size grows very large. ■

**Lemma 17** *If $G_S$ is the set of all edges in the skeleton of the graph G, and $\hat{G}_S$ is the set of all edges recovered by LMM$(D,\omega,\gamma)$, then $G_S = \hat{G}_S$.*

**Proof** From Lemma 15 and 16, after forward selection, all true edges were correctly recovered ($G_S \subseteq \hat{G}_S^f$), and the neighbor set of every node contains all its parent variables ($pa(i) \subseteq CP_i$, $\forall i$).

However, it is also expected that the forward selection might have recovered some false edges $E_{false} = \hat{G}_S^f - G_S$.

For every edge $E_{ij} \in \hat{G}_S^f$, since the graph is acyclic, then at least one of either $i$ or $j$ is a non-descendant of the other. From the local Markov property, a variable is independent from all its non-descendant variables if its parent variables are known. Therefore, $\forall\, E_{ij} \in E_{false}$, since $j \notin pa(i)$ and $i \notin pa(j)$, then either $P(E_{ij}|D)_{[CP_i]} = 0$ or $P(E_{ij}|D)_{[CP_j]} = 0$ (or both are zero) and by substitution in Equation (4):

$$P(E_{ij}|D)_{[CP_i,CP_j]} = P(E_{ij}|D)_{[CP_i]} \times P(E_{ij}|D)_{[CP_j]} = 0,\ \forall E_{ij} \in E_{false}.$$

On the other hand, if $E_{ij} \in \hat{G}_S^f$ is a correct edge, if $i$ is the descendant of $j$ then $j$ is the parent of $i$ ( not a descendant of $i$). However, $i$ cannot become independent of $j$ since $j$ will always be excluded from conditioning sets. Also, under the faithfulness property, $i$ and $j$ cannot be made independent when conditioning on any set of other variables and as a result:

$$P(E_{ij}|D)_{[CP_i,CP_j]} = P(E_{ij}|D)_{[CP_i]} \times P(E_{ij}|D)_{[CP_j]} = 1,\ \forall E_{ij} \in G_S.$$

Based on these conclusions, once the backward elimination starts, all false edges will have a joint CPPD of zero while all true edges will have a joint CPPD of one and therefore all false edges will be eliminated and thus: $G_S = \hat{G}_S$. ■

### A.2 Correctness of LMM_Orientation for CPDAG Recovery

**Lemma 18** *If CPDAG$(G)$ is the complete partially acyclic graph of the equivalence class of the correct graph G, and $\widehat{G}$ is the partially acyclic graph recovered by LMM_EO, then CPDAG$(G) = \widehat{G}$.*

**Proof** From Lemma 17, in the asymptotic limit, LMM will recover the correct skeleton of $G$:

$$skeleton(\widehat{G}) = skeleton(G)$$

Also, it follows from Lemma 13 (Section 6.5): for every non-adjacent pair $x$ and $y$ with a common neighbor $w$, that $w$ induces dependence whenever added to any set $Z$ that makes $x$ and $y$ independent if and only if $w$ is a common child of both $x$ and $y$. Therefore,

$$\min_{Z \subseteq CP_x, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) > \min_{Z \subseteq CP_x, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$$

and

$$\min_{Z \subseteq CP_y, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) > \min_{Z \subseteq CP_y, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$$

if and only if $\{x, y\} \subseteq pa(w)$.

By direct substitution in line 5 of Algorithm 3, it follows that $CD(x, w, y) > 0$ if and only if $x \rightarrow w \leftarrow y$ is a correct v-structure in the true graph G. Therefore, lines 1-17 of Algorithm 3, will recover all and only the correct v-structures of the true graph:

$$vstructures(\widehat{G}) = vstructures(G).$$

The rest of the proof follows from the correctness of the orientation rules (R:1-4) that was provided by Meek (1995), in that, when the correct skeleton and the correct and complete set of v-structures are given then, the four rules (Algorithm 3: lines 18-24) will recover the correct CPDAG of $G$. Therefore, $\widehat{G} = CPDAG(G)$. ■

## Appendix B. Effect of Overlapping Neighbor Sets on Ranking Edges

As stated in Section 6.1, the factorization in the right hand side of Equation (4) ignores the possibility that the two neighbor sets $CP_i$ and $CP_j$ might be overlapping. For example, if two nodes $x$ and $y$ found the common neighbor $w$, who also happen to be the only neighbor of both $x$ and $y$, to induce conditional independence with the highest statistical significance at both sides of the edge, the joint conditional posterior in (4) becomes $P(\rho_{xy|w} \neq 0|\hat{\rho}_{xy|w})^2$. Since $P(\rho_{xy|w} \neq 0|\hat{\rho}_{xy|w})$ is always less than one, the joint posterior in this case becomes a lower bound of the posterior that $x$ and $y$ are not conditionally independent. However, as shown in the following examples, the overlap of neighbor sets is not expected to have a negative effect on the overall ranking of edges. The presented examples are not a proof of robustness but rather a supportive argument that the proposed ranking criterion is not expected to be sensitive to the issue of overlapping neighbors sets. To illustrate why this is the case, we analytically compute the expected rank of multiple edges under different circumstances while controlling for all other variables by making the following assumptions about the learning problem:

B1: $G$ is the correct skeleton and the inference is based on finite data D.

B2: At the current iteration of the algorithm, $CP_x$, $CP_y$, $CP_i$, and $CP_j$ are the inferred neighbor sets of $x$, $y$ ,$i$ , and $j$ respectively. For simplicity and to control for other variables we will also assume:

(a) All neighbor sets have the same size.

(b) All neighbor sets contain the correct neighbors of each corresponding node.

(c) All conditioning sets are of the same size, or alternatively the effect of the size of the condition set on the sample partial correlation is negligible.

Again, all of these assumptions aim to control for variables and allow for an analytical solution for Equation (4) for the purpose of discussion.

B3: In all examples, we are only considering the average case, in that, we are sampling similar problems an extremely large number of times and we are only computing the average scenario. In all these problems, the size of the observational data and the structure of the skeleton are fixed while all other parameters of the simulation are randomized. As a result and based on Theorem 10, assumption B1, and the properties of partial correlations, it follows that there exist two constants $\rho_0$ and $\rho_0^+$ such that:

- If $D_{sep_G}(x,y|Z)$, then $E(|\rho_{xy|Z}|) = \rho_0$.
- If $\neg D_{sep_G}(x,y|Z)$, then $E(|\rho_{xy|Z}|) = \rho_0^+$.
- $1 \geq \rho_0^+ > \rho_0 \geq 0$.

These properties of the partial correlations are not new assumptions but rather a consequence of the assumption that partial correlation can identify conditional independence more accurately than random when learning from finite sample data. Based on this conclusion and by substitution in Equations (2) and (3), it follows that there also exist two constants $\mathcal{P}_0$ and $\mathcal{P}_0^+$ such that the following also holds:

- If $\exists Z \subseteq CP_x$ s.t. $D_{sep_G}(x,y|Z)$, then $E(P(E_{xy}|D)_{[CP_x]}) = \mathcal{P}_0$.
- If $\nexists Z \subseteq CP_x$ s.t. $D_{sep_G}(x,y|Z)$, then $E(P(E_{xy}|D)_{[CP_x]}) = \mathcal{P}_0^+$.
- $1 \geq \mathcal{P}_0^+ > \mathcal{P}_0 \geq 0$.

Under Assumptions B1-3, we now analytically compute the expected rank of two edges ($E_{xy}$, $E_{ij}$) in different circumstances where we assume two nodes to have full overlap of neighbors and the other two nodes to have no common neighbors.

*Case 1*: $E_{xy}, E_{ij} \in edges(G)$ and $CP_x = CP_y$ while $CP_j \cap CP_i = \emptyset$.
*Analysis*: Since $E_{xy}$ and $E_{ij}$ are in the true skeleton, there is no set $Z$ that d-separates $x$ and $y$ or $i$ and $j$. Using assumption B3 and substitution in Equation (4), the expected joint conditional posterior of both edges are equal:

$$E(P(E_{xy}|D)_{[CP_x,CP_y]}) = E(P(E_{ij}|D)_{[CP_i,CP_j]}) = (\mathcal{P}_0^+)^2.$$

As seen by this example, having common neighbors did not have negative or positive effect on ranking a correct edge against another correct edge in the average case.

*Case 2*: $E_{xy}, E_{ij} \notin edges(G)$ and $CP_x = CP_y$ while $CP_j \cap CP_i = \emptyset$.
*Analysis*: Given that $E_{xy}$ is not part of the correct skeleton, it follows from the faithfulness assumption that there is at least one subset $Z$ in either $CP_x$ or $CP_y$ that d-separates $x$ and $y$. Since $CP_x$ and $CP_y$ are identical, the d-separation set exists in both $CP_x$ and $CP_y$ (Tsamardinos and Brown, 2008). Therefore, using assumption B3 and substituting in Equation (4), the expected joint conditional posterior of $E_{xy}$ becomes:

$$E(P(E_{xy}|D)_{[CP_x,CP_y]}) = (\mathcal{P}_0)^2.$$

On the other hand, when assessing the Edge $E_{ij}$, there are two possibilities (Tsamardinos and Brown, 2008):

1. $\exists Z_1 \subseteq CP_i$ s.t. $D_{sep_G}(i,j|Z_1)$ and $\exists Z_2 \subseteq CP_j$ s.t. $D_{sep_G}(i,j|Z_2)$. Using Assumption B3 and by substitution in Equation (4), the expected joint conditional posterior of $E_{ij}$ becomes:

$$E(P(E_{ij}|D)_{[CP_i,CP_j]}) = (\mathcal{P}_0)^2.$$

2. There is at least one subset in either $CP_i$ or $CP_j$ (not both) that d-separates $i$ and $j$. Using Assumption B3 and by substitution in Equation (4), the expected joint conditional posterior of $E_{ij}$ becomes:

$$E(P(E_{ij}|D)_{[CP_i,CP_j]}) = (\mathcal{P}_0) \times (\mathcal{P}_0^+).$$

As illustrated by Case 1 and Case 2, on average, the algorithm is expected to rank correct edges (using Equation 4) higher than false edges regardless whether there is or there is not overlap among neighbor sets:

$$(\mathcal{P}_0^+)^2 > (\mathcal{P}_0) \times (\mathcal{P}_0^+) > (\mathcal{P}_0)^2.$$

On the other hand, it is evident from Case 2 that Equation (4) might introduce bias in ranking false edges among each other in the presence of overlapping neighbor sets. This bias however is not enough to rank false edges higher than correct edges and this is the most crucial part of the learning algorithm.

## Appendix C. Illustration of the CPDAG Evaluation Metric

Figure 13 shows an illustrative comparison between the structural Hamming distance (SHD) curve and the TPR curve. In both plots, the X-axis is the number of retrieved edges normalized by the correct number of edges. Also, all the metrics: true positives (TP), false negatives (FN), false positives (FP), and SHD are normalized by the number of correct edges. SHD counts the number of operations (add edge, delete edge, or change orientation) needed to transform a given CPDAG to the correct CPDAG. TP counts the number of recovered edges that are correct in presence and orientation. FP counts the number of incorrectly recovered or miss-oriented edges. FN counts the number of true edges that are either not recovered or miss-oriented.

In the TPR plot, the FN becomes (1-TP) while FP becomes the number of retrieved edges minus TP, and both of these metrics can be easily observed in the plot as shown in Figure 13:B. In contrast, it is not trivial to extract such information from the SHD curve.

Note that a mis-oriented edges is counted only once in the SHD metric, while it is counted twice in the TPR plot, once as a FP (false direction) and another as a FN (missing correct direction).

All presented plots in this paper aim at comparing methods. However, in the case of model selection, the solution with the least SHD should be the target. Another choice is to seek a CPDAG with the highest TP and the lowest FP (e.g., highest TP$-$FP). Also, we note that many authors usually use the X-axis to represent the threshold used in independence testing. However, in all plots in this paper, we used the number of retrieved edges as alternative because LMM does not use a threshold similar to MaxMin or PC algorithm.

Figures 14, 15, and 16 plot the average SHD versus the number of retrieved edges for the same experimental results plotted by the TPR plots in Figures 8, 10, and 12.

## References

C. Aliferis, I. Tsamardinos, A. Statnikov, and L. Brown. *Causal Explorer:Causal Probabilistic Network Learning Toolkit for Biomedical Discovery*, 2003. URL `http://www.dsl-lab.org/causal_explorer`.

C.F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X.D. Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification part ii: Analysis and extensions. *The Journal of Machine Learning Research*, 11:235–284, 2010.

A. Armen. Estimation and control of the false discovery rate in Bayesian network skeleton identification, with application to biological data. Master's thesis, Computer Science Department, University of Crete, Heraklion, Crete, Greece, 2011.

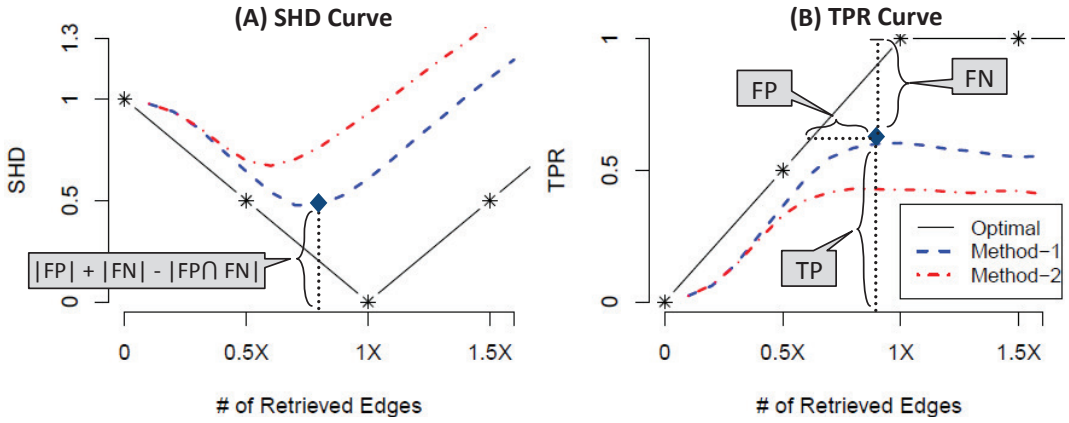



Figure 13: Illustration of A) SHD and B) TPR plots. In both plots, the X-Axis is the number of retrieved edges normalized by the number of correct edges. The metrics: FP, FN, TP, and SHD are also normalized by the correct number of edges. In the SHD plot, lower is better while in the TPR Plot, higher is better.

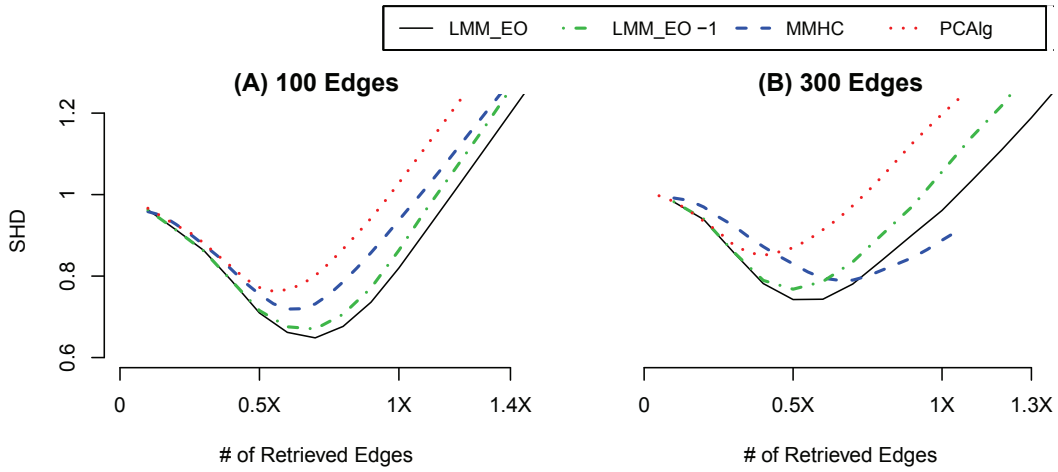



Figure 14: Average SHD when recovering CPDAGs of networks of 100 nodes and A) ∼100 or B) ∼300 true edges when the number of samples is 100. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average normalized structural Hamming distance (SHD).

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. ISSN 0004-3702.

D. Chickering, D. Heckerman, and C. Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004. ISSN 1532-4435.

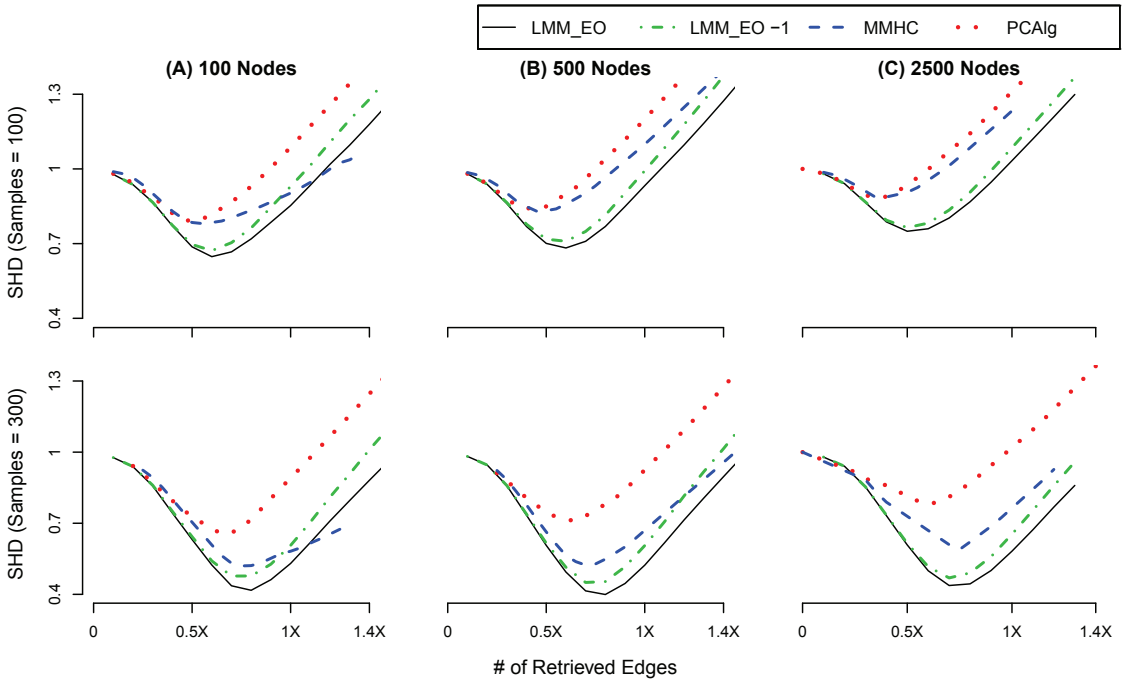



Figure 15: Average SHD when recovering CPDAGs of networks of A) 100, B) 500, and C) 2500 nodes with ∼200, ∼1,000, and ∼5,000 true edges respectively, when the number of samples is 100 (1st row) and 300 (2nd row). X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average normalized structural Hamming distance (SHD).

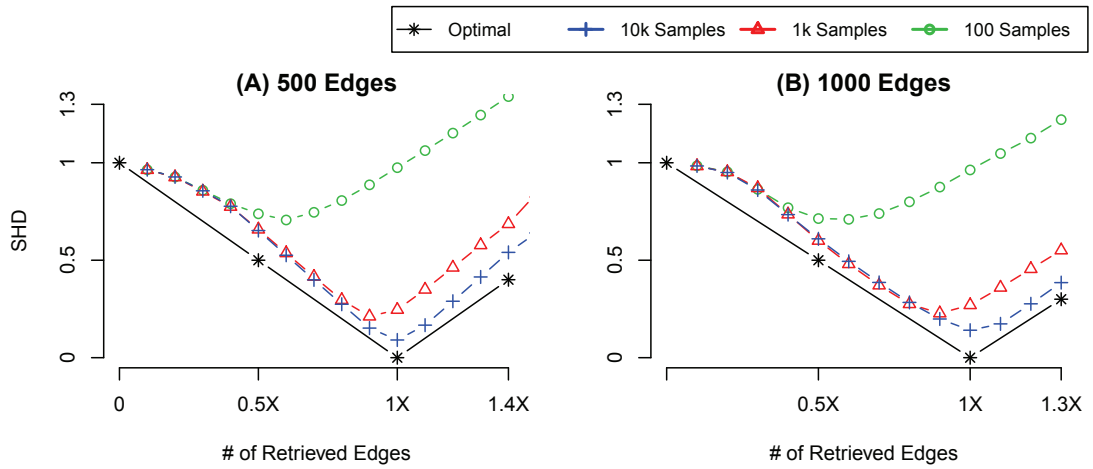



Figure 16: Average SHD of CPDAG recovery by LMM_EO as the number of observations grows large when recovering networks of 500 variables and A) ∼500 edges, and B) ∼1000 edges. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average normalized structural Hamming distance (SHD).

G. Cooper and E. Herskovits. Using Bayesian networks to analyze expression data. *Machine Learning*, 9:309–347, 10 1992.

R. Cowell, S. Lauritzen, P. David, and D. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999. ISBN 0387987673.

A.S. Fast. *Learning the Structure of Bayesian Networks with Constraint Satisfaction*. PhD thesis, University of Massachusetts Amherst, Amherst, MA, USA, 2010.

N. Friedman. Inferring cellular networks using probabilistic graphical models. *Science Magazine*, 303(5659):799–805, 2004.

N. Friedman, I. Nachman, and D. Peer. Learning Bayesian network structure from massive datasets: The sparse candidate algorithm. In *5th Conference on Uncertainty in Artificial Intelligence*, pages 206–215, 1999.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, pages 601–620, 2000. doi: 10.

H. Hotelling. New light on the correlation coefficient and its transforms. *Journal of the Royal Statistical Society. Series B (Methodological)*, 15(2):193–232, 1953.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007. ISSN 1532-4435.

M. Kalisch, M. Maechler, and D. Colombo. *PCAlg: Estimation of CPDAG/PAG and causal inference using the IDA algorithm*, 2010. URL `http://CRAN.R-project.org/package=pcalg`. R package version 1.0-2.

W. Lam and F. Bacchus. Learning Bayesian belief networks: An approach based on the mdl principle. *Computational Intelligence*, 10:269–293, 1994.

J. Li and Z.J. Wang. Controlling the false discovery rate of the association/causality structure learned with the PC algorithm. *The Journal of Machine Learning Research*, 10:475–514, 2009.

D. Margaritis and S. Thrun. A Bayesian multiresolution independence test for continuous variables. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 346–353. Morgan Kaufmann Publishers Inc., 2001.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence*, pages 403–441, San Francisco, CA, USA, 1995. Morgan Kaufmann.

R. Neapolitan. *Learning Bayesian Networks*. Pearson Printice Hall, London, UK, 2004.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 1558604790.

E. Perrier, S. Imoto, and S. Miyano. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research*, 9:2251–2286, 2008.

S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice hall, Englewood Cliffs, NJ, USA, 2009.

J. Schäfer and K. Strimmer. An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754, 2005.

M. Scutari. Learning Bayesian networks with the BNlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer, New York, NY, USA, 1993.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search, 2nd Edition*, volume 1. The MIT Press, Cambridge, MI, USA, 2001.

I. Tsamardinos and L.E. Brown. Bounding the false discovery rate in local Bayesian network learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence*, pages 1100–1105, 2008.

I. Tsamardinos, L. Brown, and C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 10 2006. ISSN 1532-4435.

T.S. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 1990.

J. Zhang and P. Spirtes. Strong faithfulness and uniform consistency in causal inference. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, pages 632–639. Citeseer, 2003.

Journal of Machine Learning Research 14 (2012) 1605-1626 Submitted 6/12; Revised 4/13; Published 6/13

# Dimension Independent Similarity Computation

**Reza Bosagh Zadeh** REZAB@STANFORD.EDU
*Institute for Computational and Mathematical Engineering*
*Stanford University*
*Stanford, CA 94305, USA*

**Ashish Goel** ASHISHG@STANFORD.EDU
*Department of Management Science and Engineering*
*Stanford University*
*Stanford, CA 94305, USA*

**Editor:** Inderjit Dhillon

## Abstract

We present a suite of algorithms for Dimension Independent Similarity Computation (DISCO) to compute all pairwise similarities between very high-dimensional sparse vectors. All of our results are provably independent of dimension, meaning that apart from the initial cost of trivially reading in the data, all subsequent operations are independent of the dimension; thus the dimension can be very large. We study Cosine, Dice, Overlap, and the Jaccard similarity measures. For Jaccard similarity we include an improved version of MinHash. Our results are geared toward the MapReduce framework. We empirically validate our theorems with large scale experiments using data from the social networking site Twitter. At time of writing, our algorithms are live in production at twitter.com.

**Keywords:** cosine, Jaccard, overlap, dice, similarity, MapReduce, dimension independent

## 1. Introduction

Computing similarity between all pairs of vectors in large-scale data sets is a challenge. Traditional approaches of sampling the data set are limited and linearly dependent on the dimension of the data. We present an approach whose complexity is independent of the data dimension and geared towards modern distributed systems, in particular the MapReduce framework (Dean and Ghemawat, 2008).

MapReduce is a programming model for processing large data sets, typically used to do distributed computing on clusters of commodity computers. With large amount of processing power at hand, it is very tempting to solve problems by brute force. However, we show how to combine clever sampling techniques with the power of MapReduce to extend its utility.

Consider the problem of finding all pairs of similarities between $D$ indicator (0/1 entries) vectors, each of dimension $N$. In particular we focus on cosine similarities between all pairs of $D$ vectors in $\mathbb{R}^N$. Further assume that each dimension is $L$-sparse, meaning each dimension has at most $L$ non-zeros across all points. For example, typical values to compute similarities between all pairs of a subset of Twitter users can be: $N = 10^9$ (the universe of Twitter users), $D = 10^7$ (a subset of Twitter users), $L = 1000$. To be more specific: each of $D$ users can be represented by an $N$ dimensional vector that indicates which of the $N$ universal users are followed. The sparsity parameter ($L = 1000$) here assumes that each user follows at most 1000 other users.

©2012 Reza Bosagh Zadeh and Ashish Goel.

There are two main complexity measures for MapReduce: "shuffle size" and "reduce-key complexity", defined shortly (Goel and Munagala, 2012). It can be easily shown that the naive approach for computing similarities will have $O(NL^2)$ emissions, which for the example parameters we gave is infeasible. The number of emissions in the map phase is called the "shuffle size", since that data needs to be shuffled around the network to reach the correct reducer. Furthermore, the maximum number of items reduced to a single key can be as large as $N$. Thus the "reduce-key complexity" for the naive scheme is $N$.

We can drastically reduce the shuffle size and reduce-key complexity by some clever sampling with the DISCO scheme described in this paper. In this case, the output of the reducers are random variables whose expectations are the similarities. Two proofs are needed to justify the effectiveness of this scheme: first, that the expectations are indeed correct and obtained with high probability, and second, that the shuffle size is greatly reduced. We prove both of these claims. In particular, in addition to correctness, we prove that to estimate similarities above $\varepsilon$, the shuffle size of the DISCO scheme is only $O(DL\log(D)/\varepsilon)$, with no dependence on the dimension $N$, hence the name.

This means as long as there are enough mappers to read the data, one can use the DISCO sampling scheme to make the shuffle size tractable. Furthermore, each reduce key gets at most $O(\log(D)/\varepsilon)$ values, thus making the reduce-key complexity tractable too. Within Twitter inc, we use the DISCO sampling scheme to compute similar users. We have also used the scheme to find highly similar pairs of words by taking each dimension to be the indicator vector that signals in which tweets the word appears. We empirically verify the proven claims and observe large reductions in shuffle size in this paper.

Our sampling scheme can be used to implement many other similarity measures. We focus on four similarity measures: Cosine, Dice, Overlap, and the Jaccard similarity measures. For Jaccard similarity we present an improved version of the well known MinHash scheme (Broder, 1997). Our framework operates under the following assumptions, each of which we justify:

First, we focus on the case where each dimension is sparse and therefore the natural way to store the data is by segmenting it into dimensions. In our application each dimension is represented by a tweet, thus this assumption was natural. Second, our sampling scheme requires a "background model", meaning the magnitude of each vector is assumed to be known and loaded into memory. In our application this was not a hurdle, since the magnitudes of vectors in our corpus needed to be computed for other tasks. In general this may require another pass over the data set. To further address the issue, in the streaming computation model, we can remove the dependence by paying an extra logarithmic factor in memory used. Third, we prove results on highly similar pairs, since common applications require thresholding the similarity score with a high threshold value.

A ubiquitous problem is finding all pairs of objects that are in some sense similar and in particular more similar than a threshold. For such applications of similarity, DISCO is particularly helpful since higher similarity pairs are estimated with provably better accuracy. There are many examples, including

- Advertiser keyword suggestions: When targeting advertisements via keywords, it is useful to expand the manually input set of keywords by other similar keywords, requiring finding all keywords more similar than a high threshold (Regelson and Fain, 2006). The vector representing a keyword will often be an indicator vector indicating in which dimensions the keywords appear.

- Collaborative filtering: Collaborative filtering applications require knowing which users have similar interests. For this reason, given a person or user, it is required to find all other objects more similar than a particular threshold, for which our algorithms are well suited.

- Web Search: Rewriting queries given to a search engine is a common trick used to expand the coverage of the search results (Abhishek and Hosanagar, 2007). Adding clusters of similar queries to a given query is a natural query expansion strategy.

These applications have been around for many years, but the scale at which we need to solve them keeps steadily increasing. This is particularly true with the rise of the web, and social networks such as Twitter, Facebook, and Google Plus among others. Google currently holds an index of more than 1 trillion webpages, on which duplicate detection must be a daunting task. For our experiments we use the Twitter data set, where the number of tweets in a single day surpasses 200 million. Many of the applications, including ours involve domains where the dimensionality of the data far exceeds the number of points. In our case the dimensionality is large (more than 200 million), and we can prove that apart from initially reading the data (which cannot be avoided), subsequent MapReduce operations will have input size independent of the number of dimensions. Therefore, as long as the data can be read in, we can ignore the number of dimensions.

A common technique to deal with large data sets is to simply sample. Indeed, our approach too, is a sampling scheme; however, we sample in a nontrivial way so that points that have nonzero entries in many dimensions will be sampled with lower probability than points that are only present in a few dimensions. Using this idea, we can remove the dependence on dimensionality while being able to mathematically prove—and empirically verify—accuracy.

Although we use the MapReduce (Dean and Ghemawat, 2008) framework and discuss shuffle size, the sampling strategy we use can be generalized to other frameworks. We focus on MapReduce because it is the tool of choice for large scale computations at Twitter, Google, and many other companies dealing with large scale data. However, the DISCO sampling strategy is potentially useful whenever one is computing a number between 0 and 1 by taking the ratio of an unknown number (the dot product in our case) by some known number (e.g., the magnitude). This is a high-level description, and we give four concrete examples, along with proofs, and experiments. Furthermore, DISCO improves the implementation of the well-known MinHash (Broder, 1997) scheme in any MapReduce setup.

## 2. Formal Preliminaries

Let $T = \{t_1, \ldots, t_N\}$ represent $N$ documents, each of a length of no more than $L$ words. In our context, the documents are tweets (documents are also the same as dimensions for us) from the social networking site Twitter, or could be fixed-window contexts from a large corpus of documents, or any other data set where dimensions have no more than $L$ nonzero entries and the data set is available dimension-by-dimension. We are interested in similarity scores between pairs of words in a dictionary containing $D$ words $\{w_1, \ldots, w_D\}$. The number of documents in which two words $w_i$ and $w_j$ co-occur is denoted $\#(w_i, w_j)$. The number of documents in which a single word $w_i$ occurs is denoted $\#(w_i)$.

To each word in the dictionary, an $N$-dimensional indicator vector is assigned, indicating in which documents the word occurs. We operate within a MapReduce framework where each document is an input record. We denote the number of items output by the map phase as the 'shuffle size'.

Our algorithms will have shuffle sizes that provably do not depend on $N$, making them particularly attractive for very high dimensional tasks.

We focus on 5 different similarity measures, including cosine similarity, which is very popular and produces high quality results across different domains (Chien and Immorlica, 2005; Chuang and Chien, 2005; Sahami and Heilman, 2006; Spertus et al., 2005). Cosine similarity is simply the vector normalized dot product: $\frac{\#(x,y)}{\sqrt{\#(x)}\sqrt{\#(y)}}$ where $\#(x) = \sum_{i=1}^{N} x[i]$ and $\#(x,y) = \sum_{i=1}^{N} x[i]y[i]$. In addition to cosine similarity, we consider many variations of similarity scores that use the dot product. They are outlined in Table 1.

To compare the performance of algorithms in a MapReduce framework, we report and analyze shuffle size, which is more reliable than wall time or any other implementation-specific measure. We define shuffle size as the total output of the Map Phase, which is what will need to be "shuffled" before the Reduce phase can begin. Our results and theorems hold across any MapReduce implementation such as Hadoop (Borthakur, 2007; Gates et al., 2009) or Google's MapReduce (Dean and Ghemawat, 2008). We focus on shuffle size because after one trivially reads in the data via mappers, the shuffle phase is the bottleneck since our mappers and reducers are all linear in their input size. In general, MapReduce algorithms are usually judged by two performance measures: largest reduce bucket and shuffle size. Since the largest reduce bucket is not at all a problem for us, we focus on shuffle size.

For many applications, and our applications in particular, input vectors are sparse, meaning that the large majority of entries are 0. A sparse vector representation for a vector $x$ is the set of all pairs $(i,x[i])$ such that $x[i] > 0$ over all $i = 1 \ldots N$. The size of a vector $x$, which we denote as $\#(x)$ is the number of such pairs. We focus on the case where each *dimension* is sparse and therefore the natural way to store the data is segmented into dimensions.

Throughout the paper we formally prove results for pairs more similar than a threshold, called $\varepsilon$. Our algorithms will often have a tradeoff between accuracy and shuffle size, where the tradeoff parameter is $p$. By design, the larger $p$ becomes, the more accurate the estimates and the larger the shuffle size becomes.

We assume that the dictionary can fit into memory but that the number of dimensions (documents) is so large that many machines will be needed to even hold the documents on disk. Note that documents and dimensions are the same thing, and we will use these terms interchangeably. We also assume that the magnitudes of the dimensions are known and available in all mappers and reducers.

| Similarity | Definition | Shuffle Size | Reduce-key size |
|---|---|---|---|
| Cosine | $\frac{\#(x,y)}{\sqrt{\#(x)}\sqrt{\#(y)}}$ | $O(DL\log(D)/\varepsilon)$ | $O(\log(D)/\varepsilon)$ |
| Jaccard | $\frac{\#(x,y)}{\#(x)+\#(y)-\#(x,y)}$ | $O((D/\varepsilon)\log(D/\varepsilon))$ | $O(\log(D/\varepsilon)/\varepsilon)$ |
| Overlap | $\frac{\#(x,y)}{\min(\#(x),\#(y))}$ | $O(DL\log(D)/\varepsilon)$ | $O(\log(D)/\varepsilon)$ |
| Dice | $\frac{2\#(x,y)}{\#(x)+\#(y)}$ | $O(DL\log(D)/\varepsilon)$ | $O(\log(D)/\varepsilon)$ |

Table 1: Similarity measures and the bounds we obtain. All sizes are independent of $N$, the dimension. These are bounds for shuffle size without combining. Combining can only bring down these sizes.

We are interested in several similarity measures outlined in Table 1. For each, we prove the shuffle size for the bottleneck step of the pipeline needed to compute all nonzero scores. Our goal is to compute similarity scores between all pairs of words, and prove accuracy results for pairs which have similarity above $\varepsilon$. The naive approach is to first compute the dot product between all pairs of words in a Map-Reduce (Dean and Ghemawat, 2008) style framework. Mappers act on each document, and emit key-value pairs of the form $(w_i, w_j) \to 1$. These pairs are collected with the sum reducer, which gives the dot product $\#(w_i, w_j)$. The product is then used to trivially compute the formulas in Table 1. The difficulty of computing the similarity scores lies almost entirely in computing $\#(w_i, w_j)$.

Shuffle size is defined as the number of key-value pairs emitted. The shuffle size for the above naive approach is $N\binom{L}{2} = O(NL^2)$, which can be prohibitive for $N$ too large. Our algorithms remove the dependence on $N$ by dampening the number of times popular words are emitted as part of a pair (e.g., see algorithm 5). Instead of emitting every pair, only some pairs are output, and instead of computing intermediary dot products, we directly compute the similarity score.

## 3. Complexity Measures

An in-depth discussion of complexity measures for MapReduce is available Goel and Munagala (2012). We focus on shuffle size and reduce-key complexity for two reasons. First, focusing on reduce key complexity captures the performance of an idealized Map-Reduce system with infinitely many mappers and reducers and no coordination overheads. A small reduce-key complexity guarantees that a Map-Reduce algorithm will not suffer from "the curse of the last reducer" (Suri and Vassilvitskii, 2011), a phenomenon in which the average work done by all reducers may be small, but due to variation in the size of reduce records, the total wall clock time may be extremely large or, even worse, some reducers may run out of memory.

Second, we focus on shuffle size since the total size of input/output by all mappers/reducers captures the total file I/O done by the algorithm, and is often of the order of the shuffle size.

Lastly, our complexity measures depend only on the algorithm, not on details of the Map-Reduce installation such as the number of machines, the number of Mappers/Reducers etc., which is a desirable property for the analysis of algorithms (Goel and Munagala, 2012).

## 4. Related Work

Previously the all-pairs similarity search problem has been studied (Broder et al., 1997; Broder, 1997), in the context of identifying near-duplicate web pages. We describe MinHash later in Section 5.2. We improve the vanilla implementation of MinHash on MapReduce to arrive at better shuffle sizes without (effectively) any loss in accuracy. We prove these results in Section 5.2, and verify them experimentally.

There are many papers discussing the all pairs similarity computation problem. In other work (Elsayed et al., 2008; Campagna and Pagh, 2012), the MapReduce framework is targeted, but there is still a dependence on the dimensionality. Other authors (Lin, 2009) have considered computing all pairs on MapReduce, but there is a focus on the life sciences domain. The all-pairs similarity search problem has also been addressed in the database community, where it is known as the similarity join problem (Arasu et al., 2006; Chaudhuri et al., 2006; Sarawagi and Kirpal, 2004). These papers are loosely assigned to one of two categories: First, signatures of the points are obtained to convert the nonexact matching problem to an exact matching problem, followed by hash-map and filtering false positives. Second, inverted list based solutions exploit information retrieval techniques.

In another paper Baraglia et al. (2010), the authors consider the problem of finding highly similar pairs of documents in MapReduce. They discuss shuffle size briefly but do not provide proven guarantees. They also discuss reduce key size, but again do not provide bounds on the size.

There is large body of work on the nearest neighbors problem, which is the problem of finding the $k$ nearest neighbors of a given query point (Charikar, 2002; Fagin et al., 2003; Gionis et al., 1999; Indyk and Motwani, 1998). Our problem, the all-pairs similarities computation, is related, but not identical to the nearest neighbor problem. As one of the reviewers astutely pointed out, when only the $k$ nearest neighbors need to be found, optimizations can be made for that. In particular, there are two problems with pairwise similarity computation, large computation time with respect to dimensions, which we address, and large computation time with respect to the number of points.

Some researchers (Pantel et al., 2009) proposed a highly scalable term similarity algorithm, implemented in the MapReduce framework, and deployed an over 200 billion word crawl of the Web to compute pairwise similarities between terms. Their results are still dependent upon the dimension and they only provide experimental evaluations of their algorithms.

Other related work includes clustering of web data (Beeferman and Berger, 2000; Chien and Immorlica, 2005; Sahami and Heilman, 2006; Spertus et al., 2005). These applications of clustering typically employ relatively straightforward exact algorithms or approximation methods for computing similarity. Our work could be leveraged by these applications for improved performance or higher accuracy through reliance on proven results. We have also used DISCO in finding similar users in a production environment Gupta et al. (2013).

## 5. Algorithms and Methods

The Naive algorithm for computing similarities first computes dot products, then simply divides by whatever is necessary to obtain a similarity score. I.e. in a MapReduce implementation:

1. Given document $t$, Map using NaiveMapper (Algorithm 1)

2. Reduce using the NaiveReducer (Algorithm 2)

**Algorithm 1** NaiveMapper($t$)

**for** all pairs $(w_1, w_2)$ in $t$ **do**
  emit $((w_1, w_2) \to 1)$
**end for**

**Algorithm 2** NaiveReducer($(w_1, w_2), \langle r_1, \ldots, r_R \rangle$)

$a = \sum_{i=1}^{R} r_i$
output $\frac{a}{\sqrt{\#(w_1)\#(w_2)}}$

The above steps will compute all dot products, which will then be scaled by appropriate factors for each of the similarity scores. Instead of using the Naive algorithm, we modify the mapper of the naive algorithm and replace it with Algorithm 3, and replace the reducer with Algorithm 4 to directly compute the actual similarity score, rather than the dot products. The following sections detail how to obtain dimensionality independence for each of the similarity scores.

**Algorithm 3** DISCOMapper($t$)

**for** all pairs $(w_1, w_2)$ in $t$ **do**
  emit using custom Emit function
**end for**

**Algorithm 4** DISCOReducer($(w_1, w_2), \langle r_1, \ldots, r_R \rangle$)

$a = \sum_{i=1}^{R} r_i$
output $a\frac{\varepsilon}{p}$

### 5.1 Cosine Similarity

To remove the dependence on $N$, we replace the emit function with Algorithm 5.

**Algorithm 5** CosineSampleEmit($w_1, w_2$) - $p/\varepsilon$ is the oversampling parameter

With probability

$$\frac{p}{\varepsilon}\frac{1}{\sqrt{\#(w_1)}\sqrt{\#(w_2)}}$$

emit $((w_1, w_2) \to 1)$

Note that the more popular a word is, the less likely it is to be output. This is the key observation leading to a shuffle size that is independent of the dimension. We use a slightly different reducer, which instead of calculating $\#(w_1, w_2)$, computes $\cos(w_1, w_2)$ directly without any intermediate steps. The exact estimator is given in the proof of Theorem 1.

Since the CosineSampleEmit algorithm is only guaranteed to produce the correct similarity score in expectation, we must show that the expected value is highly likely to be obtained. This guarantee is given in Theorem 1.

**Theorem 1** *For any two words $x$ and $y$ having $\cos(x,y) \geq \varepsilon$, let $X_1, X_2, \ldots, X_{\#(x,y)}$ represent indicators for the coin flip in calls to CosineSampleEmit with $x, y$ parameters, and let $X = \sum_{i=1}^{\#(x,y)} X_i$. For any $1 > \delta > 0$, we have*

$$\Pr\left[\frac{\varepsilon}{p}X > (1+\delta)\cos(x,y)\right] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{p}$$

*and*

$$\Pr\left[\frac{\varepsilon}{p}X < (1-\delta)\cos(x,y)\right] < \exp(-p\delta^2/2)$$

**Proof** We use $\frac{\varepsilon}{p}X$ as the estimator for $\cos(x,y)$. Note that

$$\mu_{xy} = E[X] = \#(x,y)\frac{p}{\varepsilon}\frac{1}{\sqrt{\#(x)}\sqrt{\#(y)}} = \frac{p}{\varepsilon}\cos(x,y) \geq p$$

Thus by the multiplicative form of the Chernoff bound,

$$\Pr\left[\frac{\varepsilon}{p}X > (1+\delta)\cos(x,y)\right] = \Pr\left[\frac{\varepsilon}{p}X > (1+\delta)\frac{\varepsilon}{p}\frac{p}{\varepsilon}\cos(x,y)\right]$$

$$= \Pr[X > (1+\delta)\mu_{xy}] < \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu_{xy}} \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{p}$$

Similarly, by the other side of the multiplicative Chernoff bound, we have

$$\Pr\left[\frac{\varepsilon}{p}X < (1-\delta)\cos(x,y)\right] = \Pr[X < (1-\delta)\mu_{xy}] < \exp(-\mu_{xy}\delta^2/2) \leq \exp(-p\delta^2/2)$$

■

Since there are $\binom{D}{2}$ pairs of words in the dictionary, set $p = \log(D^2) = 2\log(D)$ and use union bound with theorem 1 to ensure the above bounds hold simultaneously for all pairs $x, y$ having $\cos(x,y) \geq \varepsilon$. It is worth mentioning that $\left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)$ and $\exp(-\delta^2/2)$ are always less than 1; thus raising them to the power of $p$ brings them down exponentially in $p$. Note that although we decouple $p$ and $\varepsilon$ for the analysis of shuffle size, in a good implementation they are tightly coupled, and the expression $p/\varepsilon$ can be thought of as a single parameter to trade off accuracy and shuffle size. In a large scale experiment described in section 9.1 we set $p/\varepsilon$ in a coupled manner.

Now we show that the shuffle size is independent of $N$, which is a great improvement over the naive approach when $N$ is large. To see the usefulness of these bounds, it is worth noting that we prove them almost optimal (i.e., no other algorithm can do much better). In Theorem 2 we prove that any algorithm that purports to accurately calculate highly similar pairs must at least output them, and sometimes there are at least $DL$ such pairs, and so any algorithm that is accurate on highly similar pairs must have at least $DL$ shuffle size. We are off optimal here by only a $\log(D)/\varepsilon$ factor.

**Theorem 2** *The expected shuffle size for CosineSampleEmit is $O(DL\log(D)/\varepsilon)$ and $\Omega(DL)$.*

**Proof** The expected contribution from each pair of words will constitute the shuffle size:

$$\sum_{i=1}^{D}\sum_{j=i+1}^{D}\sum_{k=1}^{\#(w_i,w_j)}\Pr[\text{CosineSampleEmit}(w_i,w_j)]$$

$$=\sum_{i=1}^{D}\sum_{j=i+1}^{D}\#(w_i,w_j)\Pr[\text{CosineSampleEmit}(w_i,w_j)]$$

$$=\sum_{i=1}^{D}\sum_{j=i+1}^{D}\frac{p}{\varepsilon}\frac{\#(w_i,w_j)}{\sqrt{\#(w_i)}\sqrt{\#(w_j)}}$$

$$\leq\frac{p}{2\varepsilon}\sum_{i=1}^{D}\sum_{j=i+1}^{D}\#(w_i,w_j)(\frac{1}{\#(w_i)}+\frac{1}{\#(w_j)})$$

$$\leq\frac{p}{\varepsilon}\sum_{i=1}^{D}\frac{1}{\#(w_i)}\sum_{j=1}^{D}\#(w_i,w_j)$$

$$\leq\frac{p}{\varepsilon}\sum_{i=1}^{D}\frac{1}{\#(w_i)}L\#(w_i)=\frac{p}{\varepsilon}LD=O(DL\log(D)/\varepsilon).$$

The first inequality holds because of the Arithmetic-Mean-Geometric-Mean inequality applied to $\{1/\#(w_i),1/\#(w_j)\}$. The last inequality holds because $w_i$ can co-occur with at most $\#(w_i)L$ other words. It is easy to see via Chernoff bounds that the above shuffle size is obtained with high probability.

To see the lower bound, we construct a data set consisting of $D/L$ distinct documents of length $L$; furthermore each document is duplicated $L$ times. To construct this data set, consider grouping the dictionary into $D/L$ groups, each group containing $L$ words. A document is associated with every group, consisting of all the words in the group. This document is then repeated $L$ times. In each group, it is trivial to check that all pairs of words of have similarity exactly 1. There are $\binom{L}{2}$ pairs for each group and there are $D/L$ groups, making for a total of $(D/L)\binom{L}{2}=\Omega(DL)$ pairs with similarity 1, and thus also at least $\varepsilon$. Since any algorithm that purports to accurately calculate highly-similar pairs must at least *output* them, and there are $\Omega(DL)$ such pairs, we have the lower bound. ■

It is important to observe what happens if the output "probability" is greater than 1. We certainly Emit, but when the output probability is greater than 1, care must be taken during reducing to scale by the correct factor, since it won't be correct to divide by $p/\varepsilon$, which is the usual case when the output probability is less than 1. Instead, we must divide by $\sqrt{\#(w_1)}\sqrt{\#(w_2)}$ because for the pairs where the output probability is greater than 1, CosineSampleEmit and Emit are the same. Similar corrections have to be made for the other similarity scores (Dice and Overlap, but not MinHash), so we do not repeat this point. Nonetheless it is an important one which arises during implementation.

Finally it is easy to see that the largest reduce key will have at most $p/\varepsilon=O(\log(D)/\varepsilon)$ values.

**Theorem 3** *The expected number of values mapped to a single key by DISCOMapper is* $p/\varepsilon$.

**Proof** Note that the output of DISCOReducer is a number between 0 and 1. Since this is obtained by normalizing the sum of all values reduced to the key by $p/\varepsilon$, and all summands are at most 1, we trivially get that the number of summands is at most $p/\varepsilon$. ■

### 5.2 Jaccard Similarity

Traditionally MinHash (Broder, 1997) is used to compute Jaccard similarity scores between all pairs in a dictionary. We improve the MinHash scheme to run much more efficiently with a smaller shuffle size.

Let $h(t)$ be a hash function that maps documents to distinct numbers in $[0,1]$, and for any word $w$ define $g(w)$ (called the MinHash of $w$) to be the minimum value of $h(t)$ over all $t$ that contain $w$. Then $g(w_1) = g(w_2)$ exactly when the minimum hash value of the union with size $\#(w_1)+\#(w_2)-\#(w_1,w_2)$ lies in the intersection with size $\#(w_1,w_2)$. Thus

$$\Pr[g(w_1) = g(w_2)] = \frac{\#(w_1,w_2)}{\#(w_1)+\#(w_2)-\#(w_1,w_2)} = \text{Jac}(w_1,w_2)$$

Therefore the indicator random variable that is 1 when $g(w_1) = g(w_2)$ has expectation equal to the Jaccard similarity between the two words. Unfortunately the variable has too high a variance to be useful on its own. The idea of the MinHash scheme is to reduce the variance by averaging together $k$ of these variables constructed in the same way with $k$ different hash functions. We index these $k$ functions using the notation $g_j(w)$ to denote the MinHash of hash function $h_j(t)$. We denote the computation of hashes as 'MinHashMap'. Specifically, MinHashMap is defined as Algorithm 6.

To estimate $\text{Jac}(w_1,w_2)$ using this version of the scheme, we simply count the number of hash functions for which $g(w_1) = g(w_2)$, and divide by $k$ to get an estimate of $\text{Jac}(w_1,w_2)$. By the multiplicative Chernoff bound for sums of 0-1 random variables as seen in Theorem 1, setting $k = O(1/\varepsilon)$ will ensure that w.h.p. a similarity score that is above $\varepsilon$ has a relative error of no more than $\delta$. Qualitatively, this theorem is the same as that given in the original MinHash paper (Broder, 1997) (where MinHash is introduced), and we do not claim the following as a new contribution; however, we include it for completeness. More rigorously,

**Theorem 4** *Fix any two words $x$ and $y$ having $Jac(x,y) \geq \varepsilon$. Let $X_1, X_2, \ldots, X_k$ represent indicators for $\{g_1(x) = g_1(y), \ldots, g_k(x) = g_k(y)\}$ and $X = \sum_{i=1}^{k} X_i$. For any $1 > \delta > 0$ and $k = c/\varepsilon$, we have*

$$\Pr[X/k > (1+\delta)Jac(x,y)] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{c}$$

*and*

$$\Pr[X/k < (1-\delta)Jac(x,y)] \leq \exp(-c\delta^2/2)$$

**Proof** We use $X/k$ as the estimator for $\text{Jac}(x,y)$. Note that $E[X] = k\text{Jac}(x,y) = (c/\varepsilon)\text{Jac}(x,y) \geq c$. Now by standard Chernoff bounds we have

$$\Pr[X/k > (1+\delta)\text{Jac}(x,y)] = \Pr[X > (1+\delta)E[X]] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{E[X]} \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{c}$$

Similarly, by the other side of the multiplicative Chernoff bound, we have

$$\Pr[X/k < (1-\delta)\text{Jac}(x,y)] \leq \exp(-c\delta^2/2)$$

■

The MapReduce implementation of the above scheme takes in documents and for each unique word in the document outputs $k$ hash values.

**Algorithm 6** MinHashMap$(t, \langle w_1, \ldots, w_L \rangle)$

**for** $i = 1$ to $L$ **do**
  **for** $j = 1$ to $k$ **do**
    emit $((w_i, j) \to h_j(t))$
  **end for**
**end for**

1. Given document $t$, Map using MinHashMap (Algorithm 6)
2. Reduce using the min reducer
3. Now that minhash values for all hash functions are available, for each pair we can simply compute the number of hash collisions and divide by the total number of hash functions

**Algorithm 7** MinHashSampleMap$(t, \langle w_1, \ldots, w_L \rangle)$

**for** $i = 1$ to $L$ **do**
  **for** $j = 1$ to $k$ **do**
    **if** $h_j(t) \leq \frac{c \log(Dk)}{\#(w_i)}$ **then**
      emit $((w_i, j) \to h_j(t))$
    **end if**
  **end for**
**end for**

Recall that a document has at most $L$ words. This naive Mapper will have shuffle size $NLk = O(NL/\varepsilon)$, which can be improved upon. After the map phase, for each of the $k$ hash functions, the standard MinReducer is used, which will compute $g_j(w)$. These MinHash values are then simply checked for equality. We modify the initial map phase, and prove that the modification brings down shuffle size while maintaining correctness. The modification is seen in algorithm 7, note that $c$ is a small constant we take to be 3.

We now prove that MinHashSampleMap will with high probability Emit the minimum hash value for a given word $w$ and hash function $h$, thus ensuring the steps following MinHashSampleMap will be unaffected.

**Theorem 5** *If the hash functions $h_1, \ldots, h_k$ map documents to $[0,1]$ uniform randomly, and $c = 3$, then with probability at least $1 - \frac{1}{(Dk)^2}$, for all words $w$ and hash functions $h \in \{h_1, \ldots, h_k\}$, MinHashSampleMap will emit the document that realizes $g(w)$.*

**Proof** Fix a word $w$ and hash function $h$ and let $z = \min_{t|w \in t} h(t)$. Now the probability that MinHashSampleMap will not emit the document that realizes $g(w)$ is

$$\Pr\left[z > \frac{c \log(Dk)}{\#(w)}\right] = \left(1 - \frac{c \log(Dk)}{\#(w)}\right)^{\#(w)} \leq e^{-c \log(Dk)} = \frac{1}{(Dk)^c}$$

Thus for a single $w$ and $h$ we have shown MinHashSampleMap will w.h.p. emit the hash that realizes the MinHash. To show the same result for *all* hash functions and words in the dictionary, set $c = 3$ and use union bound to get a $(\frac{1}{Dk})^2$ bound on the probability of error for *any* $w_1, \ldots, w_D$ and $h_1, \ldots, h_k$. ■

Now that we have correctness via theorems 4 and 5, we move on to calculating the shuffle size for MinHashSampleMap.

**Theorem 6** *MinHashSampleMap has expected shuffle size* $O(Dk\log(Dk)) = O((D/\varepsilon)\log(D/\varepsilon))$.

**Proof** Simply adding up the expectations for the emissions indicator variables, we see the shuffle size is bounded by

$$\sum_{h\in\{h_1,\ldots,h_k\}} \sum_{w\in\{w_1,\ldots,w_D\}} \frac{c\log(Dk)}{\#(w)}\#(w) = Dkc\log(Dk)$$

Setting $c = 3$ and $k = 1/\varepsilon$ gives the desired bound. ■

All of the reducers used in our algorithms are associative and commutative operations (sum and min), and therefore can be combined for optimization. Our results do not change qualitatively when combiners are used, except for one case. A subtle point arises in our claim for improving MinHash. If we combine with $m$ mappers, then the naive MinHash implementation will have a shuffle size of $O(mDk)$ whereas DISCO provides a shuffle size of $O(Dk\log(Dk))$. Since $m$ is usually set to be a very large constant (one can easily use 10,000 mappers in a standard Hadoop implementation), removing the dependence on $m$ is beneficial. For the other similarity measures, combining the Naive mappers can only bring down the shuffle size to $O(mD^2)$, which DISCO improves upon asymptotically by obtaining a bound of $O(DL/\varepsilon\log(D))$ without even combining, so combining will help even more. In practice, DISCO can be easily combined, ensuring superior performance both theoretically and empirically.

## 6. Cosine Similarity in Streaming Model

We briefly depart from the MapReduce framework and instead work in the "Streaming" framework. In this setting, data is streamed dimension-by-dimension through a single machine that can at any time answer queries of the form "what is the similarity between points $x$ and $y$ considering all the input so far?". The main performance measure is how much memory the machine uses and queries will be answered in constant time. We describe the algorithm only for cosine similarity, but an almost identical algorithm will work for dice, and overlap similarity.

Our algorithm will be very similar to the mapreduce setup, but in place of emitting pairs to be shuffled for a reduce phase, we instead insert them into a hash map $H$, keyed by pairs of words, with each entry holding a bag of emissions. $H$ is used to track the emissions by storing them in a bag associated with the pair. Since all data streams through a single machine, for any word $x$, we can keep a counter for $\#(x)$. This will take $D$ counters worth of memory, but as we see in Theorem 1 this memory usage will be dominated by the size of $H$. Each emission is decorated with the probability of emission. There are two operations to be described: the update that occurs when a new dimension (document) arrives, and the constant time algorithm used to answer similarity queries.

*Update.* On an update we are given a document. For each pair of words $x, y$ in the document, with independent coin flips of probability $q = \frac{p}{\varepsilon}\frac{1}{\sqrt{\#(x)\#(y)}}$ we look up the bag associated with $(x,y)$ in $H$ and insert $q$ into it. It is important to note that the emission is being done with probability $q$ and $q$ is computed using the *current values* of $\#(x)$ and $\#(y)$. Thus if a query comes after this update, we must take into account all new information. This is done via subsampling and will be explained shortly. It remains to show how to use these probabilities to answer queries.

*Query.* We now describe how to answer the only query. Let the query be for the similarity between words $x$ and $y$. Recall at this point we know both $\#(x)$ and $\#(y)$ exactly, for the data seen so far. We look up the pair $(x,y)$ in $H$, and grab the associated bag of emissions. Recall from above that emission is decorated with the probability of emission $q_i$ for the $i$'th entry in the bag. Unfortunately $q_i$ will be larger than we need it to be, since it was computed at a previous time, when there had been fewer occurrences of $x$ and $y$. To remedy this, we independently subsample each of the emissions for the pair $x, y$ with coin flips of probability $\frac{p}{\varepsilon}\frac{1}{q_i\sqrt{\#(x)\#(y)}}$. For each pair $x, y$ seen in the input, there will be exactly

$$q_i \frac{p}{\varepsilon}\frac{1}{q_i\sqrt{\#(x)\#(y)}} = \frac{p}{\varepsilon}\frac{1}{\sqrt{\#(x)\#(y)}}$$

probability of surviving emission *and* the subsampling. Finally, since the pair $x, y$ is seen exactly $\#(x,y)$ times, the same estimator used in Theorem 2 will have expectation equal to $\cos(x,y)$. Furthermore, since before subsampling we output in expectation more pairs than CosineSampleEmit, the Chernoff bound of Theorem 2 still holds. Finally, to show that $H$ cannot grow too large, we bound its size in Theorem 7.

**Theorem 7** *The streaming algorithm uses at most $O(DL\lg(N)\log(D)/\varepsilon)$ memory.*

**Proof** We only need to bound the size of $H$. Consider a word $x$ and all of its occurrences in documents $t_1, \ldots, t_{\#(x)}$ at final time (i.e., after all $N$ documents have been processed). We conceptually and only for this analysis construct a new larger data set $C'$ where each word $x$ is removed and its occurrences are replaced *in order* with $\lfloor \lg \#(x) \rfloor + 1$ new words $x_1, \ldots, x_{\lfloor \lg \#(x) \rfloor + 1}$, so we are effectively segmenting (in time) the occurrences of $x$. With this in mind, we construct $C'$ so that each $x_i$ will replace $2^{i-1}$ occurrences of $x$, in time order. That is, we will have $\#(x_i) = 2^{i-1}$.

Note that our streaming algorithm updates the counters for words with every update. Consider what happens if instead of updating every time, the streaming algorithm somehow in advance knew and used the final $\#(x)$ values after all documents have been processed. We call this the "all-knowing'" version. The size of $H$ for such an all-knowing algorithm is the same as the shuffle size for the DISCO sampling scheme analyzed in Theorem 2, simply because there is a bijection between the emits of CosineSampleEmit and inserts into $H$ with exactly the same coin flip probability. We now use this observation and $C'$.

We show that the memory used by $H$ when our algorithm is run on $C'$ with the all-knowing counters dominates (in expectation) the size of $H$ for the original data set in the streaming model, thus achieving the claimed bound in the current theorem statement.

Let PrHashMapInsert$(x,y)$ denote the probability of inserting the pair $x, y$ when we run the all-knowing version of the streaming algorithm on $C'$. Let PrStreamEmit$(x,y,a,b)$ denote the probability of emitting the pair $x, y$ in the streaming model with input $C$, after observing $x, y$ exactly $a, b$

times, respectively. With these definitions we have

$$\text{PrStreamEmit}(x,y,a,b) = \frac{p}{\varepsilon}\frac{1}{\sqrt{ab}}$$

$$\leq \frac{p}{\varepsilon}\frac{1}{2^{\lfloor \lg a \rfloor} 2^{\lfloor \lg b \rfloor}} \leq \frac{p}{\varepsilon}\frac{1}{\sqrt{\#(x_{\lfloor \lg a \rfloor})}\sqrt{\#(x_{\lfloor \lg b \rfloor})}}$$

$$= \text{PrHashMapInsert}(x_{\lfloor \lg a \rfloor + 1}, y_{\lfloor \lg b \rfloor + 1}).$$

The first inequality holds by properties of the floor function. The second inequality holds by definition of $C'$. The dictionary size for $C'$ is $\mathrm{O}(D \lg(N))$ where $D$ is the original dictionary size for $C$. Using the same analysis used in Theorem 2, the shuffle size for $C'$ is at most $O(DL\lg(N)\log(D)/\varepsilon)$ and therefore so is the size of $H$ for the all-knowing algorithm run on $C'$, and, by the analysis above, so is the hash map size for the original data set $C$. ■

## 7. Correctness and Shuffle Size Proofs for other Similarity Measures

We now give consideration to similarity scores other than cosine similarity, along proofs of their shuffle size.

### 7.1 Overlap Similarity

Overlap similarity follows the same pattern we used for cosine similarity; thus we only explain the parts that are different. The emit function changes to Algorithm 8.

**Algorithm 8** OverlapSampleEmit$(w_1, w_2)$ - $p/\varepsilon$ is the oversampling parameter

With probability

$$\frac{p}{\varepsilon}\frac{1}{\min(\#(w_1), \#(w_2))}$$

emit $((w_1, w_2) \to 1)$

The correctness proof is nearly identical to that of cosine similarity so we do not restate it. The shuffle size for OverlapSampleEmit is given by the following theorem.

**Theorem 8** *The expected shuffle size for OverlapSampleEmit is* $O(DL\log(D)/\varepsilon)$.

**Proof** The expected contribution from each pair of words will constitute the shuffle size:

$$\sum_{i=1}^{D}\sum_{j=i+1}^{D}\frac{p}{\varepsilon}\frac{\#(w_i,w_j)}{\min(\#(w_i),\#(w_j))}$$

$$\leq \frac{p}{\varepsilon}\sum_{i=1}^{D}\sum_{j=i+1}^{D}\#(w_i,w_j)\left(\frac{1}{\#(w_i)}+\frac{1}{\#(w_j)}\right)$$

$$\leq \frac{2p}{\varepsilon}\sum_{i=1}^{D}\frac{1}{\#(w_i)}\sum_{j=1}^{D}\#(w_i,w_j)$$

$$\leq \frac{2p}{\varepsilon}\sum_{i=1}^{D}\frac{1}{\#(w_i)}L\#(w_i) = \frac{2p}{\varepsilon}LD = O(DL\log(D)/\varepsilon)$$

The first inequality holds trivially. The last inequality holds because $w_i$ can co-occur with at most $\#(w_i)L$ other words. It is easy to see via Chernoff bounds that the above shuffle size is obtained with high probability. ■

### 7.2 Dice Similarity

Dice similarity follows the same pattern as we used for cosine similarity, thus we only explain the parts that are different. The emit function changes to Algorithm 9.

**Algorithm 9** DiceSampleEmit$(w_1, w_2)$ - $p/\varepsilon$ is the oversampling parameter

With probability

$$\frac{p}{\varepsilon}\frac{2}{\#(w_1)+\#(w_2)}$$

emit $((w_1, w_2) \to 1)$

The correctness proof is nearly identical to cosine similarity so we do not restate it. The shuffle size for DiceSampleEmit is given by the following theorem:

**Theorem 9** *The expected shuffle size for DiceSampleEmit is* $O(DL\log(D)/\varepsilon)$.

**Proof** The expected contribution from each pair of words will constitute the shuffle size:

$$2\sum_{i=1}^{D}\sum_{j=i+1}^{D}\frac{p}{\varepsilon}\frac{\#(w_i,w_j)}{\#(w_i)+\#(w_j)}$$

$$\leq \frac{2p}{\varepsilon}\sum_{i=1}^{D}\sum_{j=1}^{D}\frac{\#(w_i,w_j)}{\#(w_i)}$$

$$\leq \frac{2p}{\varepsilon}\sum_{i=1}^{D}\frac{1}{\#(w_i)}L\#(w_i) = \frac{2p}{\varepsilon}LD = O(DL\log(D)/\varepsilon)$$

The first inequality holds trivially. The last inequality holds because $w_i$ can co-occur with at most $\#(w_i)L$ other words. It is easy to see via Chernoff bounds that the above shuffle size is obtained with high probability. ■

## 8. Cross Product

Consider the case in which we have $N$ vectors (and *not* $D$) in $\mathbb{R}^N$ and wish to compute the cross product of all similarities between a subset $D$ of the $N$ vectors with the remaining $N-D$ vectors. This is different from computing all $\binom{D}{2}$ similarities, which has been the focus of the previous sections.

It is worth a brief mention that the same bounds and almost the same proof of Theorem 2 hold for this case, with one caveat, namely that the magnitudes of all $N$ points must be available to the mappers and reducers. We can mitigate this problem by annotating each dimension with the magnitudes of points beforehand in a separate MapReduce job. However, this MapReduce job will have a shuffle size dependent on $N$, so we do not consider it a part of the DISCO scheme.

## 9. Experiments

We use data from the social networking site Twitter, a very popular social network platform. Users interact with Twitter through a web interface, instant messaging clients, or sending mobile text messages. Public updates by users are viewable to the world, and a large majority of Twitter accounts are public. These public tweets provide a large real-time corpus of what is happening in the world.

We have two sets of experiments. The first is a large scale experiment that is in production, to compute similar users in the Twitter follow graph. The second is an experiment to find similar words. The point of the first experiment is show the scalability of our algorithms and the point of the second set of experiments is to show accuracy.

### 9.1 Similar Users

Consider the problem of finding all pairs of similarities between a subset of $D = 10^7$ twitter users. We would like to find all $\binom{D}{2}$ similarities between these users. The number of dimensions of each user is larger than $D$, and is denoted $N = 5 \times 10^8$. We used $p/\varepsilon = 100$. We have a classifier trained to pick the edges with most interaction and thus can limit the sparsity of the data set with $L = 1000$.

We compute these user similarities daily in a production environment using Hadoop and Pig implementations at Twitter (Olston et al., 2008), with the results supplementing Twitter.com. Using 100 Hadoop nodes we can compute the above similarities in around 2 hours. In this case, the naive scheme would have shuffle size $NL^2 = 5 \times 10^{14}$, which is infeasible but becomes possible with DISCO.

### 9.2 Similar Words

In the second experiment we used all tweets seen by Twitter.com in one day. The point in the experiment is to find all pairs of similar words, where words are similar if they appear in the same tweets. The number of dimensions $N = 198,134,530$ in our data is equal to the number of tweets and each tweet is a document with size at most 140 characters, providing a small upper bound for $L$. These documents are ideal for our framework, since our shuffle size upper bound depends on $L$, which in this case is very small. We used a dictionary of 1000 words advertisers on Twitter are currently targeting to show Promoted Trends, Trends, and Accounts. We also tried a uniformly random sampled dictionary without a qualitative change in results.

The reason we only used $D = 1000$ is because for the purpose of validating our work (i.e., reporting the small errors occurred by our algorithms), we have to compute the true cosine similarities, which means computing true co-occurrence for every pair, which is a challenging task computationally. This was a bottleneck only in *our experiments* for this paper, and does not affect users of our algorithms. We ran experiments with $D = 10^6$, but cannot report true error since finding the true cosine similarities is too computationally intensive. In this regime however, our theorems guarantee that the results are good with high probability.

### 9.3 Shuffle Size vs Accuracy

We have two parameters to tweak to tradeoff accuracy and shuffle size: $\varepsilon$ and $p$. However, since they only occur in our algorithms as the ratio $p/\varepsilon$, we simply use that as the tradeoff parameter. The reason we separated $p$ and $\varepsilon$ was for the theorems to go through nicely, but in reality we only see a single tweaking parameter.

We increase $p/\varepsilon$ exponentially on the x axis and record the ratio of DISCO shuffle size to the naive implementation. In all cases we can achieve a 90% reduction in shuffle size compared to the naive implementation without sacrificing much accuracy, as see in Figures 2, 4, and 6. The accuracy we report is with respect to true cosine, dice, and overlap similarity.

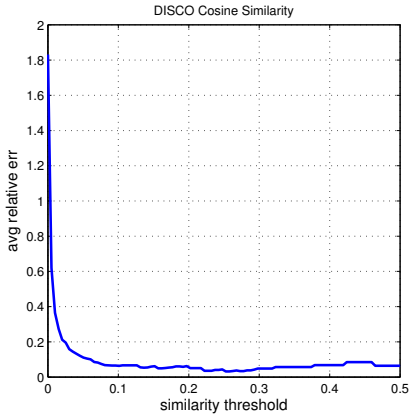



Figure 1: Average error for all pairs with similarity $\geq \varepsilon$. DISCO estimated Cosine error decreases for more similar pairs. Shuffle reduction from naive implementation: 99.39%.

### 9.4 Error vs Similarity Magnitude

All of our theorems report better accuracy for pairs that have higher similarity than otherwise. To see this empirically, we plot the average error of all pairs that have true similarity above $\varepsilon$. These can be seen in Figures 1, 3, 5, 7, and 8. Note that the reason for large portions of the error being constant in these plots is that there are very few pairs with very high similarities, and therefore the error remains constant while $\varepsilon$ is between the difference of two such very high similarity pairs. Further note that Figure 7 and 8 are so similar because our proposed DISCO MinHash very closely mimics the original MinHash. This is reflected in the strength of the bound in Theorem 5.

## 10. Conclusions and Future Directions

We presented the DISCO suite of algorithms to compute all pairwise similarities between very high dimensional sparse vectors. All of our results are provably independent of dimension, meaning that apart from the initial cost of trivially reading in the data, all subsequent operations are independent of the dimension and thus the dimension can be very large.

Although we use the MapReduce (Dean and Ghemawat, 2008) and Streaming computation models to discuss shuffle size and memory, the sampling strategy we use can be generalized to

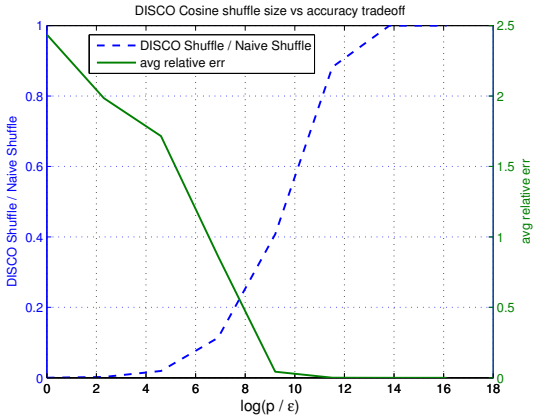



Figure 2: As $p/\varepsilon$ increases, shuffle size increases and error decreases. There is no thresholding for highly similar pairs here. Ground truth is computed with the naive algorithm. When the true similarity is zero, DISCO always also returns zero, so we always get those pairs right. It remains to estimate those pairs with similarity $> 0$, and that is the average relative error for those pairs that we report here.

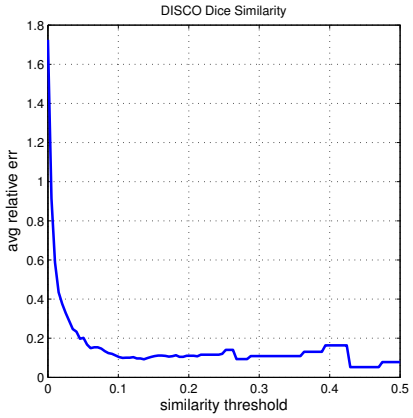



Figure 3: Average error for all pairs with similarity $\geq \varepsilon$. DISCO estimated Dice error decreases for more similar pairs. Shuffle reduction from naive implementation: 99.76%.

other frameworks. We anticipate the DISCO sampling strategy to be useful whenever one is computing a number between 0 and 1 by taking the ratio of an unknown number (the dot product in our case) to some known number (e.g., $\sqrt{\#(x)\#(y)}$ for cosine similarity) (Zadeh and Carlsson, 2013). This is a high-level description, and in order to make it more practical, we give five concrete examples, along with proofs and experiments.

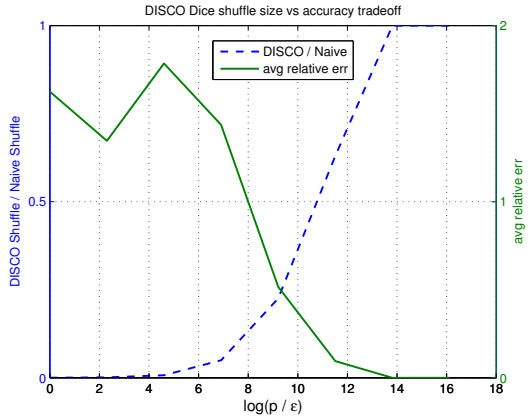



Figure 4: As $p/\varepsilon$ increases, shuffle size increases and error decreases. There is no thresholding for highly similar pairs here.

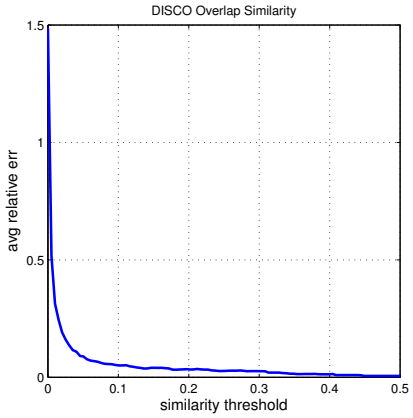



Figure 5: Average error for all pairs with similarity $\geq \varepsilon$. DISCO estimated Overlap error decreases for more similar pairs. Shuffle reduction from naive implementation: 97.86%.

## Acknowledgments

We thank the reviewers for their immensely helpful and thorough comments. We also thank the Twitter Personalization and Recommender Systems team for allowing use of production data. Supported in part by the DARPA xdata program, by grant FA9550- 12-1-0411 from the U.S. Air Force Office of Scientific Research (AFOSR) and the Defense Advanced Research Projects Agency (DARPA), and by NSF Award 0915040.

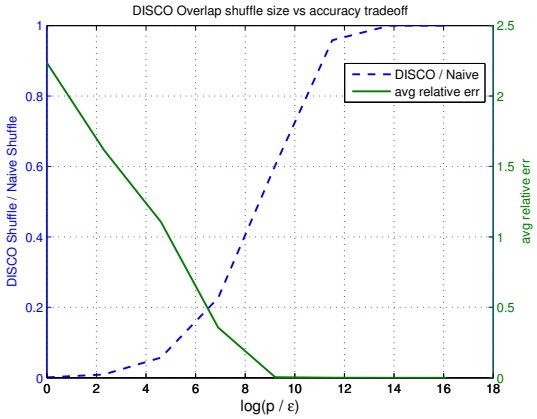



Figure 6: As $p/\varepsilon$ increases, shuffle size increases and error decreases. There is no thresholding for highly similar pairs here.

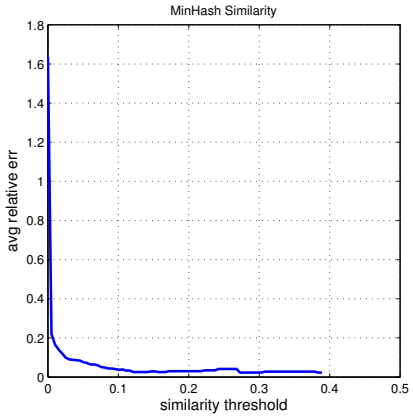



Figure 7: Average error for all pairs with similarity $\geq \varepsilon$. MinHash Jaccard similarity error decreases for more similar pairs. We are computing error with MinHash here, not ground truth.

## References

V. Abhishek and K. Hosanagar. Keyword generation for search engine advertising using semantic similarity between terms. In *EC 2007*, pages 89–94. ACM, 2007.

A. Arasu, V. Ganti, and R. Kaushik. Efficient exact set-similarity joins. In *VLDB 2006*, pages 918–929. VLDB Endowment, 2006.

R. Baraglia, G. De Francisci Morales, and C. Lucchese. Document similarity self-join with mapreduce. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 731–736. IEEE, 2010.

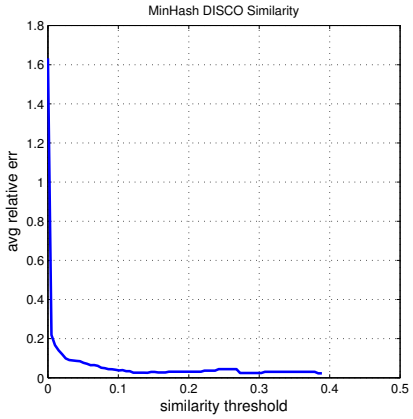



Figure 8: Average error for all pairs with similarity $\geq \varepsilon$. DISCO MinHash Jaccard similarity error decreases for more similar pairs. We are computing error with MinHash here, not ground truth.

D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *SIGKDD 2000*, pages 407–416. ACM, 2000.

D. Borthakur. The hadoop distributed file system: Architecture and design. *Hadoop Project Website*, 2007.

A.Z. Broder. On the resemblance and containment of documents. In *CCS 1997*, pages 21–29. IEEE, 1997.

A.Z. Broder, S.C. Glassman, M.S. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.

A. Campagna and R. Pagh. Finding associations and computing similarity via biased pair sampling. *Knowledge and Information Systems*, 31(3):505–526, 2012.

M.S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC 2002*, pages 380–388. ACM, 2002.

S. Chaudhuri, V. Ganti, and R. Kaushik. A primitive operator for similarity joins in data cleaning. In *ICDE 2006*, pages 5–5. IEEE, 2006.

S. Chien and N. Immorlica. Semantic similarity between search engine queries using temporal correlation. In *WWW 2005*, pages 2–11. ACM, 2005.

S.L. Chuang and L.F. Chien. Taxonomy generation for text segments: A practical web-based approach. *ACM TOIS*, 23(4):363–396, 2005.

J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008. ISSN 0001-0782.

T. Elsayed, J. Lin, and D.W. Oard. Pairwise document similarity in large collections with mapreduce. In *ACL 2008*, pages 265–268. Association for Computational Linguistics, 2008.

R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *ACM SIGMOD 2003*, pages 301–312. ACM, 2003.

A.F. Gates, O. Natkovich, S. Chopra, P. Kamath, S.M. Narayanamurthy, C. Olston, B. Reed, S. Srinivasan, and U. Srivastava. Building a high-level dataflow system on top of map-reduce: the pig experience. *Proceedings of the VLDB Endowment*, 2(2):1414–1425, 2009.

A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *VLDB 1999*, pages 518–529. Morgan Kaufmann Publishers Inc., 1999.

A Goel and K Munagala. Complexity measures for map-reduce, and comparison to parallel computing. *Manuscript* `http://www.stanford.edu/~ashishg/papers/mapreducecomplexity.pdf`, 2012.

P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. Wtf: The who to follow service at twitter. *The WWW 2013 Conference*, 2013.

P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC 1998*, pages 604–613. ACM, 1998.

J. Lin. Brute force and indexed approaches to pairwise document similarity comparisons with mapreduce. In *SIGIR 2009*, pages 155–162. ACM, 2009.

C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins. Pig latin: a not-so-foreign language for data processing. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1099–1110. ACM, 2008.

P. Pantel, E. Crestan, A. Borkovsky, A.M. Popescu, and V. Vyas. Web-scale distributional similarity and entity set expansion. In *EMNLP 2009*. ACL, 2009.

M. Regelson and D. Fain. Predicting click-through rate using keyword clusters. In *Proceedings of the Second Workshop on Sponsored Search Auctions*, volume 9623. Citeseer, 2006.

M. Sahami and T.D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *WWW 2006*, pages 377–386. ACM, 2006.

S. Sarawagi and A. Kirpal. Efficient set joins on similarity predicates. In *ACM SIGMOD 2004*, pages 743–754. ACM, 2004.

E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network. In *SIGKDD 2005*, pages 678–684. ACM, 2005.

S. Suri and S. Vassilvitskii. Counting triangles and the curse of the last reducer. In *Proceedings of the 20th International Conference on World Wide Web*, pages 607–614. ACM, 2011.

R.B. Zadeh and G. Carlsson. Dimension independent matrix square using mapreduce. *Symposium on Theory of Computing*, 2013.

Journal of Machine Learning Research 14 (2013) 1627-1663 Submitted 5/12; Revised 12/12; Published 6/13

# Dynamic Affine-Invariant Shape-Appearance Handshape Features and Classification in Sign Language Videos

**Anastasios Roussos** TROUSSOS@EECS.QMUL.AC.UK
*Queen Mary, University of London*
*School of Electronic Engineering and Computer Science*
*Mile End Road, London E1 4NS, UK*

**Stavros Theodorakis** STH@CS.NTUA.GR
**Vassilis Pitsikalis** VPITSIK@CS.NTUA.GR
**Petros Maragos** MARAGOS@CS.NTUA.GR
*National Technical University of Athens*
*School of Electrical and Computer Engineering*
*Zografou Campus, Athens 15773, Greece*

**Editor:** Isabelle Guyon and Vassilis Athitsos

## Abstract

We propose the novel approach of dynamic affine-invariant shape-appearance model (Aff-SAM) and employ it for handshape classification and sign recognition in sign language (SL) videos. Aff-SAM offers a compact and descriptive representation of hand configurations as well as regularized model-fitting, assisting hand tracking and extracting handshape features. We construct SA images representing the hand's shape and appearance *without* landmark points. We model the variation of the images by linear combinations of eigenimages followed by affine transformations, accounting for 3D hand pose changes and improving model's compactness. We also incorporate static and dynamic handshape priors, offering robustness in occlusions, which occur often in signing. The approach includes an *affine signer adaptation* component at the visual level, without requiring training from scratch a new singer-specific model. We rather employ a short development data set to adapt the models for a new signer. Experiments on the Boston-University-400 continuous SL corpus demonstrate improvements on handshape classification when compared to other feature extraction approaches. Supplementary evaluations of sign recognition experiments, are conducted on a multi-signer, 100-sign data set, from the Greek sign language lemmas corpus. These explore the fusion with movement cues as well as signer adaptation of Aff-SAM to multiple signers providing promising results.

**Keywords:** affine-invariant shape-appearance model, landmarks-free shape representation, static and dynamic priors, feature extraction, handshape classification

## 1. Introduction

Sign languages (SL), that is, languages that convey information via visual patterns, commonly serve as an alternative or complementary mode of human communication. The visual patterns of SL are formed mainly by handshapes and manual motion, as well as by non-manual patterns. The hand localization and tracking in a sign video as well as the derivation of features that reliably describe the configuration of the signer's hand are crucial for successful handshape classification. All the above are essential components for automatic sign language recognition systems or for gesture

©2013 Anastasios Roussos, Stavros Theodorakis, Vassilis Pitsikalis and Petros Maragos.

based human-computer interaction. Nevertheless, these tasks still pose several challenges, which are mainly due to the fast movement and the great variation of the hand's 3D shape and pose.

In this article, we propose a novel modeling of the shape and dynamics of the hands during signing that leads to efficient handshape features, employed to train statistical handshape models and finally for handshape classification and sign recognition. Based on 2D images acquired by a monocular camera, we employ a video processing approach that outputs reliable and accurate masks for the signer's hands and head. We construct *Shape-Appearance (SA) images* of the hand by combining 1) the hand's shape, as determined by its 2D hand mask, with 2) the hand's appearance, as determined by a normalized mapping of the colors inside the hand mask. The proposed modeling does not employ any landmark points and bypasses the point correspondence problem. In order to design a model of the variation of the SA images, which we call *Affine Shape-Appearance Model* (Aff-SAM), we modify the classic linear combination of eigenimages by incorporating *2D affine transformations*. These effectively account for various changes in the 3D hand pose and improve the model's compactness. After developing a procedure for the training of the Aff-SAM, we design a robust hand tracking system by adopting regularized model fitting that exploits prior information about the handshape and its dynamics. Furthermore, we propose to use as handshape features the Aff-SAM's eigenimage weights estimated by the fitting process.

The extracted features are fed into statistical classifiers based on Gaussian mixture models (GMM), via a supervised training scheme. The overall framework is evaluated and compared to other methods in extensive handshape classification experiments. The SL data are from the Boston University BU400 corpus (Neidle and Vogler, 2012). The experiments are based on manual annotation of handshapes that contain 3D pose parameters and the American Sign Language (ASL) handshape configuration. Next, we define classes that account for varying dependency of the handshapes w.r.t. the orientation parameters. The experimental evaluation addresses first, in a qualitative analysis the feature spaces via a cluster quality index. Second, we evaluate via supervised training a variety of classification tasks accounting for dependency w.r.t. orientation/pose parameters, with/without occlusions. In all cases we also provide comparisons with other baseline approaches or more competitive ones. The experiments demonstrate improved feature quality indices as well as classification accuracies when compared with other approaches. Improvements in classification accuracy for the non-occlusion cases are on average of 35% over baseline methods and 3% over more competitive ones. Improvements by taking into account the occlusion cases are on average of 9.7% over the more competitive methods.

In addition to the above, we explore the impact of Aff-SAM features in a sign recognition task based on statistical data-driven subunits and hidden Markov models. These experiments are applied on data from the Greek Sign Language (GSL) lemmas corpus (DictaSign, 2012), for two different signers, providing a test-bed for the fusion with movement-position cues, and as evaluation of the affine-adapted SA model to a new signer, for which there has been no Aff-SAM training. These experiments show that the proposed approach can be practically applied to multiple signers without requiring training from scratch for the Aff-SAM models.

## 2. Background and Related Work

The first step of a hand gesture analysis system is the localization of the hands. This is usually implemented using several types of visual features, as skin color, edge information, shape and motion. Color cues are applicable because of the characteristic colors of the human skin. Many methods,

including the one presented here, use skin color segmentation for hand detection (Argyros and Lourakis, 2004; Yang et al., 2002; Sherrah and Gong, 2000). Some degree of robustness to illumination changes can be achieved by selecting color spaces, as the *HSV*, *YCbCr* or the *CIE-Lab*, that separate the chromaticity from the luminance components (Terrillon et al., 2000; Kakumanu et al., 2007). In our approach, we adopt the *CIE-Lab* color space, due to its property of being perceptually uniform. Cui and Weng (2000) and Huang and Jeng (2001) employ motion cues assuming the hand is the only moving object on a stationary background, and that the signer is relatively still.

The next visual processing step is the hand tracking. This is usually based on blobs (Starner et al., 1998; Tanibata et al., 2002; Argyros and Lourakis, 2004), hand appearance (Huang and Jeng, 2001), or hand boundary (Chen et al., 2003; Cui and Weng, 2000). The frequent occlusions during signing make this problem quite challenging. In order to achieve robustness against occlusions and fast movements, Zieren et al. (2002), Sherrah and Gong (2000) and Buehler et al. (2009) apply probabilistic or heuristic reasoning for simultaneous assignment of labels to the possible hand/face regions. Our strategy for detecting and labeling the body-parts shares similarities with the above. Nevertheless, we have developed a more elaborate preprocessing of the skin mask, which is based on the mathematical morphology and helps us separate the masks of different body parts even in cases of overlaps.

Furthermore, a crucial issue to address in a SL recognition system is hand feature extraction, which is the focus of this paper. A commonly extracted positional feature is the 2D or 3D center-of-gravity of the hand blob (Starner et al., 1998; Bauer and Kraiss, 2001; Tanibata et al., 2002; Cui and Weng, 2000), as well as motion features (e.g., Yang et al., 2002; Chen et al., 2003). Several works use geometric measures related to the hand, such as shape moments (Hu, 1962; Starner et al., 1998) or sizes and distances between fingers, palm, and back of the hand (Bauer and Kraiss, 2001), though the latter employs color gloves. In other cases, the contour that surrounds the hand is used to extract translation, scale, and/or in-plane rotation invariant features, such as Fourier descriptors (Chen et al., 2003; Conseil et al., 2007).

Segmented hand images are usually normalized for size, in-plane orientation, and/or illumination and afterwards principal component analysis (PCA) is often applied for dimensionality reduction and descriptive representation of handshape (Sweeney and Downton, 1996; Birk et al., 1997; Cui and Weng, 2000; Wu and Huang, 2000; Deng and Tsui, 2002; Dreuw et al., 2008; Du and Piater, 2010). Our model uses a similar framework but differs from these methods mainly in the following aspects. First, we employ a more general class of transforms to align the hand images, namely affine transforms that extend both similarity transforms, used, for example, by Birk et al. (1997) and translation-scale transforms as in the works of Cui and Weng (2000), Wu and Huang (2000) and Du and Piater (2010). In this way, we can effectively approximate a wider range of changes in the 3D hand pose. Second, the estimation of the optimum transforms is done simultaneously with the estimation of the PCA weights, instead of using a pipeline to make these two sets of estimations. Finally, unlike all the above methods, we incorporate combined static and dynamic priors, which make these estimations robust and allow us to adapt an existing model on a new signer.

Closely related to PCA approaches, active shape and active appearance models (Cootes and Taylor, 2004; Matthews and Baker, 2004) are employed for handshape feature extraction and recognition (Ahmad et al., 1997; Huang and Jeng, 2001; Bowden and Sarhadi, 2002; Fillbrandt et al., 2003). Our proposed shape-appearance model follows the same paradigm with these methods but differs: the modeled images are Shape-Appearance images and the image warps are not controlled by the shape landmarks but more simply by the 6 parameters of the affine transformation. In this

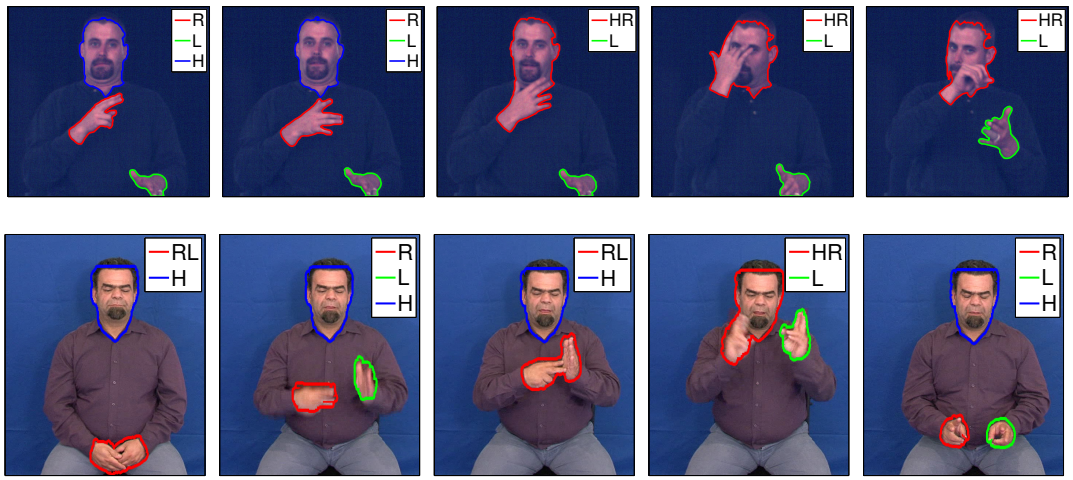



Figure 1: Output of the initial hands and head tracking in two videos of two different signers, from different databases. Example frames with extracted skin region masks and assigned body-part labels *H* (head), *L* (left hand), *R* (right hand).

way, it avoids shape representation through landmarks and the cumbersome manual annotation related to that.

Other more general purpose approaches have also been seen in the literature. A method earlier employed for action-type features is the histogram of oriented gradients (HOG): these descriptors are used for the handshapes of a signer (Buehler et al., 2009; Liwicki and Everingham, 2009; Ong et al., 2012). Farhadi et al. (2007) employ the scale invariant feature transform (SIFT) descriptors. Finally, Thangali et al. (2011) take advantage of linguistic constraints and exploit them via a Bayesian network to improve handshape recognition accuracy. Apart from the methods that process 2D hand images, there are methods built on a 3D hand model, in order to estimate the finger joint angles and the 3D hand pose (Athitsos and Sclaroff, 2002; Fillbrandt et al., 2003; Stenger et al., 2006; Ding and Martinez, 2009; Agris et al., 2008). These methods have the advantage that they can potentially achieve view-independent tracking and feature extraction; however, their model fitting process might be computationally slow.

Finally, regarding our related work, Roussos et al. (2010b) have included a short description of an initial tracking system similar to the one we adopt here. A preliminary version of the Aff-SAM method was presented by Roussos et al. (2010a). This is substantially extended here in many aspects, the main of which are the following: 1) We incorporate dynamic and static handshape priors offering robustness in cases of occlusions, 2) We develop an affine signer adaptation component, exploring the adaptation of Aff-SAM to multiple signers, 3) Extensive handshape classification experiments are presented, 4) Sign recognition experiments are conducted on a multi-signer database. In the sign recognition experiments of Section 8, we employ the handshape subunits construction presented by Roussos et al. (2010b). Finally, Theodorakis et al. (2012) and Theodorakis et al. (2011) present preliminary results on movement-handshape integration for continuous sign recognition.

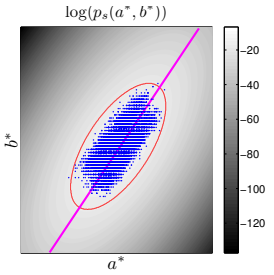



Figure 2: Skin color modeling. Training samples in the $a^*$-$b^*$ space and fitted pdf $p_s(a^*, b^*)$. The ellipse bounds the colors that are classified to skin, according to the thresholding of $p_s(a^*(x), b^*(x))$. The straight line corresponds to the first PCA eigendirection on the skin samples and determines the projection that defines the mapping $g(I)$ used in the Shape-Appearance images formation.

## 3. Visual Front-End Preprocessing

The initial step of the visual processing is not the main focus of our method, nevertheless we describe it for completeness and reproducibility. The output of this subsystem at every frame is a set of skin region masks together with one or multiple *labels* assigned to every region, Figure 1. These labels correspond to the *body-parts of interest* for sign language recognition: head ($H$), left hand ($L$) and right hand ($R$). The case that a mask has multiple labels reflects an *overlap* of the 2D regions of the corresponding body-parts, that is, there is an *occlusion* of some body-parts. Referring for example to the right hand, there are the following cases: 1) The system outputs a mask that contains the right hand only, therefore there is *no occlusion* related to that hand, and 2) The output mask includes the right hand as well as other body-part region(s), therefore there is an *occlusion*. As presented in Section 4, the framework of SA refines this tracking while extracting handshape features.

### 3.1 Probabilistic Skin Color Modeling

We are based on the color cue for body-parts detection. We consider a Gaussian model of the signer's skin color in the perceptually uniform color space *CIE-Lab*, after keeping the two chromaticity components $a^*$, $b^*$, to obtain robustness to illumination (Cai and Goshtasby, 1999). We assume that the $(a^*,b^*)$ values of skin pixels follow a bivariate Gaussian distribution $p_s(a^*, b^*)$, which is fitted using a training set of color samples (Figure 2). These samples are automatically extracted from pixels of the signer's face, detected using a face detector (Viola and Jones, 2003).

### 3.2 Morphological Processing of Skin Masks

In each frame, a first estimation of the skin mask $S_0$ is derived by thresholding at every pixel $x$ the value $p_s(a^*(x), b^*(x))$ of the learned skin color distribution, see Figures 2, 3(b). The corresponding threshold is determined so that a percentage of the training skin color samples are classified to skin. This percentage is set to 99% to cope with training samples outliers. The skin mask $S_0$ may contain spurious regions or holes inside the head area due to parts with different color, as for instance eyes, mouth. For this, we regularize $S_0$ with tools from mathematical morphology (Soille, 2004; Maragos, 2005): First, we use the concept of *holes* $\mathcal{H}(S)$ of a binary image $S$, that is, the set of background

(a) Input (b) $S_0$ (c) $S_2$ (d) $S_2 \ominus B_c$ (e) Segmented $S_2$

Figure 3: Results of skin mask extraction and morphological segmentation. (a) Input. (b) Initial skin mask estimation $S_0$. (c) Final skin mask $S_2$ (morphological refinement). (d) Erosion $S_2 \ominus B_c$ of $S_2$ and separation of overlapped regions. (e) Segmentation of $S_2$ based on competitive reconstruction opening.

components, not connected to the border of the image. In order to fill also some background regions that are not holes in the strict sense but are connected to the image border passing from a small "canal", we designed a filter that we call *generalized hole filling*. This filter yields a refined skin mask estimation $S_1 = S_0 \cup \mathcal{H}(S_0) \cup \{\mathcal{H}(S_0 \bullet B) \oplus B\}$ where $B$ is a structuring element with size $5 \times 5$ pixels, and $\oplus$ and $\bullet$ denotes Minkowski dilation, closing respectively. The connected components (CCs) of relevant skin regions can be at most three (corresponding to the head and the two hands) and cannot have an area smaller than a threshold $A_{min}$, which corresponds to the smallest possible area of a hand region for the current signer and video acquisition conditions. Therefore, we apply an *area opening* with a varying threshold value: we find all CCs of $S_1$, compute their areas and finally discard all the components whose area is not on the top 3 or is less than $A_{min}$. This yields the final skin mask $S_2$, see Figure 3(c).

### 3.3 Morphological Segmentation of the Skin Masks

In the frames where $S_2$ contains three CCs, these yield an adequate segmentation. On the contrary, when $S_2$ contains less than three CCs, the skin regions of interest occlude each other. In such cases though, the occlusions are not always essential: different skin regions in $S_2$ may be connected via a thin connection, Figure 3(c). Therefore we further segment the skin masks of some frames by separating occluded skin regions with thin connections: If $S_2$ contains $N_{cc} < 3$ connected components, we find the CCs of $S_2 \ominus B_c$, Figure 3(d), for a structuring element $B_c$ of small radius, for example, 3 pixels and discard those CCs whose area is smaller than $A_{min}$. A number of remaining CCs not greater than $N_{cc}$ implies the absence of any thin connection, thus does not provide any occlusion separation. Otherwise, we use each one of these CCs as the seed of a different segment and expand it to cover $S_2$. For this we propose a *competitive reconstruction opening*, see Figure 3(e), described by the following iterative algorithm: In every iteration 1) each evolving segment expands using its conditional dilation by the $3 \times 3$ cross, relative to $S_2$, 2) pixels belonging to more than one segment are excluded from all segments. This means that segments are expanded inside $S_2$ but their expansion stops wherever they meet other segments. The above two steps are repeated until all segments remain unchanged.

### 3.4 Body-part Label Assignment

This algorithm yields 1) an assignment of one or multiple body-part labels, *head, left* and *right hand*, to all the segments and 2) an estimation of ellipses at segments with multiple labels (occluded).

Note that these ellipses yield a rough estimate of the shapes of the occluded regions and contribute to the correct assignment of labels after each occlusion. A detailed presentation of this algorithm falls beyond the scope of this article. A brief description follows. *Non-occlusions*: For the hands' labels, given their values in the previous frames, we employ a prediction of the centroid position of each hand region taking into account three preceding frames and using a constant acceleration model. Then, we assign the labels based on minimum distances between the predicted positions and the segments' centroids. We also fit one ellipse on each segment since an ellipse can coarsely approximate the hand or head contour. *Occlusions*: Using the parameters of the body-part ellipses already computed from the three preceding frames, we employ similarly forward prediction for all ellipses parameters, assuming constant acceleration. We face non-disambiguated cases by obtaining an auxiliary centroid estimation of each body-part via template matching of the corresponding image region between consecutive frames. Then, we repeat the estimations backwards in time. Forward and backward predictions, are fused yielding a final estimation of the ellipses' parameters for the signer's head and hands. Figure 1 depicts the output of the initial tracking in sequences of frames with non-occlusion and occlusion cases. We observe that the system yields accurate skin extraction and labels assignment.

## 4. Affine Shape-Appearance Modeling

In this section, we describe the proposed framework of dynamic affine-invariant shape-appearance model which offers a descriptive representation of the hand configurations as well as a simultaneous hand tracking and feature extraction process.

### 4.1 Representation by Shape-Appearance images

We aim to model all possible configurations of the dominant hand during signing, using directly the 2D hand images. These images exhibit a high diversity due to the variations on the configuration and 3D hand pose. Further, the set of the visible points of the hand is significantly varying. Therefore, it is more effective to represent the 2D handshape without using any landmarks. We thus represent the handshape by implicitly using its binary mask $M$, while incorporating also the *appearance* of the hand, that is, the color values inside this mask. These values depend on the hand texture and shading, and offer crucial 3D information.

If $I(x)$ is a cropped part of the current color frame around the hand mask $M$, then the hand is represented by the following *Shape-Appearance (SA) image* (see Figure 4):

$$f(x) = \begin{cases} g(I(x)), & \text{if } x \in M \\ -c_b, & \text{otherwise} \end{cases},$$

where $g : \mathbb{R}^3 \to \mathbb{R}$ maps the color values of the skin pixels to a color parameter that is appropriate for the hand appearance representation. This mapping is more descriptive for hand representation than a common color-to-gray transform. In addition, $g$ is normalized so that the mapped values $g(I)$ of skin colors $I$ have zero mean and unit variance. $c_b > 1$ is a background constant that controls the balance between shape and appearance. As $c_b$ gets larger, the appearance variation gets relatively less weighted and more emphasis is given to the shape part. In the experiments, we have used $c_b = 3$ (that is three times the standard deviation of the foreground values $g(I)$).

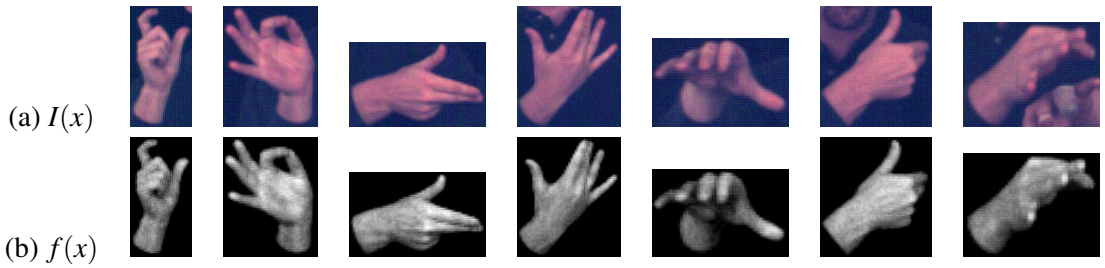



Figure 4: Construction of Shape-Appearance images. (a) Cropped hand images $I(x)$. (b) Corresponding Shape-Appearance images $f(x)$. For the foreground of $f(x)$ we use the most descriptive feature of the skin chromaticity. The background has been replaced by a constant value that is out of the range of the foreground values.

The mapping $g(I)$ is constructed as follows. First we transform each color value $I$ to the *CIE-Lab* color space, then keep only the chromaticity components $a^*$,$b^*$. Finally, we output the normalized weight of the first principal eigendirection of the PCA on the skin samples, that is the major axis of the Gaussian $p_s(a^*,b^*)$, see Section 3.1 and Figure 2(c). The output $g(I)$ is the most descriptive value for the skin pixels' chromaticity. Furthermore, if considered together with the training of $p_s(a^*,b^*)$, the mapping $g(I)$ is invariant to global similarity transforms of the values ($a^*$,$b^*$). Therefore, the SA images are invariant not only to changes of the luminance component $L$ but also to a wide set of global transforms of the chromaticity pair ($a^*$,$b^*$). As it will be described in Section 5, this facilitates the signer adaptation.

### 4.2 Modeling the Variation of Hand Shape-Appearance Images

Following Matthews and Baker (2004), the SA images of the hand, $f(x)$, are modeled by a linear combination of predefined variation images followed by an affine transformation:

$$f(W_p(x)) \approx A_0(x) + \sum_{i=1}^{N_c} \lambda_i A_i(x), x \in \Omega_M \ . \tag{1}$$

$A_0(x)$ is the mean image, $A_i(x)$ are $N_c$ eigenimages that model the linear variation. These images can be considered as affine-transformation-free images. In addition, $\lambda = (\lambda_1 \cdots \lambda_{N_c})$ are the weights of the linear combination and $W_p$ is an affine transformation with parameters $p = (p_1 \cdots p_6)$ that is defined as follows:

$$W_p(x,y) = \begin{pmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

The affine transformation models similarity transforms of the image as well as a significant range of changes in the 3D hand pose. It has a non-linear impact on the SA images and reduces the variation that is to be explained by the linear combination part, as compared to other appearance-based approaches that use linear models directly in the domain of the original images, (e.g., Cui and Weng, 2000). The linear combination of (1) models the changes in the configuration of the hand and the changes in the 3D orientation that cannot be modeled by the affine transform.

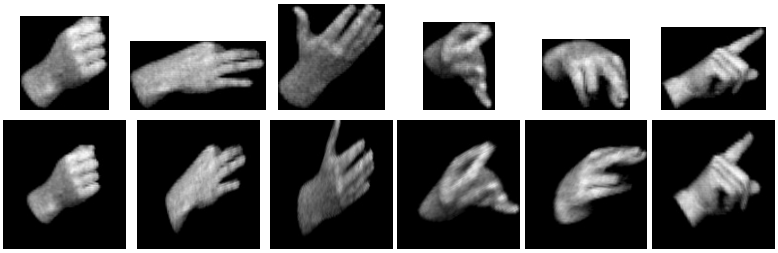


Figure 5: Semi-automatic affine alignment of a training set of Shape-Appearance images. (*Top row*) 6 out of 500 SA images of the training set. (*Bottom row*) Corresponding transformed images, after affine alignment of the training set. A video that demonstrates this affine alignment is available online (see text).

We will hereafter refer to the proposed model as *Shape-Appearance Model (SAM)*. A specific model of hand SA images is defined from the base image $A_0(x)$ and the eigenimages $A_i(x)$, which are statistically learned from training data. The vectors $p$ and $\lambda$ are the model parameters that fit the model to the hand SA image of every frame. These parameters are considered as features of hand pose and shape respectively.

### 4.3 Training of the SAM Linear Combination

In order to train the hand SA images model, we employ a representative set of handshape images from frames where the modeled hand is fully visible and non-occluded. Currently, this set is constructed by a random selection of approximately 500 such images. To exclude the variation that can be explained by the affine transformations of the model, we apply a semi-automatic affine alignment of the training SA images. For this, we use the framework of *procrustes analysis* (Cootes and Taylor, 2004; Dryden and Mardia, 1998), which is an iterative process that is repeatedly applying 1-1 alignments between pairs of training samples. In our case, the 1-1 alignments are affine alignments, implemented by applying the inverse-compositional (IC) algorithm (Gross et al., 2005) on pairs of SA images.

The IC algorithm result depends on the initialization of the affine warp, since the algorithm converges to a local optimum. Therefore, in each 1-1 alignment we test two different initializations: Using the binary masks $M$ of foreground pixels of the two SA images, these initializations correspond to the two similarity transforms that make the two masks have the same centroid, area and orientation.[1] Among the two alignment results, the plausible one is kept, according to manual feedback from a user.

It must be stressed that the manual annotation of plausible alignment results is needed only during the training of the SA model, not during the fitting of the model. Also, compared to methods that use landmarks to model the shape (e.g., Cootes and Taylor, 2004; Matthews and Baker, 2004; Ahmad et al., 1997; Bowden and Sarhadi, 2002), the amount of manual annotation during training is substantially decreased: The user here is not required to annotate points but just make a binary decision by choosing the plausible result of 1-1 alignments. Other related methods for aligning sets of images are described by Learned-Miller (2005) and Peng et al. (2010). However, the adopted Pro-

1. The existence of two such transforms is due to the modulo-$\pi$ ambiguity of the orientation.

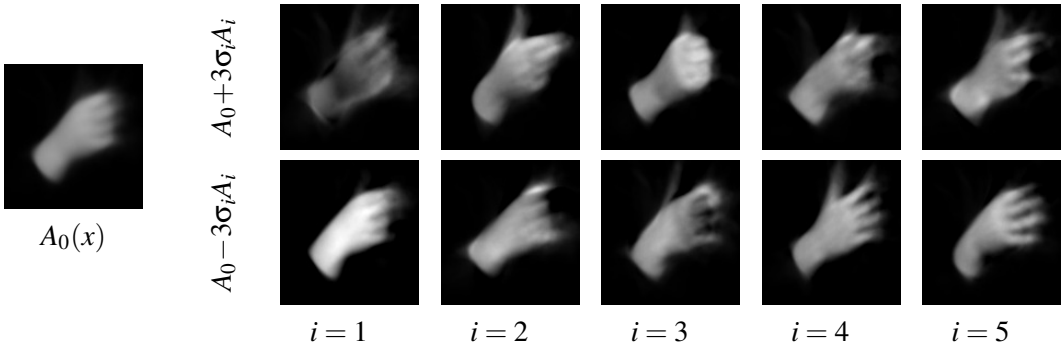



Figure 6: Result of the PCA-based learning of the linear variation images of Equation (1): Mean image $A_0(x)$ and principal modes of variation that demonstrate the first 5 eigenimages. The top (bottom) row corresponds to deviating from $A_0$ in the direction of the corresponding eigenimage, with a weight of $3\sigma_i$ (-$3\sigma_i$), where $\sigma_i$ is the standard deviation of the corresponding component.

crustes analysis framework facilitates the incorporation of the manual annotation in the alignment procedure. Figure 5 shows some results from the affine alignment of the training set. For more details, please refer to the following URL that contains a video demonstration of the training set alignment: `http://cvsp.cs.ntua.gr/research/sign/aff_SAM`. We observe that the alignment produces satisfactory results, despite the large variability of the images of the training set. Note that the resolution of the aligned images is $127\times133$ pixels.

Then, the images $A_i$ of the linear combination of the SA model are statistically learned using principal component analysis (PCA) on the aligned training SA images. The number $N_c$ of eigenimages kept is a basic parameter of the SA model. Using a larger $N_c$, the model can better discriminate different hand configurations. On the other hand, if $N_c$ gets too large, the model may not generalize well, in the sense that it will be consumed on explaining variation due to noise or indifferent information. In the setup of our experiments, we have practically concluded that the value $N_c = 35$ is quite effective. With this choice, the eigenimages kept explain 78% of the total variance of the aligned images.

Figure 6 demonstrates results of the application of PCA. Even though the modes of principal variation do not correspond to real handshapes, there is some intuition behind the influence of each eigenimage at the modeled hand SA image. For example, the first eigenimage $A_1$ has mainly to do with the foreground appearance: as its weight gets larger, the foreground intensities get darker and vice-versa. As another example, we see that by increasing the weight of the second eigenimage $A_2$, the thumb is extended. Note also that when we decrease the weight of $A_4$ all fingers extend and start detaching from each other.

### 4.4 Regularized SAM Fitting with Static and Dynamic Priors

After having built the shape-appearance model, we fit it in the frames of an input sign language video, in order to track the hand and extract handshape features. Precisely, we aim to find in every frame $n$ the parameters $\lambda = \lambda[n]$ and $p = p[n]$ that generate a model-based synthesized image that is sufficiently "close" to the current input SA image $f(x)$. In parallel, to achieve robustness against

occlusions, we exploit prior information about the handshape and its dynamics. Therefore, we minimize the following energy:

$$E(\lambda, p) = E_{rec}(\lambda, p) + w_S E_S(\lambda, p) + w_D E_D(\lambda, p) \,, \tag{2}$$

where $E_{rec}$ is a reconstruction error term. The terms $E_S(\lambda, p)$ and $E_D(\lambda, p)$ correspond to static and dynamic priors on the SAM parameters $\lambda$ and $p$. The values $w_S$,$w_D$ are positive weights that control the balance between the 3 terms.

*The reconstruction error term* $E_{rec}$ is a mean square difference defined by:

$$E_{rec}(\lambda, p) = \frac{1}{N_M} \sum_x \left\{ A_0(x) + \sum_{i=1}^{N_c} \lambda_i A_i(x) - f(W_p(x)) \right\}^2 ,$$

where the above summation is done over all the $N_M$ pixels $x$ of the domain of the images $A_i(x)$.

*The static priors term* $E_S(\lambda, p)$ ensures that the solution stays relatively close to the parameters mean values $\lambda_0$,$p_0$ :

$$E_S(\lambda, p) = \frac{1}{N_c} \|\lambda - \lambda_0\|^2_{\Sigma_\lambda} + \frac{1}{N_p} \|p - p_0\|^2_{\Sigma_p} \,,$$

where $N_c$ and $N_p$ are the dimensions of $\lambda$ and $p$ respectively (since we model affine transforms, $N_p$=6). These numbers act as normalization constants, since they correspond to the expected values of the quadratic terms that they divide. Also, $\Sigma_\lambda$ and $\Sigma_p$ are the covariance matrices of $\lambda$ and $p$ respectively,[2] which are estimated during the training of the priors (Section 4.4.2). We denote by $\|y\|_A$, with $A$ being a $N \times N$ symmetric positive-definite matrix and $y \in \mathbb{R}^N$, the following Mahalanobis distance from $y$ to 0:

$$\|y\|_A \triangleq \sqrt{y^T A^{-1} y} \,.$$

Using such a distance, the term $E_S(\lambda, p)$ penalizes the deviation from the mean values but in a weighted way, according to the appropriate covariance matrices.

*The dynamic priors term* $E_D(\lambda, p)$ makes the solution stay close to the parameters estimations $\lambda^e = \lambda^e[n]$, $p^e = p^e[n]$ based on already fitted values on adjacent frames (for how these estimations are derived, see Section 4.4.1):

$$E_D(\lambda, p) = \frac{1}{N_c} \|\lambda - \lambda^e\|^2_{\Sigma_{\varepsilon_\lambda}} + \frac{1}{N_p} \|p - p^e\|^2_{\Sigma_{\varepsilon_p}} \,, \tag{3}$$

where $\Sigma_{\varepsilon_\lambda}$ and $\Sigma_{\varepsilon_p}$ are the covariance matrices of the estimation errors of $\lambda$ and $p$ respectively, see Section 4.4.2 for the training of these quantities too. The numbers $N_c$ and $N_p$ act again as normalization constants. Similarly to $E_S(\lambda, p)$, the term $E_D(\lambda, p)$ penalizes the deviation from the predicted values in a weighted way, by taking into account the corresponding covariance matrices. Since the parameters $\lambda$ are the weights of the eigenimages $A_i(x)$ derived from PCA, we assume that their mean $\lambda_0 = 0$ and their covariance matrix $\Sigma_\lambda$ is diagonal, which means that each component of $\lambda$ is independent from all the rest.

It is worth mentioning that the energy-balancing weights $w_S$,$w_D$ are not constant through time, but depend on whether the modeled hand in the current frame is occluded or not (this information is provided by the initial tracking preprocessing step of Section 3). In the occlusion cases, we are

2. We have assumed that the parameters $\lambda$ and $p$ are statistically independent.

less confident than in the non-occlusion cases about the input SA image $f(x)$, which is involved in the term $E_{rec}(\lambda, p)$. Therefore, in these cases we obtain more robustness by increasing the weights $w_S$,$w_D$. In parallel, we decrease the relative weight of the dynamic priors term $\frac{w_D}{w_S+w_D}$, in order to prevent error accumulation that could be propagated in long occlusions via the predictions $\lambda^e$, $p^e$. After parameters tuning, we have concluded that the following choices are effective for the setting of our experiments: 1) $w_S$=0.07, $w_D$=0.07 for the non-occluded cases and 2) $w_S$=0.98, $w_D$=0.42 for the occluded cases.

An input video is split into much smaller temporal segments, so that the SAM fitting is *sequential* inside every segment as well *independent* from the fittings in all the rest segments: All the video segments of consecutive non-occluded and occluded frames are found and the middle frame of each segment is specified. For each non-occluded segment, we start from its middle frame and we get 1) a segment with forward direction by ending to the middle frame of the next occluded segment and 2) a segment with backward direction by ending after the middle frame of the previous occluded segment. With this splitting, we increase the confidence of the beginning of each sequential fitting, since in a non-occluded frame the fitting can be accurate even without dynamic priors. In the same time, we also get the most out of the dynamic priors, which are mainly useful in the occluded frames. Finally, this splitting strategy allows a high degree of parallelization.

#### 4.4.1 DYNAMICAL MODELS FOR PARAMETER PREDICTION

In order to extract the parameter estimations $\lambda^e$, $p^e$ that are used in the dynamic prior term $E_D$ (3), we use linear prediction models (Rabiner and Schafer, 2007). At each frame $n$, a varying number $K = K(n)$ of already fitted frames is used for the parameter prediction. If the frame is far enough from the beginning of the current sequential fitting, $K$ takes its maximum value, $K_{max}$. This maximum length of a prediction window is a parameter of our system (in our experiments, we used $K_{max} = 8$ frames). If on the other hand, the frame is close to the beginning of the corresponding segment, then $K$ varies from 0 to $K_{max}$, depending on the number of frames of the segment that have been already fitted.

If $K = 0$, we are at the starting frame of the sequential fitting, therefore no prediction from other available frames can be made. In this case, which is degenerate for the linear prediction, we consider that the estimations are derived from the prior means $\lambda^e = \lambda_0$, $p^e = p_0$ and also that $\Sigma_{\varepsilon_\lambda} = \Sigma_\varepsilon$, $\Sigma_{\varepsilon_p} = \Sigma_p$, which results to $E_D(\lambda, p) = E_S(\lambda, p)$. In all the rest cases, we apply the framework that is described next.

Given the prediction window value $K$, the parameters $\lambda$ are predicted using the following autoregressive model:

$$\lambda^e[n] = \sum_{\nu=1}^{K} A_\nu \lambda[n \mp \nu] \,,$$

where the $-$ sign ($+$ sign) corresponds to the case of forward (backward) prediction. Also, $A_\nu$ are $N_c \times N_c$ weight matrices that are learned during training (see Section 4.4.2). Note that for every prediction direction and for every $K$, we use a different set of weight matrices $A_\nu$ that is derived from a separate training. This is done to optimize the prediction accuracy for the specific case of every prediction window. Since the components of $\lambda$ are assumed independent to each other, it is reasonable to consider that all weight matrices $A_\nu$ are diagonal, which means that each component has an independent prediction model.

As far as the parameters $p$ are concerned, they do not have zero mean and we cannot consider them as independent since, in contrast to $\lambda$, they are not derived from a PCA. Therefore, in order to apply the same framework as above, we consider the following re-parametrization:

$$\widetilde{p} = U_p^T (p - p_0) \Leftrightarrow p = p_0 + U_p \widetilde{p} \,,$$

where the matrix $U_p$ contains column-wise the eigenvectors of $\Sigma_p$. The new parameters $\widetilde{p}$ have zero mean and diagonal covariance matrix. Similarly to $\lambda$, the normalized parameters $\widetilde{p}$ are predicted using the following model:

$$\widetilde{p}^e[n] = \sum_{\nu=1}^{K} \mathrm{B}_\nu \widetilde{p}[n \mp \nu] \,,$$

where $\mathrm{B}_\nu$ are the corresponding weight matrices which again are all considered diagonal.

#### 4.4.2 AUTOMATIC TRAINING OF THE STATIC AND DYNAMIC PRIORS

In order to apply the regularized SAM fitting, we first learn the priors on the parameters $\lambda$ and $p$ and their dynamics. This is done by training subsequences of frames where the modeled hand is not occluded. This training does not require any manual annotation. We first apply a random selection of such subsequences from videos of the same signer. Currently, the randomly selected subsequences used in the experiments are 120 containing totally 2882 non-occluded frames and coming from 3 videos. In all the training subsequences, we fit the SAM in each frame independently by minimizing the energy in Equation (2) with $w_S$=$w_D$=0 (that is without prior terms). In this way, we extract fitted parameters $\lambda$, $p$ for all the training frames. These are used to train the static and dynamic priors.

#### 4.4.3 STATIC PRIORS

In this case, for both cases of $\lambda$ and $p$, the extracted parameters from all the frames are used as samples of the same multivariate distribution, without any consideration of their successiveness in the training subsequences. In this way, we form the training sets $T_\lambda$ and $T_p$ that correspond to $\lambda$ and $p$ respectively. Concerning the parameter vector $\lambda$, we have assumed that its mean $\lambda_0 = 0$ and its covariance matrix $\Sigma_\lambda$ is diagonal. Therefore, only the diagonal elements of $\Sigma_\lambda$, that is the variances $\sigma^2_{\lambda_i}$ of the components of $\lambda$, are to be specified. This could be done using the result of the PCA (Section 4.2), but we employ the training parameters of $T_\lambda$ that come from the direct SAM fitting, since they are derived from a process that is closer to the regularized SAM fitting. Therefore, we estimate each $\sigma^2_{\lambda_i}$ from the empirical variance of the corresponding component $\lambda_i$ in the training set $T_\lambda$. Concerning the parameters $p$, we estimate $p_0$ and $\Sigma_p$ from the empirical mean and covariance matrix of the training set $T_p$.

#### 4.4.4 DYNAMIC PRIORS

As already mentioned, for each prediction direction (forward, backward) and for each length $K$ of the prediction window, we consider a different prediction model. The $(K+1)$-plets[3] of samples for each one of these models are derived by sliding the appropriate window in the training sequences. In order to have as good accuracy as possible, we do not make any zero (or other) padding in unknown parameter values. Therefore, the samples are picked only when the window fits entirely inside the

3. The $(K+1)$-plets follow from the fact that we need $K$ neighbouring samples + the current sample.

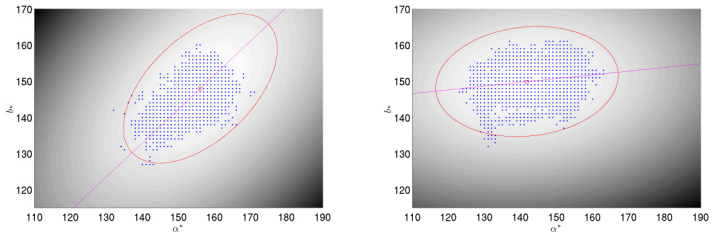
Figure 7: Regularized Shape-Appearance Model fitting in a sign language video. In every input frame, we superimpose the model-based reconstruction of the hand in the frame domain, $A_0(W_p^{-1}(x)) + \sum \lambda_i A_i(W_p^{-1}(x))$. In the upper-right corner, we display the reconstruction in the model domain, $A_0(x) + \sum \lambda_i A_i(x)$, which determines the optimum weights $\lambda$. A demo video is available online (see text).

training sequence. Similarly to linear predictive analysis (Rabiner and Schafer, 2007) and other tracking methods that use dynamics (e.g., Blake and Isard, 1998) we learn the weight matrices $A_\nu$, $B_\nu$ by minimizing the mean square estimation error over all the prediction-testing frames. Since we have assumed that $A_\nu$ and $B_\nu$ are diagonal, this optimization is done independently for each component of $\lambda$ and $\widetilde{p}$, which is treated as 1D signal. The predictive weights for each component are thus derived from the solution of an ordinary least squares problem. The optimum values of the mean squared errors yield the diagonal elements of the prediction errors' covariance matrices $\Sigma_{\varepsilon_\lambda}$ and $\Sigma_{\varepsilon_{\widetilde{p}}}$, which are diagonal.

#### 4.4.5 IMPLEMENTATION AND RESULTS OF SAM FITTING

The energy $E(\lambda, p)$ (2) of the proposed regularized SAM fitting is a special case of the general objective function that is minimized by the *simultaneous inverse compositional with a prior* (SICP) algorithm of Baker et al. (2004). Therefore, in order to minimize $E(\lambda, p)$, we specialize this algorithm for the specific types of our prior terms. Details are given in the Appendix A. At each frame $n$ of a video segment, the fitting algorithm is initialized as follows. If the current frame is not the starting frame of the sequential fitting (that is $K(n) \neq 0$), then the parameters $\lambda$, $p$ are initialized from the predictions $\lambda^e$, $p^e$. Otherwise, if $K(n) = 0$, we test as initializations the two similarity transforms that, when applied to the SAM mean image $A_0$, make its mask have the same centroid, area and orientation as the mask of the current frame's SA image. We twice apply the SICP algorithm using these two initializations, and finally choose the initialization that yields the smallest regularized energy $E(\lambda, p)$.

Figure 7 demonstrates indicative results of the regularized fitting of the dominant hand's SAM in a sign language video. For more details, please refer to the following URL that contains a video of these results: `http://cvsp.cs.ntua.gr/research/sign/aff_SAM`. We observe that in non-occlusion cases, this regularized method is effective and accurately tracks the handshape. Further, in occlusion cases, even after a lot of occluded frames, the result is especially robust. Nevertheless, the accuracy of the extracted handshape is smaller in cases of occlusions, compared to the non-




Signer A Signer B

Figure 8: Skin color modeling for the two signers of the GSL lemmas corpus, where we test the signer adaptation. Training samples in the $a^*$-$b^*$ chromaticity space and fitted pdf's $p_s(a^*, b^*)$. In each case, the straight line defines the normalized mapping $g(I)$ used in the Shape-Appearance images formation.

occlusion cases, since the prior terms keep the result closer to the SAM mean image $A_0$. In addition, extensive handshape classification experiments were performed in order to evaluate the extracted handshape features employing the proposed Aff-SAM method (see Section 7).

## 5. Signer Adaptation

We develop a method for adapting a trained Aff-SAM model to a new signer. This adaptation is facilitated by the characteristics of the Aff-SAM framework. Let us consider an Aff-SAM model trained to a signer, using the procedure described in Section 4.3. We aim to reliably adapt and fit the existing Aff-SAM model on videos from a *new signer*.

### 5.1 Skin Color and Normalization

The employed skin color modeling adapts on the characteristics of the skin color of a new signer. Figure 8 illustrates the skin color modeling for the two signers of the GSL lemmas corpus, where we test the adaptation. For each new signer, the color model is built from skin samples of a face tracker (Section 3.1, Section 4.1). Even though there is an intersection, the domain of colors classified as skin is different between the two. In addition, the mapping $g(I)$ of skin color values, used to create the SA images, is normalized according to the skin color distribution of each signer. The differences in the lines of projection reveal that the normalized mapping $g(I)$ is different in these two cases. This skin color adaptation makes the body-parts label extraction of the visual front-end preprocessing to behave robustly over different signers. In addition, the extracted SA images have the same range of values and are directly comparable across signers.

### 5.2 Hand Shape and Affine Transforms

Affine transforms can reliably compensate for the anatomical differences of the hands of different signers. Figure 9 demonstrates some examples. In each case, the right hands of the signers are in a similar configuration and viewpoint. We observe that there exist pairs of affine transformations that successfully align the handshapes of both signers to the common model domain. For instance,

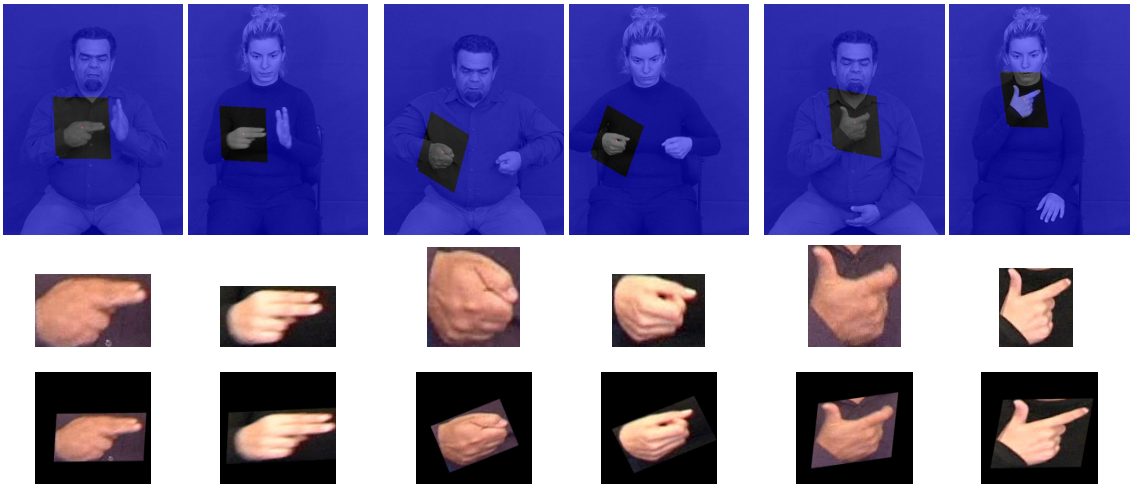


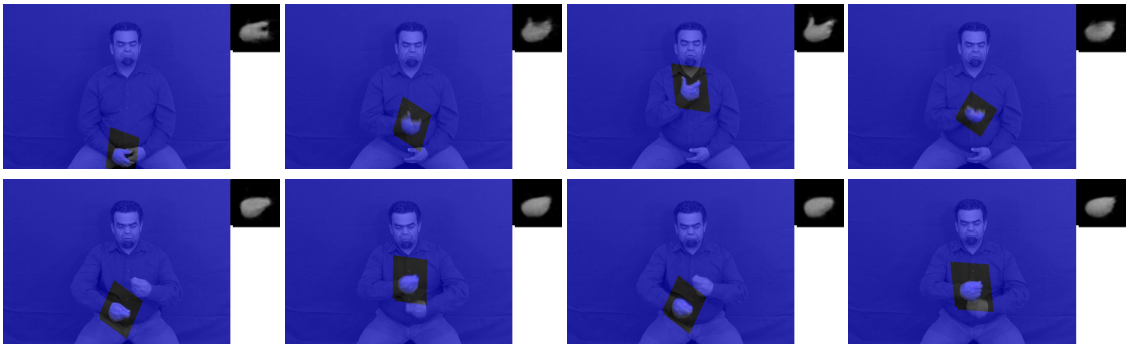
Figure 9: Alignment of the hands of two different signers, using affine transformations. *First row*: Input frames with superimposed rectangles that visualize the affine transformations. *Second row*: Cropped images around the hand. *Third row*: Alignment of the cropped images in a common model domain, using the affine transformations.

the affine transforms have the ability to stretch or shrink the hand images over the major hand axis. They thus automatically compensate for the fact that the second signer has thinner hands and longer fingers. In general, the class of affine transforms can effectively approximate the transformation needed to align the 2D hand shapes of different signers.

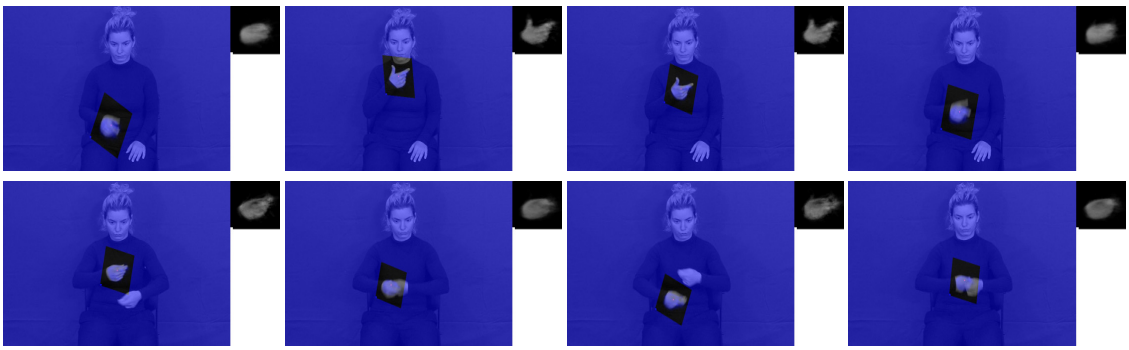
### 5.3 New Signer Fitting

To process a new signer the visual front-end is applied as in Section 3. Then, we only need to retrain the static and dynamic priors on the new signer. For this, we randomly select frames where the hand is not occluded. Then, for the purposes of this training, the existing SAM is fitted on them by minimizing the energy in Equation (2) with $w_S$=$w_D$=0, namely the reconstruction error term without prior terms. Since the SAM is trained on another signer, this fitting is not always successful, at this step. At that point, the user annotates the frames where this fitting has succeeded. This feedback is binary and is only needed during training and for a relatively small number of frames. For example, in the case of the GSL lemmas corpus, we sampled frames from approximately 1.2% of all corpus videos of this signer. In 15% of the sampled frames, this fitting with no priors was annotated as successful. Using the samples from these frames, we learn the static and dynamic priors of $\lambda$ and $p$, as described in Section 4.4.2 for the new signer. The regularized SAM fitting is implemented as in Section 4.4.5.

Figure 10 demonstrates results of the SAM fitting, in the case of signer adaptation. The SAM eigenimages are learned using solely Signer A. The SAM is then fitted on the signer B, as above. For comparison, we also visualize the result of the SAM fitting to the signer A, for the same sign. Demo videos for these fittings also are included in the following URL: `http://cvsp.cs.ntua.gr/research/sign/aff_SAM`. We observe that, despite the anatomical differences of the two signers,



Source signer (A)



New signer (B)

Figure 10: Regularized Shape-Appearance Model fitting on 2 signers. The SA model was trained on Signer A and adapted for Signer B. Demo videos are available online (see text).

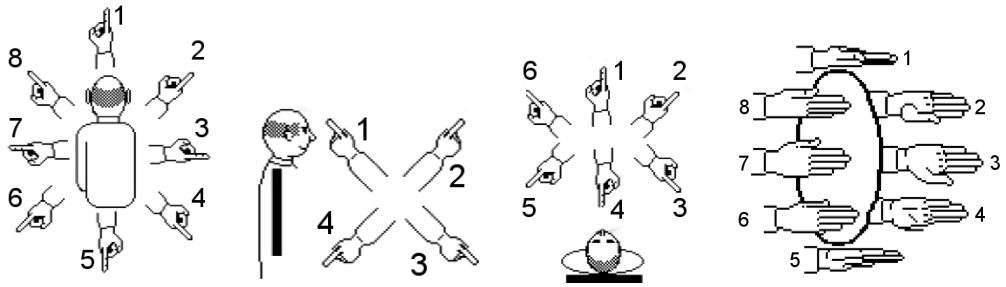
the performance of the SAM fitting is satisfactory after the adaptation. In both signers, the fitting yields accurate shape estimation in non-occlusion cases.

## 6. Data Set and Handshape Annotation for Handshape Classification

The *SL Corpus BU400* (Neidle and Vogler, 2012) is a continuous American sign language database. The background is uniform and the images have a resolution of 648x484 pixels, recorded at 60 frames per second. In the classification experiments we employ the front camera video, data from a single signer, and the story 'Accident'. We next describe the annotation parameters required to produce the ground-truth labels. These concern the pose and handshape configurations and are essential for the supervised classification experiments.

### 6.1 Handshape Parameters and Annotation

The parameters that need to be specified for the annotation of the data are the (pose-independent) handshape configuration and the 3D hand pose, that is the orientation of the hand in the 3D space. For the annotation of the handshape configurations we followed the *SignStream annotation conventions* (Neidle, 2007). For the 3D hand pose we parametrized the 3D hand orientations inspired




(*a*) Front (F) (*b*) Side (S) (*c*) Bird's (B) (*d*) Palm (P)

Figure 11: 3D Hand Orientation parameters: (*a-c*) Extended Finger Direction Parameters: (*a*) Signer's front view (F), (*b*) Side view (S), (*c*) Birds' view (B); (*d*) Palm orientation (P). Note that we have modified the corresponding figures of Hanke (2004) with numerical parameters.

by the HamNoSys description (Hanke, 2004). The adopted annotation parameters are as follows: 1) *Handshape identity (HSId)* which defines the handshape configuration, that is, ('*A*', '*B*', '*1*', '*C*' etc.), see Table 1 for examples. 2) *3D Hand Orientation* (hand pose) consisting of the following parameters (see Figure 11): i) *Extended Finger Direction* parameters that define the orientation of the hand axis. These correspond to the hand orientation relatively to the three planes that are defined relatively to: the Signer's Front view (referred to as F), the Bird's view (B) and the Side view (S). ii) *Palm Orientation* parameter (referred to as P) for a given extended finger direction. This parameter is defined w.r.t. the bird's view, as shown in Figure 11(d).

### 6.2 Data Selection and Classes

We select and annotate a set of occluded and non-occluded handshapes so that 1) they cover substantial handshape and pose variation as they are observed in the data and 2) they are quite frequent. More specifically we have employed three different data sets (DS): 1) *DS-1*: 1430 non-occluded handshape instances with 18 different HSIds. 2) *DS-1-extend*: 3000 non-occluded handshape instances with 24 different HSIds. 3) *DS-2*: 4962 occluded and non-occluded handshape instances with 42 different HSIds. Table 1 presents an indicative list of annotated handshape configurations and 3D hand orientation parameters.

## 7. Handshape Classification Experiments

In this section we present the experimental framework consisting of the statistical system for handshape classification. This is based 1) on the handshape features extracted as described in Section 4; 2) on the annotations as described in Section 6.1 as well as 3) on the data selection and classes (Section 6.2). Next, we describe the experimental protocol containing the main experimental variations of the data sets, of the class dependency, and of the feature extraction method.

| HSId | *1* | *1* | *4* | *4* | *5C* | *5* | *5* | *5* | *A* | *A* | *BL* | *BL* | *BL* | *BL* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3D hand pose F | 8 | 1 | 7 | 6 | 1 | 7 | 8 | 1 | 8 | 8 | 8 | 7 | 8 | 8 |
| 3D hand pose S | 0 | 0 | 0 | 3 | 1 | 0 | 2 | 2 | 0 | 2 | 0 | 0 | 0 | 0 |
| 3D hand pose B | 0 | 0 | 0 | 6 | 4 | 0 | 1 | 1 | 0 | 6 | 0 | 0 | 0 | 0 |
| 3D hand pose P | 1 | 8 | 3 | 1 | 3 | 3 | 1 | 5 | 3 | 2 | 2 | 3 | 3 | 4 |
| # insts. | 14 | 24 | 10 | 12 | 27 | 38 | 14 | 19 | 14 | 31 | 10 | 15 | 23 | 30 |
| exmpls. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

| HSId | *BL* | *CUL* | *F* | *F* | *U* | *UL* | *V* | *Y* | *b1* | *c5* | *c5* | *cS* | *cS* | *fO2* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3D hand pose F | 8 | 7 | 7 | 1 | 7 | 7 | 8 | 8 | 7 | 8 | 8 | 7 | 8 | 8 |
| 3D hand pose S | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 |
| 3D hand pose B | 6 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 6 | 6 | 0 |
| 3D hand pose P | 4 | 3 | 3 | 3 | 2 | 3 | 2 | 2 | 3 | 3 | 1 | 3 | 3 | 1 |
| # insts. | 20 | 13 | 23 | 13 | 10 | 60 | 16 | 16 | 10 | 17 | 18 | 10 | 34 | 12 |
| exmpls. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Table 1: Samples of annotated handshape identities (HSId) and corresponding 3D hand orientation (pose) parameters for the D-HFSBP class dependency and the corresponding experiment; in this case each model is fully dependent on all of the orientation parameters. '# insts.' corresponds to the number of instances in the dataset. In each case, we show an example handshape image that is randomly selected among the corresponding handshape instances of the same class.

### 7.1 Experimental Protocol and Other Approaches

The experiments are conducted by employing cross-validation by selecting five different random partitions of the dataset into train-test sets. We employ 60% of the data for training and 40% for testing. This partitioning samples data, among all realizations per handshape class in order to equalize class occurrence. The number of realizations per handshape class are on average 50, with a minimum and maximum number of realizations in the range of 10 to 300 depending on the experiment and the handshape class definition. We assign to each experiment's training set one GMM per handshape class; each has one mixture and diagonal covariance matrix. The GMMs are uniformly initialized and are afterwards trained employing Baum-Welch re-estimation (Young et al., 1999). Note that we are not employing other classifiers since we are interested in the evaluation of the handshape features and not the classifier. Moreover this framework fits with common hidden Markov model (HMM)-based SL recognition frameworks (Vogler and Metaxas, 1999), as in Section 8.

#### 7.1.1 Experimental Parameters

The experiments are characterized by the dataset employed, the class dependency and the feature extraction method as follows:

| Class Dependency label | Annotation Parameters HSId(H) | Front(F) | Side(S) | Bird's(B) | Palm(P) |
|---|---|---|---|---|---|
| D-HFSBP | D | D | D | D | D |
| D-HSBP | D | * | D | D | D |
| D-HBP | D | * | * | D | D |
| D-HP | D | * | * | * | D |
| D-H | D | * | * | * | * |

Table 2: Class dependency on orientation parameters. One row for each model dependency w.r.t. the annotation parameters. The dependency or non-dependency state to a particular parameter for the handshape trained models is noted as 'D' or '*' respectively. For instance the D-HBP model is dependent on the HSId and Bird's view and Palm orientation parameters.

*Data Set (DS):* We have experimented employing three different data sets DS-1, DS-1-extend and DS-2 (Section 6.2 for details).

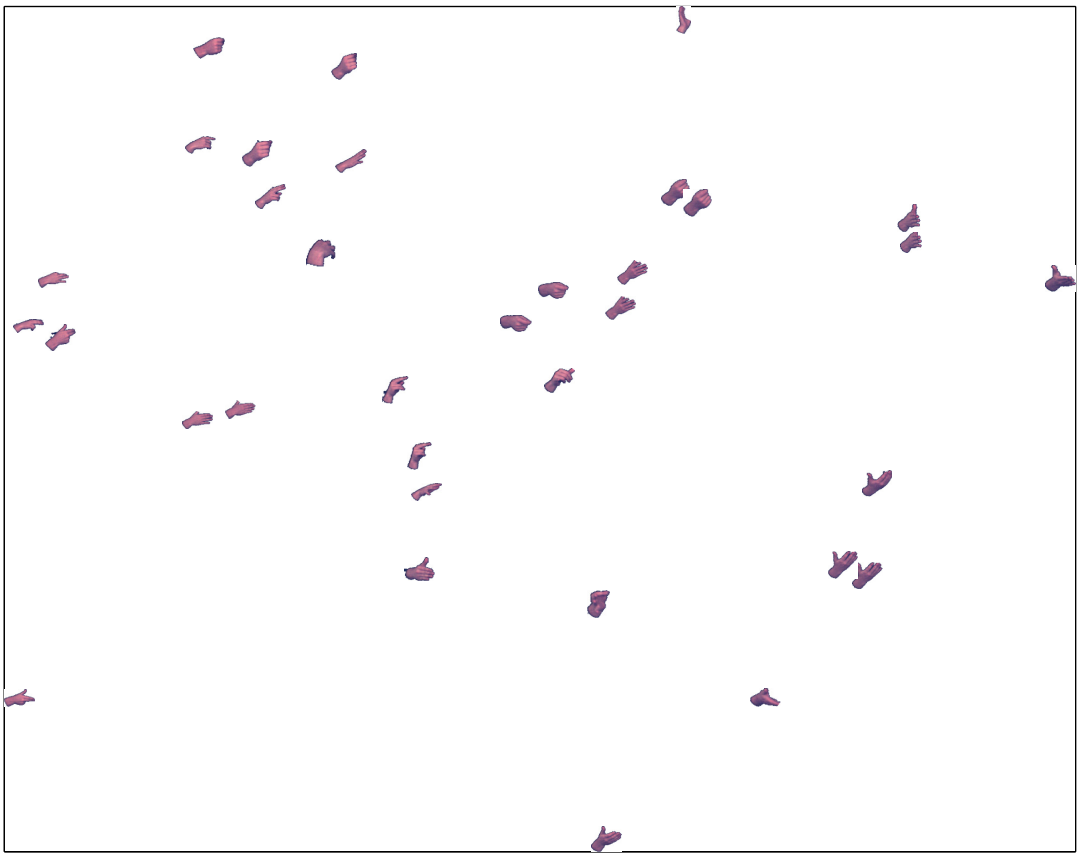
*Class dependency (CD):* The class dependency defines the orientation parameters in which our trained models are dependent to (Table 2). Take for instance the orientation parameter 'Front' (F). There are two choices, either 1) construct handshape models independent to this parameter or 2) construct different handshape models for each value of the parameter. In other words, at one extent CD restricts the models generalization by making each handshape model specific to the annotation parameters, thus highly discriminable, see for instance in Table 2 the experiment corresponding to D-HFSBP. At the other extent CD extends the handshape models generalization w.r.t. to the annotation parameters, by letting the handshape models account for pose variability (that is depend only on the HSId; same HSId's with different pose parameters are tied), see for instance experiment corresponding to the case D-H (Table 2). The CD field takes the values shown in Table 2.

#### 7.1.2 FEATURE EXTRACTION METHOD

Apart from the proposed Aff-SAM method, the methods employed for handshape feature extraction are the following:

*Direct Similarity Shape-Appearance Modeling* (DS-SAM): Main differences of this method with Aff-SAM are as follows: *1)* we replace the affine transformations that are incorporated in the SA model (1) by simpler *similarity* transforms and *2)* we replace the regularized model fitting by direct estimation (*without* optimization) of the similarity transform parameters using the centroid, area and major axis orientation of the hand region followed by projection into the PCA subspace to find the eigenimage weights. Note that in the occlusion cases, this simplified fitting is done directly on the SA image of the region that contains the modeled hand as well as the other occluded body-part(s) (that is the other hand and/or the head), without using any static or dynamic priors as those of Section 4.4. This approach is similar to Birk et al. (1997) and is adapted to fit our framework.

*Direct Translation Scale Shape-Appearance Modeling* (DTS-SAM): The main differences of this method with Aff-SAM are the following: *1)* we replace the affine transformations that are incorporated in the Shape-Appearance model (1) by simpler *translation-scale* transforms and *2)* we replace the regularized model fitting by direct estimation of the translation and scale parameters



(a)

Figure 12: Feature space for the Aff-SAM features and the D-HFSBP experiment case (see text). The trained models are visualized via projections on the $\lambda_1 - \lambda_2$ plane that is formed from the weights of the two principal Aff-SAM eigenimages. Cropped handshape images are placed at the models' centroids.

using the square that tightly surrounds the hand mask, followed again by projection into the PCA subspace to find the eigenimage weights. In this simplified version too, the hand occlusion cases are treated by simply fitting the model to the Shape-Appearance image that contains the occlusion, without static or dynamic priors. This approach is similar to Cui and Weng (2000), Wu and Huang (2000) and Du and Piater (2010) and is adapted so as to fit our proposed framework.

Other tested methods from the literature contain the *Fourier Descriptors* (FD): These are derived from the Fourier coefficients of the contour that surrounds the hand, after appropriate normalizations for scale and rotation invariance (Chen et al., 2003; Conseil et al., 2007). For dimensionality reduction, we keep the descriptors that correspond to the first $N_{FD}$ frequencies. We tested different values for the parameter $N_{FD}$ and finally kept $N_{FD} = 50$ that yield the best performance. *Moments* (M): These consist of the seven Hu moment invariants of the hand region (Hu, 1962). These depend only on the central moment of the binary shape of the hand region and are invariant to similarity transforms of the hand region. *Region Based* (RB): These consist of the area, eccentricity, com-

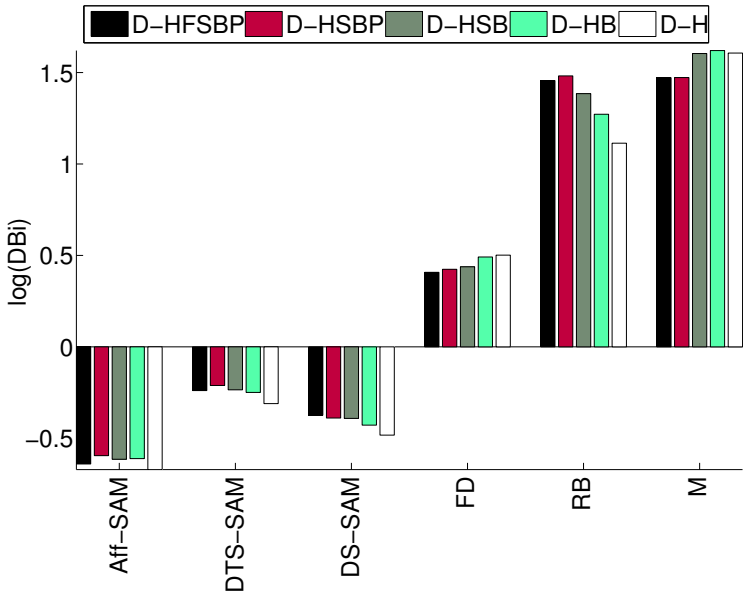



Figure 13: Davies-Bouldin index (DBi) in logarithmic scale (y-axis) for multiple feature spaces and varying models class dependency to the orientation parameters. Lower values of DBi indicate better compactness and separability of classes.

pactness and minor and major axis lengths of the hand region (Agris et al., 2008). Compared to the proposed Aff-SAM features we consider the rest five sets of features belonging to either *baseline features* or *more advanced features*. First, the baseline features contain the FD, M and RB approaches. Second, the more advanced features contain the DS-SAM and DTS-SAM methods which we have implemented as simplified versions of the proposed Aff-SAM. As it will be revealed by the evaluations, the more advanced features are more competitive than the baseline features and the comparisons with them are more challenging.

### 7.2 Feature Space Evaluation Results

Herein we evaluate the feature space of the Aff-SAM method. In order to approximately visualize it, we employ the weights $\lambda_1, \lambda_2$ of the two principal eigenimages of Aff-SAM. Figure 12(a) provides a visualization of the trained models per class, for the experiment corresponding to D-HFSBP class dependency (that is each class is fully dependent on orientation parameters). It presents a single indicative cropped handshape image per class to add intuition on the presentation: these images correspond to the points in the feature space that are closest to the specific classes' centroids. We observe that similar handshape models share close positions in the space. The presented feature space is indicative and it seems clear when compared to feature spaces of other methods. To support this we compare the feature spaces with the Davies-Boulding index (DBi), which quantifies their quality. In brief, the DBi is the average over all $n$ clusters, of the ratio of intra-cluster distances $\sigma_i$ versus the inter-cluster distance $d_{i,j}$ of $i, j$ clusters, as a measure of their separation: $DBi = \frac{1}{n}\sum_{i=1}^{n} \max_{i \neq j}(\frac{\sigma_i + \sigma_j}{d_{i,j}})$ (Davies and Bouldin, 1979). Figure 13 presents the results. The reported

| Data Set | # HSIds | CD | Occ. | Feat. Method | Avg.Acc.% | Std. |
|---|---|---|---|---|---|---|
| DS-1 | 18 | Table. 2 | ✗ | Aff-SAM | **93.7** | 1.5 |
| | | | | DS-SAM | 93.4 | 1.6 |
| | | | | DTS-SAM | 89.2 | 1.9 |
| DS-1-extend | 24 | 'D-H' | ✗ | Aff-SAM | **77.2** | 1.6 |
| | | | | DS-SAM | 74 | 2.3 |
| | | | | DTS-SAM | 67 | 1.4 |
| DS-2 | 42 | Table. 2 | ✓ | Aff-SAM | **74.9** | 0.9 |
| | | | | DS-SAM | 66.1 | 1.1 |
| | | | | DTS-SAM | 62.7 | 1.4 |

Table 3: Experiments overview with selected average overall results over different main feature extraction methods and experimental cases of DS and CD experiments, with occlusion or not (see Section 7.1). CD: class dependency. Occ.: indicates whether the dataset includes occlusion cases. # HSIds: the number of HSId employed, Avg.Acc.: average classification accuracy, Std.: standard deviation of the classification accuracy.

indices are for varying CD field, that is the orientation parameters on which the handshape models are dependent or not (as discussed in Section 7.1) and are referred in Table 2. We observe that the DBi's for the Aff-SAM features are lower that is the classes are more compact and more separable, compared to the other cases. The closest DBi's are these of DS-SAM. In addition, the proposed features show stable performance over experiments w.r.t. class-dependency, indicating robustness to some amount of pose variation.

### 7.3 Results of Classification Experiments

We next show average classification accuracy results after 5-fold cross-validation for each experiment. together with the standard deviation of the accuracies. The experiments consist of 1) Class dependency and Feature variation for non-occlusion cases and 2) Class dependency and Feature variation for both occlusion and non-occlusion cases. Table 3 presents averages as well as comparisons with other features for the three main experimental data sets discussed. The averages are over all cross-validation cases, and over the multiple experiments w.r.t. class dependency, where applicable. For instance, in the first block for the case 'DS-1', that is non-occluded data from the dataset DS-1, the average is taken over all cases of class dependency experiments as described in Table 2. For the 'DS-1-extend' case, the average is taken over the D-H class dependency experiment, since we want to increase the variability within each class.

#### 7.3.1 FEATURE COMPARISONS FOR NON-OCCLUDED CASES

Next, follow comparisons by employing the referred feature extraction approaches, for two cases of data sets, while accounting for non-occluded cases.

#### 7.3.2 DATA SET DS-1

In Figure 14 we compare the employed methods, while varying the models' dependency w.r.t. the annotation parameters (x axis). We employ the DS-1 data set, consisting of 18 handshape types

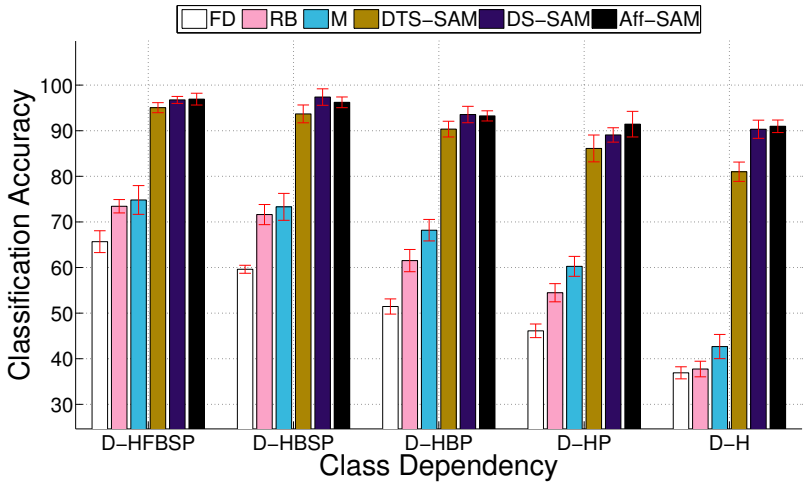



Figure 14: Classification experiments for non-occlusion cases, dataset DS-1. Classification Accuracy for varying experiments (x-axis) that is the dependency of each class w.r.t. the annotation parameters [H,F,B,S,P] and the feature employed (legend). For the numbers of classes per experiment see Table 4.

| Class dependency Parameters | D-HFSBP | D-HSBP | D-HBP | D-HP | D-H |
|---|---|---|---|---|---|
| # Classes | 34 | 33 | 33 | 31 | 18 |

Table 4: Number of classes for each type of class dependency (classification experiments for Non-Occlusion cases).

from non-occlusion cases. The number of classes are shown in Table 4. In Figure 14 we depict the performance over the different methods and models' dependency. At the one extent (that is 'D-HFBSP') we trained one GMM model for each different combination of the handshape configuration parameters (H,F,B,S,P). Thus, the trained models were dependent on the 3D handshape pose and so are the classes for the classification (34 different classes). In the other extent ('D-H') we trained one GMM model for each HSId thus the trained models were independent to the 3D handshape pose and so are the classes for the classification (18 different classes). Furthermore we observe that the proposed method outperforms the baseline methods (FD, RB, M) and DTS-SAM. However the classification performance of Aff-SAM and DS-SAM methods is quite close in some cases. This is due to the easy classification task (small number of HSIds and 3D pose variability and non-occlusion cases). The classification performance of the proposed method is slightly affected from the decrease of the dependency on the annotation parameters. This strengthens our previous observation that the proposed method can handle small pose variations. For a results' overview see Table 3 (DS-1 block). The averages are across all pose-dependency cases.

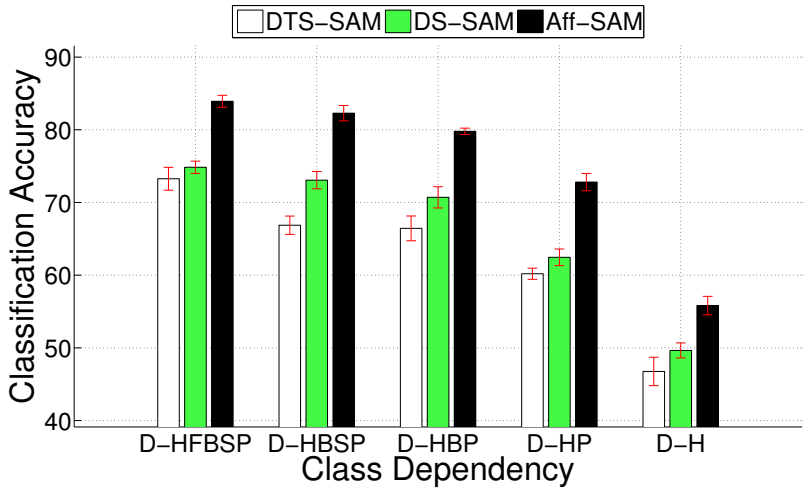



Figure 15: Classification experiments for both occluded and non-occluded cases. Classification Accuracy by varying the dependency of each class w.r.t. to the annotation parameters [H,F,B,S,P] (x-axis) and the feature employed (legend). For the numbers of classes per experiment see Table 5.

### 7.3.3 DATA SET DS-1-EXTEND

This is an extension of DS-1 and consists of 24 different HSIds with much more 3D handshape pose variability. We trained models independent to the 3D handshape pose. Thus, these experiments refer to the D-H case. Table 3 (DS-1-extend block) shows average results for the three competitive methods. We observe that Aff-SAM outperforms both DS-SAM and DTS-SAM achieving average improvements of 3.2% and 10.2% respectively. This indicates the advancement of the Aff-SAM over the other two competitive methods (DS-SAM and DTS-SAM) in more difficult tasks. It also shows that, by incorporating more data with extended variability w.r.t. pose parameters, there is an increase in the average improvements.

| Class dependency Parameters | D-HFSBP | D-HSBP | D-HBP | D-HP | D-H |
|---|---|---|---|---|---|
| # Classes | 100 | 88 | 83 | 72 | 42 |

Table 5: Number of classes for each type of class dependency (classification experiments for Occlusion and Non-Occlusion cases).

### 7.3.4 FEATURE COMPARISONS FOR OCCLUDED AND NON-OCCLUDED CASES

In Figure 15 we vary the models' dependency w.r.t. the annotation parameters similar to Section 7.3.1. However, DS-2 data set consists of 42 handshape HSIds for *both* occlusion and non-occlusion cases. For the number of classes per experiment see Table 5. Aff-SAM outperforms both DS-SAM and DST-SAM obtaining on average 10% performance increase in all cases (Figure 15).

This indicates that Aff-SAM handles handshape classification obtaining decent results even during occlusions. The performance for the other baseline methods is not shown since they cannot handle occlusions and the results are lower. The comparisons with the two more competitive methods show the differential gain due to the *claimed* contributions of the Aff-SAM. By making our models independent to 3D pose orientation, that is,-H, the classification performance decreases. This makes sense since by taking into consideration the occlusion cases the variability of the handshapes' 3D pose increases; as a consequence the classification task is more difficult. Moreover, the classification during occlusions may already include errors at the visual modeling level concerning the estimated occluded handshape. In this experiment, the range of 3D pose variations is larger than the amount handled by the affine transforms of the Aff-SAM.

## 8. Sign Recognition

Next, we evaluate the Aff-SAM approach, on automatic sign recognition experiments, while fusing with movement/position cues, as well as concerning its application on multiple signers. The experiments are applied on data from the GSL lexicon corpus (DictaSign, 2012). By employing the presented framework for tracking and feature extraction (Section 3) we extract the Aff-SAM features (Section 4). These are then employed to construct data-driven subunits as in Roussos et al. (2010b) and Theodorakis et al. (2012), which are further statistically trained. The lexicon corpus contains data from two different signers, A and B. Given the Aff-SAM based models from signer A these are then adapted and fitted to another signer (B) as in Section 5 for which no Aff-SAM models have been trained. The features resulting as a product of the visual level adaptation, are employed next in the recognition experiment. For signer A, the features are extracted from the signer's own model. Note that, there are other aspects concerning signer adaptation during SL recognition, as for instance the manner of signing or the different pronunciations, which are not within the focus of this article.

*GSL Lemmas:* We employ 100 signs from the *GSL lemmas corpus*. These are articulated in isolation with five repetitions each, from two native signers (male and female). The videos have a uniform background and a resolution of 1440x1080 pixels, recorded at 25 fps.

### 8.1 Sub-unit Modeling and Sign Recognition

The SL recognition framework consists of the following: 1) First by employing the movement-position cue we construct dynamic/static SUs based on dynamic and static discrimination (Pitsikalis et al., 2010; Theodorakis et al., 2012). 2) Second we employ the handshape features and the sub-unit construction via clustering of the handshape features (Roussos et al., 2010b). 3) We then create one lexicon for each information cue, that is, movement-position and handshape. For the movement-position lexicon we recompose the constructed dynamic/static SUs, whereas for the Handshape lexicon we recompose the handshape subunits (HSU) to form each sign realization. 4) Next, for the training of the SUs we employ a GMM for the static and handshape subunits and an 5-state HMM for the dynamic subunits. Concerning the training, we employ four realizations for each sign for training and one for testing. 5) Finally, we fuse the movement-position and handshape cues via one possible late integration scheme, that is Parallel HMMs (PaHMMs) (Vogler and Metaxas, 1999).

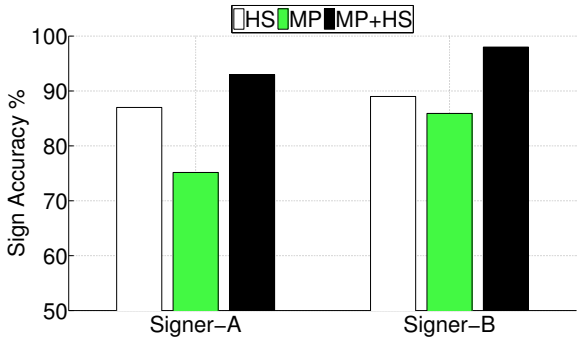



Figure 16: Sign recognition in GSL lemmas corpus employing 100 signs for each signer A and B, and multiple cues: Hanshape (HS), Movement-Position (MP) cue and MP+HS fusion between both via Parallel HMMs.

### 8.2 Sign Recognition Results

In Figure 16 we present the sign recognition performance on the GSL lemmas corpus employing 100 signs from two signers, A and B, while varying the cues employed: movement-position (MP), handshape (HS) recognition performance and the fusion of both MP+HS cues via PaHMMs. For both signers A and B, handshape-based recognition outperforms the one of movement-position cue. This is expected, and indicates that handshape cue is crucial for sign recognition. Nevertheless, the main result we focus is the following: The sign recognition performance in Signer-B is similar to Signer-A, where the Aff-SAM model has been trained. Thus by applying the affine adaptation procedure and employing only a small development set, as presented in Section 5 we can extract reliable handshape features for multiple signers. As a result, when both cues are employed, and for both signers, the recognition performance increases, leading to a 15% and 7.5% absolute improvement w.r.t. the single cues respectively.

## 9. Conclusions

In this paper, we propose a new framework that incorporates dynamic affine-invariant Shape - Appearance modeling and feature extraction for handshape classification. The proposed framework leads to the extraction of effective features for hand configurations. The main contributions of this work are the following: 1) We employ Shape-Appearance hand images for the representation of the hand configurations. These images are modeled with a linear combination of eigenimages followed by an affine transformation, which effectively accounts for some 3D hand pose variations. 2) In order to achieve robustness w.r.t. occlusions, we employ a regularized fitting of the SAM that exploits prior information on the handshape and its dynamics. This process outputs an accurate tracking of the hand as well as descriptive handshape features. 3) We introduce an affine-adaptation for different signers than the signer that was used to train the model. 4) All the above features are integrated in a statistical handshape classification GMM and a sign recognition HMM-based system.

The overall visual feature extraction and classification framework is evaluated on classification experiments as well as on sign recognition experiments. These explore multiple tasks of gradual difficulty in relation to the orientation parameters, as well as both occlusion and non-occlusion

cases. We compare with existing baseline features as well as with more competitive features, which are implemented as simplifications of the proposed SAM method. We investigate the quality of the feature spaces and evaluate the compactness-separation of the different features in which the proposed features show superiority. The Aff-SAM features yield improvements in classification accuracy too. For the non-occlusion cases, these are on average 35% over the baseline methods (FD, RB, M) and 3% over the most competitive SAM methods (DS-SAM, DST-SAM). Furthermore, when we also consider the occlusion cases, the improvements in classification accuracy are on average 9.7% over the most competitive SAM methods (DS-SAM, DST-SAM). Although DS-SAM yields similar performance in some cases, it under-performs in the more difficult and extended data set classification tasks. On the task of sign recognition for a 100-sign lexicon of GSL lemmas, the approach is evaluated via handshape subunits and also fused with movement-position cues, leading to promising results. Moreover, it is shown to have similar results, even if we do not train an explicit signer dependent Aff-SA model, given the introduction of the affine-signer adaptation component. In this way, the approach can be easily applicable to multiple signers.

To conclude with, given that handshape is among the main sign language phonetic parameters, we address issues that are indispensable for automatic sign language recognition. Even though the framework is applied on SL data, its application is extendable on other gesture-like data. The quantitative evaluation and the intuitive results presented show the perspective of the proposed framework for further research.

## Acknowledgments

This research work was supported by the EU under the research program Dictasign with grant FP7-ICT-3-231135. A. Roussos was also supported by the ERC Starting Grant 204871-HUMANIS.

## Appendix A. Details about the Regularized Fitting Algorithm

We provide here details about the algorithm of the regularized fitting of the shape-appearance model. The total energy $E(\lambda, p)$ that is to be minimized can be written as (after a multiplication with $N_M$ that does not affect the optimum parameters):

$$\begin{aligned} J(\lambda,p) = &\sum_x \left\{ A_0(x) + \sum_{i=1}^{N_c} \lambda_i A_i(x) - f(W_p(x)) \right\}^2 + \\ &\frac{N_M}{N_c} \left( w_S \left\| \lambda - \lambda_0 \right\|^2_{\Sigma_\lambda} + w_D \left\| \lambda - \lambda^e \right\|^2_{\Sigma_{\varepsilon_\lambda}} \right) + \\ &\frac{N_M}{N_p} \left( w_S \left\| p - p_0 \right\|^2_{\Sigma_p} + w_D \left\| p - p^e \right\|^2_{\Sigma_{\varepsilon_p}} \right) . \end{aligned} \tag{4}$$

If $\sigma_{\lambda_i}$, $\sigma_{\tilde{p}_i}$ are the standard deviations of the components of the parameters $\lambda$, $\tilde{p}$ respectively and $\sigma_{\varepsilon_{\lambda,i}}$, $\sigma_{\varepsilon_{\tilde{p},i}}$ are the standard deviations of the components of the parameters' prediction errors $\varepsilon_\lambda$, $\varepsilon_{\tilde{p}}$, then the corresponding covariance matrices $\Sigma_\lambda$, $\Sigma_{\tilde{p}}$, $\Sigma_{\varepsilon_\lambda}$, $\Sigma_{\varepsilon_{\tilde{p}}}$, which are diagonal, can be written as:

$$\begin{aligned} &\Sigma_\lambda = \mathrm{diag}(\sigma^2_{\lambda_1}, \ldots, \sigma^2_{\lambda_{N_c}}), \Sigma_{\tilde{p}} = \mathrm{diag}(\sigma^2_{\tilde{p}_1}, \ldots, \sigma^2_{\tilde{p}_{N_c}}), \\ &\Sigma_{\varepsilon_\lambda} = \mathrm{diag}(\sigma^2_{\varepsilon_{\lambda,1}}, \ldots, \sigma^2_{\varepsilon_{\lambda,N_c}}), \Sigma_{\varepsilon_{\tilde{p}}} = \mathrm{diag}(\sigma^2_{\varepsilon_{\tilde{p},1}}, \ldots, \sigma^2_{\varepsilon_{\tilde{p},N_p}}). \end{aligned}$$

The squared norms of the prior terms in Equation (4) are thus given by:

$$\|\lambda-\lambda_0\|^2_{\Sigma_\lambda}=\sum_{i=1}^{N_c}\left(\frac{\lambda_i}{\sigma_{\lambda_i}}\right)^2,$$

$$\|\lambda-\lambda^e\|^2_{\Sigma_{\varepsilon_\lambda}}=\sum_{i=1}^{N_c}\left(\frac{\lambda_i-\lambda_i^e}{\sigma_{\varepsilon_{\lambda,i}}}\right)^2,$$

$$\|p-p_0\|^2_{\Sigma_p}=(p-p_0)^T U_p\Sigma_{\tilde{p}}^{-1}U_p^T(p-p_0)=\|\widetilde{p}\|^2_{\Sigma_{\tilde{p}}}=\sum_{i=1}^{N_p}\left(\frac{\tilde{p}_i}{\sigma_{\tilde{p}_i}}\right)^2,$$

$$\|p-p^e\|^2_{\Sigma_{\varepsilon_p}}=\|\widetilde{p}-\widetilde{p}^e\|^2_{\Sigma_{\varepsilon_{\tilde{p}}}}=\sum_{i=1}^{N_p}\left(\frac{\tilde{p}_i-\tilde{p}_i^e}{\sigma_{\varepsilon_{\tilde{p},i}}}\right)^2.$$

Therefore, if we set:

$$m_1=\sqrt{w_S N_M/N_c}\,,\ m_2=\sqrt{w_D N_M/N_c}\,,$$
$$m_3=\sqrt{w_S N_M/N_p}\,,\ m_4=\sqrt{w_D N_M/N_p}\,,$$

the energy in Equation (4) takes the form:

$$J(\lambda,p)=\sum_x\left\{A_0(x)+\sum_{i=1}^{N_c}\lambda_i A_i(x)-f(W_p(x))\right\}^2+\sum_{i=1}^{N_G}G_i^2(\lambda,p)\,, \tag{5}$$

with $G_i(\lambda,p)$ being $N_G=2N_c+2N_p$ prior functions defined by:

$$G_i(\lambda,p)=\begin{cases} m_1\frac{\lambda_i}{\sigma_{\lambda_i}}, & 1\le i\le N_c \\ m_2\frac{\lambda_j-\lambda_j^e}{\sigma_{\varepsilon_{\lambda,j}}},\ j=i-N_c, & N_c+1\le i\le 2N_c \\ m_3\frac{\tilde{p}_j}{\sigma_{\tilde{p}_j}},\ j=i-2N_c, & 2N_c+1\le i\le 2N_c+N_p \\ m_4\frac{\tilde{p}_j-\tilde{p}_j^e}{\sigma_{\varepsilon_{\tilde{p},j}}},\ j=i-2N_c-N_p, & 2N_c+N_p+1\le i\le 2N_c+2N_p \end{cases}. \tag{6}$$

Each component $\tilde{p}_j$, $j=1,\ldots,N_p$, of the re-parametrization of $p$ can be written as:

$$\tilde{p}_j=v_{\tilde{p}_j}^T(p-p_0)\,, \tag{7}$$

where $v_{\tilde{p}_j}$ is the $j$-th column of $U_p$, that is the eigenvector of the covariance matrix $\Sigma_p$ that corresponds to the $j$-th principal component $\tilde{p}_j$.

In fact, the energy $J(\lambda,p)$, Equation (5), for general prior functions $G_i(\lambda,p)$, has exactly the same form as the energy that is minimized by the algorithm of Baker et al. (2004). Next, we describe this algorithm and then we specialize it in the specific case of our framework.

### A.1 Simultaneous Inverse Compositional Algorithm with a Prior

We briefly present here the algorithm *simultaneous inverse compositional with a prior* (SICP) (Baker et al., 2004). This is a *Gauss-Newton* algorithm that finds a local minimum of the energy $J(\lambda, p)$ (5) for general cases of prior functions $G_i(\lambda, p)$ and warps $W_p(x)$ that are controlled by some parameters $p$.

The algorithm starts from some initial estimates of $\lambda$ and $p$. Afterwards, in every iteration, the previous estimates of $\lambda$ and $p$ are updated to $\lambda'$ and $p'$ as follows. It is considered that a vector $\Delta\lambda$ is added to $\lambda$:

$$\lambda' = \lambda + \Delta\lambda \tag{8}$$

and a warp with parameters $\Delta p$ is applied to the synthesized image $A_0(x) + \sum \lambda_i A_i(x)$. As an approximation, the latter is taken as equivalent to updating the warp parameters from $p$ to $p'$ by composing $W_p(x)$ with the inverse of $W_{\Delta p}(x)$ :

$$W_{p'} = W_p \circ W_{\Delta p}^{-1} . \tag{9}$$

From the above relation, given that $p$ is constant, $p'$ can be expressed as a $\mathbb{R}^{N_p} \to \mathbb{R}^{N_p}$ function of $\Delta p$, $p' = p'(\Delta p)$ , with $p'(\Delta p = 0) = p$. Further, $p'(\Delta p)$ is approximated with a first order Taylor expansion around $\Delta p = 0$:

$$p'(\Delta p) = p + \frac{\partial p'}{\partial \Delta p}\Delta p . \tag{10}$$

where $\frac{\partial p'}{\partial \Delta p}$ is the Jacobian of the function $p'(\Delta p)$, which generally depends on $\Delta p$.

Based on the aforementioned type of updates of $\lambda$ and $p$ as well as the considered approximations, the values $\Delta\lambda$ and $\Delta p$ are specified by minimizing the following energy:

$$\begin{aligned} F(\Delta\lambda, \Delta p) = \sum_x \Bigg\{ & A_0\big(W_{\Delta p}(x)\big) + \sum_{i=1}^{N_c} (\lambda_i + \Delta\lambda_i) A_i\big(W_{\Delta p}(x)\big) \\ & - f\big(W_p(x)\big) \Bigg\}^2 + \sum_{i=1}^{N_G} G_i^2 \left( \lambda + \Delta\lambda, p + \frac{\partial p'}{\partial \Delta p}\Delta p \right) , \end{aligned}$$

simultaneously with respect to $\Delta\lambda$ and $\Delta p$. By applying first order Taylor approximations on the two terms of the above energy $F(\lambda, p)$, one gets:

$$\begin{aligned} F(\Delta\lambda, \Delta p) \approx & \sum_x \left\{ E_{sim}(x) + SD_{sim}(x) \begin{pmatrix} \Delta\lambda \\ \Delta p \end{pmatrix} \right\}^2 + \\ & \sum_{i=1}^{N_G} \left\{ G_i(\lambda, p) + SD_{G_i} \begin{pmatrix} \Delta\lambda \\ \Delta p \end{pmatrix} \right\}^2 , \end{aligned} \tag{11}$$

where $E_{sim}(x)$ is the image of reconstruction error evaluated at the model domain:

$$E_{sim}(x) = A_0(x) + \sum_{i=1}^{N_c} \lambda_i A_i(x) - f\big(W_p(x)\big)$$

and $SD_{sim}(x)$ is a vector-valued "steepest descent" image with $N_c + N_p$ channels, each one of them corresponding to a specific component of the parameter vectors $\lambda$ and $p$:

$$SD_{sim}(x) = \left[ A_1(x), \ldots, A_{N_c}(x), \left( \nabla A_0(x) + \sum_{i=1}^{N_c} \lambda_i \nabla A_i(x) \right) \frac{\partial W_p(x)}{\partial p} \right] , \tag{12}$$

where the gradients $\nabla A_i(x) = \left[\frac{\partial A_i}{\partial x_1}, \frac{\partial A_i}{\partial x_2}\right]$ are considered as row vector functions. Also $SD_{G_i}$, for each $i = 1,..,N_G$, is a row vector with dimension $N_c + N_p$ that corresponds to the steepest descent direction of the prior term $G_i(\lambda, p)$:

$$SD_{G_i} = \left(\frac{\partial G_i}{\partial \lambda}, \frac{\partial G_i}{\partial p}\frac{\partial p'}{\partial \Delta p}\right) . \tag{13}$$

The approximated energy $F(\lambda, p)$ (11) is quadratic with respect to both $\Delta\lambda$ and $\Delta p$, therefore the minimization can be done analytically and leads to the following solution:

$$\begin{pmatrix} \Delta\lambda \\ \Delta p \end{pmatrix} = -H^{-1}\left[\sum_x SD^T_{sim}(x)E_{sim}(x) + \sum_{i=1}^{N_G} SD^T_{G_i}G_i(\lambda, p)\right], \tag{14}$$

where $H$ is the matrix (which approximates the Hessian of $F$):

$$H = \sum_x SD^T_{sim}(x)SD_{sim}(x) + \sum_{i=1}^{N_G} SD^T_{G_i}SD_{G_i} .$$

In conclusion, in every iteration of the SICP algorithm, the Equation (14) is applied and the parameters $\lambda$ and $p$ are updated using Equations (8) and (10). This process terminates when a norm of the update vector $\begin{pmatrix} \Delta\lambda \\ \Delta p \end{pmatrix}$ falls below a relatively small threshold and then it is considered that the process has converged.

#### A.1.1 COMBINATION WITH LEVENBERG-MARQUARDT ALGORITHM

In the algorithm described above, there is no guarantee that the original energy (5), that is the objective function before any approximation, decreases in every iteration; it might increase if the involved approximations are not accurate. Therefore, following Baker and Matthews (2002), we use a modification of this algorithm by combining it with the *Levenberg-Marquardt* algorithm: In Equation (14) that specifies the updates, we replace the Hessian approximation $H$ by $H + \delta\,\mathrm{diag}(H)$, where $\delta$ is a positive weight and $\mathrm{diag}(H)$ is the diagonal matrix that contains the diagonal elements of $H$. This corresponds to an interpolation between the updates given by the Gauss-Newton algorithm and weighted gradient descent. As $\delta$ increases, the algorithm has a behavior closer to gradient descent, which means that from the one hand is slower but from the other hand yields updates that are more reliable, in the sense that the energy will eventually decrease for sufficiently large $\delta$.

In every iteration, we specify the appropriate weight $\delta$ as follows. Starting from setting $\delta$ to $1/10$ of its value in the previous iteration (or from $\delta = 0.01$ if this is the first iteration), we compute the updates $\Delta\lambda$ and $\Delta p$ using the Hessian approximation $H + \delta\,\mathrm{diag}(H)$ and then evaluate the original energy (5). If the energy has decreased we keep the updates and finish the iteration. If the energy has increased, we set $\delta \to 10\,\delta$ and try again. We repeat that step until the energy decreases.

### A.2 Specialization in the Current Framework

In this section, we derive the SICP algorithm for the special case that concerns our method. This case arises when 1) the general warps $W_p(x)$ are specialized to affine transforms and 2) the general prior functions $G_i(\lambda, p)$ are given by Equation (6).

### A.2.1 THE CASE OF AFFINE TRANSFORMS

In our framework, the general warps $W_p(x)$ of the SICP algorithm are specialized to affine transforms with parameters $p = (p_1 \cdots p_6)$ that are defined by:

$$W_p(x,y) = \begin{pmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

In this special case, which is analyzed also in Baker et al. (2004), the Jacobian $\frac{\partial W_p(x)}{\partial p}$ that is used in Equation (12) is given by:

$$\frac{\partial W_p(x)}{\partial p} = \begin{pmatrix} x_1 & 0 & x_2 & 0 & 1 & 0 \\ 0 & x_1 & 0 & x_2 & 0 & 1 \end{pmatrix}.$$

The restriction to affine transforms implies also a special form for the Jacobian $\frac{\partial p'}{\partial \Delta p}$ that is used in Equation (13). More precisely, as described in Baker et al. (2004), a first order Taylor approximation is first applied to the inverse warp $W_{\Delta p}^{-1}$ and yields $W_{\Delta p}^{-1} \approx W_{-\Delta p}$. Afterwards, based on Equation (9) and the fact that the parameters of a composition $W_r = W_p \circ W_q$ of two affine transforms are given by:

$$r = \begin{pmatrix} p_1 + q_1 + p_1 q_1 + p_3 q_2 \\ p_2 + q_2 + p_2 q_1 + p_4 q_2 \\ p_3 + q_3 + p_1 q_3 + p_3 q_4 \\ p_4 + q_4 + p_2 q_3 + p_4 q_4 \\ p_5 + q_5 + p_1 q_5 + p_3 q_6 \\ p_6 + q_6 + p_2 q_5 + p_4 q_6 \end{pmatrix},$$

the function $p'(\Delta p)$ (10) is approximated as:

$$p'(\Delta p) = \begin{pmatrix} p_1 - \Delta p_1 - p_1 \Delta p_1 - p_3 \Delta p_2 \\ p_2 - \Delta p_2 - p_2 \Delta p_1 - p_4 \Delta p_2 \\ p_3 - \Delta p_3 - p_1 \Delta p_3 - p_3 \Delta p_4 \\ p_4 - \Delta p_4 - p_2 \Delta p_3 - p_4 \Delta p_4 \\ p_5 - \Delta p_5 - p_1 \Delta p_5 - p_3 \Delta p_6 \\ p_6 - \Delta p_6 - p_2 \Delta p_5 - p_4 \Delta p_6 \end{pmatrix}.$$

Therefore, its Jacobian is given by:

$$\frac{\partial p'}{\partial \Delta p} = -\begin{pmatrix} 1+p_1 & p_3 & 0 & 0 & 0 & 0 \\ p_2 & 1+p_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1+p_1 & p_3 & 0 & 0 \\ 0 & 0 & p_2 & 1+p_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1+p_1 & p_3 \\ 0 & 0 & 0 & 0 & p_2 & 1+p_4 \end{pmatrix}.$$

### A.2.2 SPECIFIC TYPE OF PRIOR FUNCTIONS

Apart from the restriction to affine transforms, in the proposed framework of the regularized shape-appearance model fitting, we have derived the specific formulas of Equation (6) for the prior functions $G_i(\lambda, p)$ of the energy $J(\lambda, p)$ in Equation (5). Therefore, in our case, their partial derivatives,

which are involved in the above described SICP algorithm (see Equation (13)), are specialized as follows:

$$\frac{\partial G_i}{\partial p} \overset{(7)}{=} \begin{cases} 0\,, & 1 \le i \le 2N_c \\ \frac{m_3}{\sigma_{\tilde{p}_j}} v_{\tilde{p}_j}^T\,,\ j = i - 2N_c\,, & 2N_c+1 \le i \le 2N_c+N_p \\ \frac{m_4}{\sigma_{\varepsilon_{\tilde{p},j}}} v_{\tilde{p}_j}^T\,,\ j = i - 2N_c - N_p\,, & 2N_c+N_p+1 \le i \le 2N_c+2N_p \end{cases},$$

$$\frac{\partial G_i}{\partial \lambda} = \begin{cases} \frac{m_1}{\sigma_{\lambda_i}} e_i^T\,, & 1 \le i \le N_c \\ \frac{m_2}{\sigma_{\varepsilon_{\lambda,j}}} e_j^T\,,\ j = i - N_c\,, & N_c+1 \le i \le 2N_c \\ 0\,, & 2N_c+1 \le i \le 2N_c+2N_p \end{cases},$$

where $e_i$, $1 \le i \le N_c$, is the $i$-th column of the $N_c \times N_c$ identity matrix.

## References


U. Agris, J. Zieren, U. Canzler, B. Bauer, and K. F. Kraiss. Recent developments in visual sign language recognition. *Universal Access in the Information Society*, 6:323–362, 2008.

T. Ahmad, C.J. Taylor, and T.F. Lanitis, A. Cootes. Tracking and recognising hand gestures, using statistical shape models. *Image and Visual Computing*, 15(5):345–352, 1997.

A. Argyros and M. Lourakis. Real time tracking of multiple skin-colored objects with a possibly moving camera. In *Proceedings of the European Conference on Computer Vision*, 2004.

V. Athitsos and S. Sclaroff. An appearance-based framework for 3d hand shape classification and camera viewpoint estimation. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 45–52, 2002.

S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework: Part 1. Technical report, Carnegie Mellon University, 2002.

S. Baker, R. Gross, and I. Matthews. Lucas-kanade 20 years on: A unifying framework: Part 4. Technical report, Carnegie Mellon University, 2004.

B. Bauer and K. F. Kraiss. Towards an automatic sign language recognition system using subunits. In *Proceedings of the International Gesture Workshop*, volume 2298, pages 64–75, 2001.

H. Birk, T.B. Moeslund, and C.B. Madsen. Real-time recognition of hand alphabet gestures using principal component analysis. In *Proceedings of the Scandinavian Conference Image Analysis*, 1997.

A. Blake and M. Isard. *Active Contours*. Springer, 1998.

R. Bowden and M. Sarhadi. A nonlinear model of shape and motion for tracking fingerspelt american sign language. *Image and Visual Computing*, 20:597–607, 2002.

P. Buehler, M. Everingham, and A. Zisserman. Learning sign language by watching TV (using weakly aligned subtitles). In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009.

J. Cai and A. Goshtasby. Detecting human faces in color images. *Image and Visual Computing*, 18: 63–75, 1999.

F.-S. Chen, C.-M. Fu, and C.-L. Huang. Hand gesture recognition using a real-time tracking method and hidden markov models. *Image and Visual Computing*, 21(8):745–758, 2003.

S. Conseil, S. Bourennane, and L. Martin. Comparison of Fourier descriptors and Hu moments for hand posture recognition. In *Proceedings of the European Conference on Signal Processing*, 2007.

T.F. Cootes and C.J. Taylor. Statistical models of appearance for computer vision. Technical report, University of Manchester, 2004.

Y. Cui and J. Weng. Appearance-based hand sign recognition from intensity image sequences. *Computer Vision and Image Understanding*, 78(2):157–176, 2000.

L. Davies, David and W. Bouldin, Donald. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:224 – 227, April 1979.

J.-W. Deng and H.T. Tsui. A novel two-layer PCA/MDA scheme for hand posture recognition. In *Proceedings of the International Conference on Pattern Recognition*, volume 1, pages 283–286, 2002.

DictaSign. Greek sign language corpus. http://www.sign-lang.uni-hamburg.de/dicta-sign/portal, 2012.

L. Ding and A. M. Martinez. Modelling and recognition of the linguistic components in american sign language. *Image and Visual Computing*, 27(12):1826 – 1844, 2009.

P. Dreuw, J. Forster, T. Deselaers, and H. Ney. Efficient approximations to model-based joint tracking and recognition of continuous sign language. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, Sep. 2008.

I.L. Dryden and K.V. Mardia. *Statistical Shape Analysis*. John Wiley and Sons, 1998.

W. Du and J. Piater. Hand modeling and tracking for video-based sign language recognition by robust principal component analysis. In *Proceedings of the ECCV Workshop on Sign, Gesture and Activity*, September 2010.

A. Farhadi, D. Forsyth, and R. White. Transfer learning in sign language. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

H. Fillbrandt, S. Akyol, and K.-F. Kraiss. Extraction of 3D hand shape and posture from images sequences from sign language recognition. In *Proceedings of the International Workshop on Analysis and Modeling of Faces and Gestures*, pages 181–186, 2003.

R. Gross, I. Matthews, and S. Baker. Generic vs. person specific active appearance models. *Image and Visual Computing*, 23(12):1080–1093, 2005.

T. Hanke. HamNoSys Representing sign language data in language resources and language processing contexts. In *Proceedings of the International Conference on Language Resources and Evaluation*, 2004.

M.-K. Hu. Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, 8(2):179–187, February 1962.

C.-L. Huang and S.-H. Jeng. A model-based hand gesture recognition system. *Machine Vision and Application*, 12(5):243–258, 2001.

P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recognition*, 40(3):1106–1122, Mar. 2007.

E. Learned-Miller. Data driven image models through continuous joint alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(2):236–250, 2005.

S. Liwicki and M. Everingham. Automatic recognition of fingerspelled words in British sign language. In *Proceedings of the CVPR Workshop on Human Communicative Behavior Analysis*, 2009.

P. Maragos. *Morphological Filtering for Image Enhancement and Feature Detection*, chapter The Image and Video Processing Handbook. Elsevier, 2005.

I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135–164, 2004.

C. Neidle. Signstream annotation: Addendum to conventions used for the american sign language linguistic research project. Technical report, 2007.

C. Neidle and C. Vogler. A new web interface to facilitate access to corpora: development of the ASLLRP data access interface. In *Proceedings of the International Conference on Language Resources and Evaluation*, 2012.

E.J. Ong, H. Cooper, N. Pugeault, and R. Bowden. Sign language recognition using sequential pattern trees. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2200–2207. IEEE, 2012.

Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2010.

V. Pitsikalis, S. Theodorakis, and P. Maragos. Data-driven sub-units and modeling structure for continuous sign language recognition with multiple cues. In *LREC Workshop Repr. & Proc. SL: Corpora and SL Technologies*, 2010.

L. R. Rabiner and R.W. Schafer. Introduction to digital speech processing. *Foundations and Trends in Signal Processing*, 1(1-2):1–194, 2007.

A. Roussos, S. Theodorakis, V. Pitsikalis, and P. Maragos. Affine-invariant modeling of shape-appearance images applied on sign language handshape classification. In *Proceedings of the International Conference on Image Processing*, Sep. 2010a.

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 ECCV Workshop on Sign, Gesture and Activity*, September 2010b.

J. Sherrah and S. Gong. Resolving visual uncertainty and occlusion through probabilistic reasoning. In *Proceedings of the British Machine Vision Conference*, pages 252–261, 2000.

P. Soille. *Morphological Image Analysis: Principles and Applications*. Springer, 2004.

T. Starner, J. Weaver, and A. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375, Dec. 1998.

B. Stenger, A. Thayananthan, P.H.S Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9): 1372–1384, Sep. 2006.

G.J. Sweeney and A.C. Downton. Towards appearance-based multi-channel gesture recognition. In *Proceedings of the International Gesture Workshop*, pages 7–16, 1996.

N. Tanibata, N. Shimada, and Y. Shirai. Extraction of hand features for recognition of sign language words. In *Proceedings of the International Conference on Vision Interface*, pages 391–398, 2002.

J. Terrillon, M. Shirazi, H. Fukamachi, and S. Akamatsu. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 54–61, 2000.

A. Thangali, J.P. Nash, S. Sclaroff, and C. Neidle. Exploiting phonological constraints for handshape inference in asl video. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 521–528. IEEE, 2011.

S. Theodorakis, V. Pitsikalis, and P. Maragos. Advances in dynamic-static integration of movement and handshape cues for sign language recognition. In *Proceedings of the International Gesture Workshop*, 2011.

S. Theodorakis, V. Pitsikalis, I. Rodomagoulakis, and P. Maragos. Recognition with raw canonical phonetic movement and handshape subunits on videos of continuous sign language. In *Proceedings of the International Conference on Image Processing*, 2012.

M. Viola and M. J. Jones. Fast multi-view face detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2003.

C. Vogler and D. Metaxas. Parallel hidden markov models for american sign language recognition. In *Proceedings of the International Conference on Computer Vision*, volume 1, pages 116–122, 1999.

Y. Wu and T.S. Huang. View-independent recognition of hand postures. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pages 88–94, 2000.

M.-H. Yang, N. Ahuja, and M. Tabb. Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (8):1061–1074, Aug. 2002.

S. Young, D. Kershaw, J. Odell, D. Ollason, V. Woodland, and P. Valtchevand. *The HTK Book*. Entropic Ltd., 1999.

J. Zieren, N. Unger, and S. Akyol. Hands tracking from frontal view for vision-based gesture recognition. In *Pattern Recognition*, LNCS, pages 531–539, 2002.

Journal of Machine Learning Research 14 (2013) 1665-1714 Submitted 9/11; Revised 8/12; Published 7/13

# Nonparametric Sparsity and Regularization

**Lorenzo Rosasco**[*] LROSASCO@MIT.EDU
*DIBRIS*
*University of Genova*
*Via Dodecaneso 35*
*16146 Genova, ITALY*

**Silvia Villa** SILVIA.VILLA@IIT.IT
*Istituto Italiano di Tecnologia*
*Via Morego 30*
*16163 Genova, ITALY*

**Sofia Mosci** SOFIA.MOSCI@UNIGE.IT
*DIBRIS*
*University of Genova*
*Via Dodecaneso 35*
*16146 Genova, ITALY*

**Matteo Santoro** MATTEO.SANTORO@IIT.IT
*Istituto Italiano di Tecnologia*
*Via Morego 30*
*16163 Genova, ITALY*

**Alessandro Verri** ALESSANDRO.VERRI@UNIGE.IT
*DIBRIS*
*University of Genova*
*Via Dodecaneso 35*
*16146 Genova, ITALY*

**Editor:** John Lafferty

## Abstract

In this work we are interested in the problems of supervised learning and variable selection when the input-output dependence is described by a nonlinear function depending on a few variables. Our goal is to consider a sparse nonparametric model, hence avoiding linear or additive models. The key idea is to measure the importance of each variable in the model by making use of partial derivatives. Based on this intuition we propose a new notion of nonparametric sparsity and a corresponding least squares regularization scheme. Using concepts and results from the theory of reproducing kernel Hilbert spaces and proximal methods, we show that the proposed learning algorithm corresponds to a minimization problem which can be provably solved by an iterative procedure. The consistency properties of the obtained estimator are studied both in terms of prediction and selection performance. An extensive empirical analysis shows that the proposed method performs favorably with respect to the state-of-the-art methods.

**Keywords:** sparsity, nonparametric, variable selection, regularization, proximal methods, RKHS

∗. Also at Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy and Massachusetts Institute of Technology, Bldg. 46-5155, 77 Massachusetts Avenue, Cambridge, MA 02139, USA.

©2013 Lorenzo Rosasco, Silvia Villa, Sofia Mosci, Matteo Santoro and Alessandro Verri.

## 1. Introduction

It is now common to see practical applications, for example in bioinformatics and computer vision, where the dimensionality of the data is in the order of hundreds, thousands and even tens of thousands. It is known that learning in such a high dimensional regime is feasible only if the quantity to be estimated satisfies some regularity assumptions (Devroye et al., 1996). In particular, the idea behind, so called, *sparsity* is that the quantity of interest depends only on a few relevant variables (dimensions). In turn, this latter assumption is often at the basis of the construction of interpretable data models, since the relevant dimensions allow for a compact, hence interpretable, representation. An instance of the above situation is the problem of learning from samples a multivariate function which depends only on a (possibly small) subset of *relevant* variables. Detecting such variables is the problem of variable selection.

Largely motivated by recent advances in compressed sensing (Candès and Tao, 2006; Donoho, 2006), the above problem has been extensively studied under the assumption that the function of interest (target function) depends *linearly* to the relevant variables. While a naive approach (trying all possible subsets of variables) would not be computationally feasible it is known that meaningful approximations can be found either by greedy methods (Tropp and Gilbert, 2007), or convex relaxation ($\ell^1$ regularization a.k.a. basis pursuit or LASSO, Tibshirani, 1996; Chen et al., 1999; Efron et al., 2004). In this context efficient algorithms (see Schmidt et al., 2007 and Loris, 2009 and references therein) as well as theoretical guarantees are now available (see Bühlmann and van de Geer, 2011 and references therein). In this paper we are interested into the situation where the target function depends *non-linearly* to the relevant variables. This latter situation is much less understood. Approaches in the literature are mostly restricted to additive models (Hastie and Tibshirani, 1990). In such models the target function is assumed to be a sum of (non-linear) univariate functions. Solutions to the problem of variable selection in this class of models include Ravikumar et al. (2008) and are related to multiple kernel learning (Bach et al., 2004). Higher order additive models can be further considered, encoding explicitly dependence among the variables—for example assuming the target function to be also sum of functions depending on couples, triplets etc. of variables, as in Lin and Zhang (2006) and Bach (2009). Though this approach provides a more interesting, while still interpretable, model, its size/complexity is essentially more than exponential in the initial variables. Only a few works, that we discuss in details in Section 2, have considered notions of sparsity beyond additive models.

In this paper, we propose a new approach based on the idea that the importance of a variable, while learning a non-linear functional relation, can be captured by the corresponding partial derivative. This observation suggests a way to define a new notion of nonparametric sparsity and a corresponding regularizer which favors functions where most partial derivatives are essentially zero. The question is how to make this intuition precise and how to derive a feasible computational learning scheme. The first observation is that, while we cannot measure a partial derivative *everywhere*, we can do it at the training set points and hence design a data-dependent regularizer. In order to derive an actual algorithm we have to consider two further issues: How can we estimate reliably partial derivatives in high dimensions? How can we ensure that the data-driven penalty is sufficiently stable? The theory of reproducing kernel Hilbert spaces (RKHSs) provides us with tools to answer both questions. In fact, partial derivatives in a RKHS are bounded linear functionals and hence have a suitable representation that allows efficient computations. Moreover, the norm in the RKHS provides a natural further regularizer ensuring stable behavior of the empirical,

derivative based penalty. Our contribution is threefold. First, we propose a new notion of sparsity and discuss a corresponding regularization scheme using concept from the theory of reproducing kernel Hilbert spaces. Second, since the proposed algorithm corresponds to the minimization of a convex, but not differentiable functional, we develop a suitable optimization procedure relying on forward-backward splitting and proximal methods. Third, we study properties of the proposed methods both in theory, in terms of statistical consistency, and in practice, by means of an extensive set of experiments.

Some preliminary results have appeared in a short conference version of this paper (Rosasco et al., 2010). With respect to the conference version, the current version contains: the detailed discussion of the derivation of the algorithm with all the proofs, the consistency results of Section 4, an augmented set of experiments and several further discussions. The paper is organized as follows. In Section 3 we discuss our approach and present the main results in the paper. In Section 4 we discuss the computational aspects of the method. In Section 5 we prove consistency results. In Section 6 we provide an extensive empirical analysis. In Section 7 we conclude with a summary of our study and a discussion of future work. Finally, a list of symbols and notations can be found at the end of the paper.

## 2. Problem Setting and Previous Work

Given a training set $\mathbf{z}_n = (\mathbf{x}, \mathbf{y}) = (x_i, y_i)_{i=1}^n$ of input output pairs, with $x_i \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y_i \in \mathcal{Y} \subseteq \mathbb{R}$, we are interested into learning about the functional relationship between input and output. More precisely, in statistical learning the data are assumed to be sampled identically and independently from a probability measure $\rho$ on $\mathcal{X} \times \mathcal{Y}$ so that if we measure the error by the square loss function, the regression function $f_\rho(x) = \int y d\rho(x,y)$ minimizes the expected risk $\mathcal{E}(f) = \int (y - f(x))^2 d\rho(x,y)$.

Finding an estimator $\hat{f}$ of $f_\rho$ from finite data is possible, if $f_\rho$ satisfies some suitable prior assumption (Devroye et al., 1996). In this paper we are interested in the case where the regression function is *sparse* in the sense that it depends only on a subset $R_\rho$ of the possible $d$ variables. Estimating the set $R_\rho$ of *relevant* variables is the problem of variable selection.

### 2.1 Linear and Additive Models

The sparsity requirement can be made precise considering linear functions $f(x) = \sum_{a=1}^d \beta_a x^a$ with $x = (x^1, \ldots, x^d)$. In this case the sparsity of a function is quantified by the so called *zero-norm* $\Omega_0(f) = \#\{a = 1, \ldots, d \mid \beta_a \neq 0\}$. The zero norm, while natural for variable selection, does not lead to efficient algorithms and is often replaced by the $\ell^1$ norm, that is $\Omega_1(f) = \sum_{a=1}^d |\beta_a|$. This approach has been studied extensively and is now fairly well understood, see Bühlmann and van de Geer (2011) and references therein. Regularization with $\ell^1$ regularizers, obtained by minimizing

$$\widehat{\mathcal{E}}(f) + \lambda \Omega_1(f), \qquad \widehat{\mathcal{E}}(f) = \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2,$$

can be solved efficiently and, under suitable conditions, provides a solution close to that of the zero-norm regularization.

The above scenario can be generalized to additive models $f(x) = \sum_{a=1}^d f_a(x^a)$, where $f_a$ are univariate functions in some (reproducing kernel) Hilbert spaces $\mathcal{H}_a$, $a = 1, \ldots, d$. In this case, $\Omega_0(f) = \#\{a \in \{1, \ldots, d\} \ : \ \|f_a\| \neq 0\}$ and $\Omega_1(f) = \sum_{a=1}^d \|f_a\|$ are the analogous of the zero and

$\ell^1$ norm, respectively. This latter setting, related to multiple kernel learning (Bach et al., 2004; Bach, 2008), has been considered for example in Ravikumar et al. (2008), see also Koltchinskii and Yuan (2010) and references therein. Considering additive models limits the way in which the variables can interact. This can be partially alleviated considering higher order terms in the model as it is done in ANOVA decomposition (Wahba et al., 1995; Gu, 2002). More precisely, we can add to the simplest additive model functions of couples $f_{a,b}(x^a,x^b)$, triplets $f_{a,b,c}(x^a,x^b,x^c)$, etc. of variables—see Lin and Zhang (2006). For example one can consider functions of the form $f(x) = \sum_{a=1}^d f_a(x^a) + \sum_{a<b} f_{a,b}(x^a,x^b)$. In this case the analogous to the zero and $\ell^1$ norms are $\Omega_0(f) = \#\{a = 1,\dots,d \ : \ \|f_a\| \neq 0\} + \#\{(a,b) \ : \ a < b, \ \|f_{a,b}\| \neq 0\}$ and $\Omega_1(f) = \sum_{a=1}^d \|f_a\| + \sum_{a<b} \|f_{a,b}\|$, respectively. Note that in this case sparsity will not be in general with respect to the original variables but rather with respect to the elements in the additive model. Clearly, while this approach provides a more interesting and yet interpretable model, its size/complexity is essentially more than exponential in the number of variables. Some proposed attempts to tackle this problem are based on restricting the set of allowed sparsity patterns and can be found in Bach (2009).

### 2.2 Nonparametric Approaches

The above discussion naturally raises the question:

*What if we are interested into learning and performing variable selection when the functions of interest are not described by an additive model?*

Few papers have considered this question. Here we discuss in some more details Lafferty and Wasserman (2008), Bertin and Lecué (2008), Miller and Hall (2010) and Comminges and Dalalyan (2012), to which we also refer for further references.

The first three papers (Lafferty and Wasserman, 2008; Bertin and Lecué, 2008; Miller and Hall, 2010) follow similar approaches focusing on the pointwise estimation of the regression function and of the relevant variables. The basic idea is to start from a locally linear (or polynomial) pointwise estimator $f_n(x)$ at a point $x$ obtained from the minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \langle w, x_i - x\rangle_{\mathbb{R}^d})^2 K_H(x_i - x) \tag{1}$$

where $K_H$ is a localizing window function depending on a matrix (or a vector) $H$ of smoothing parameters. Different techniques are used to (locally) select variables. In the RODEO algorithm (Lafferty and Wasserman, 2008), the localizing window function depends on one smoothing parameter per variable and the partial derivative of the local estimator with respect to the smoothing parameter is used to select variables. In Bertin and Lecué (2008), selection is considering a *local lasso*, that is an $\ell_1$ to the local empirical risk functional (1). In the LABAVS algorithm discussed in Miller and Hall (2010) several variable selection criterion are discussed including the local lasso, hard thresholding, and backward stepwise approach. The above approaches typically lead to cumbersome computations and do not scale well with the dimensionality of the space and with the number of relevant variables.

Indeed, in all the above works the emphasis is in the theoretical analysis quantifying the estimation error of the proposed methods. It is shown in Lafferty and Wasserman (2008) that the RODEO algorithm is a nearly optimal pointwise estimator of the regression function, under assumption on the marginal distribution and the regression function. These results are further improved in Bertin and Lecué (2008) where optimal rates are derived under milder assumptions and sparsistency (the

recovery of $R_\rho$) is also studied. Uniform error estimates are derived in Miller and Hall (2010) (see Section 2.6 in Miller and Hall (2010) for further discussions and comparison). More recently, an estimator based on the comparison of some well chosen empirical Fourier coefficients to a prescribed significance level is described and studied in Comminges and Dalalyan (2012) where a careful statistical analysis is proposed considering different regimes for $n, d$ and $d^*$, where $d^*$ is the cardinality of $R_\rho$. Finally, in a slightly different context, DeVore et al. (2011) studies the related problem of determining the number of function values at adaptively chosen points that are needed in order to correctly estimate the set of globally relevant variables.

## 3. Sparsity Beyond Linear Models

In this section we present our approach and summarize our main contributions.

### 3.1 Sparsity and Regularization Using Partial Derivatives

Our study starts from the observation that, if a function $f$ is differentiable, the relative importance of a variable at a point $x$ can be captured by the magnitude of the corresponding partial derivative[1]

$$\left|\frac{\partial f}{\partial x^a}(x)\right|.$$

This observation can be developed into a new notion of sparsity and corresponding regularization scheme that we study in the rest of the paper. We note that regularization using derivatives is not new. Indeed, the classical splines (Sobolev spaces) regularization (Wahba, 1990), as well as more modern techniques such as manifold regularization (Belkin and Niyogi, 2008) use derivatives to measure the regularity of a function. Similarly total variation regularization uses derivatives to define regular functions. None of the above methods though allows to capture a notion of sparsity suitable both for learning and variable selection—see Remark 1.

Using partial derivatives to define a new notion of a sparsity and design a regularizer for learning and variable selection requires considering the following two issues. First, we need to quantify the relevance of a variable beyond a single input point to define a proper (global) notion of sparsity. If the partial derivative is continuous[2] then a natural idea is to consider

$$\left\|\frac{\partial f}{\partial x^a}\right\|_{\rho_X} = \sqrt{\int_X \left(\frac{\partial f(x)}{\partial x^a}\right)^2 d\rho_X(x)}. \tag{2}$$

where $\rho_X$ is the marginal probability measure of $\rho$ on $X$. While considering other $L^p$ norms is possible, in this paper we restrict our attention to $L^2$. A notion of nonparametric sparsity for a smooth, non-linear function $f$ is captured by the following functional

$$\Omega_0^{\mathrm{D}}(f) = \#\left\{a = 1,\ldots,d \;:\; \left\|\frac{\partial f}{\partial x^a}\right\|_{\rho_X} \neq 0\right\}, \tag{3}$$

1. In order for the partial derivatives to be defined at all points we always assume that the closure of $X$ coincides with the closure of its interior.
2. In the following, see Remark 2, we will see that further appropriate regularity properties on $f$ are needed depending on whether the support of $\rho_X$ is connected or not.

and the corresponding relaxation is

$$\Omega_1^{\mathrm{D}}(f) = \sum_{a=1}^{d} \left\| \frac{\partial f}{\partial x^a} \right\|_{\rho_X}.$$

The above functionals encode the notion of sparsity that we are going to consider. While for linear models, the above definition subsumes the classic notion of sparsity, the above definition is non constrained to any (parametric) additive model.

Second, since $\rho_X$ is only known through the training set, to obtain a practical algorithm we start by replacing the $L^2$ norm with an empirical version

$$\left\| \frac{\partial f}{\partial x^a} \right\|_n = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left( \frac{\partial f(x_i)}{\partial x^a} \right)^2}$$

and by replacing (2) by the data-driven regularizer,

$$\widehat{\Omega}_1^D(f) = \sum_{a=1}^{d} \left\| \frac{\partial f}{\partial x^a} \right\|_n. \tag{4}$$

While the above quantity is a natural estimate of (2) in practice it might not be sufficiently stable to ensure good function estimates where data are poorly sampled. In the same spirit of manifold regularization (Belkin and Niyogi, 2008), we then propose to further consider functions in a reproducing kernel Hilbert space (RKHS) defined by a differentiable kernel and use the penalty,

$$2\widehat{\Omega}_1^D(f) + \nu \|f\|_{\mathcal{H}}^2,$$

where $\nu$ is a small positive number. The latter terms ensures stability while making the regularizer strongly convex. This latter property is a key for well-posedness and generalization, as we discuss in Section 5. As we will see in the following, RKHS will also be a key tool allowing computations of partial derivative of potentially high dimensional functions.

The final learning algorithm is given by the minimization of the functional

$$\frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \tau \left( 2 \sum_{a=1}^{d} \left\| \frac{\partial f}{\partial x^a} \right\|_n + \nu \|f\|_{\mathcal{H}}^2 \right). \tag{5}$$

The remainder of the paper is devoted to the analysis of the above regularization algorithm. Before summarizing our main results we add two remarks.

**Remark 1 (Comparison with Derivative Based Regularizers)** *It is perhaps useful to remark the difference between the regularizer we propose and other derivative based regularizers. We start by considering*

$$\sum_{a=1}^{d} \left\| \frac{\partial f}{\partial x^a} \right\|_n^2 = \frac{1}{n} \sum_{i=1}^{n} \sum_{a=1}^{d} \left( \frac{\partial f(x_i)}{\partial x^a} \right)^2 = \frac{1}{n} \sum_{i=1}^{n} \|\nabla f(x_i)\|^2,$$

*where $\nabla f(x)$ is the gradient of $f$ at $x$. This is essentially a data-dependent version of the classical penalty in Sobolev spaces which writes $\int \|\nabla f(x)\|^2 dx$, where the uniform (Lebesgue) measure is considered. It is well known that while this regularizer measure the smoothness it does not yield any*

Figure 1: Difference between $\ell^1/\ell^1$ and $\ell^1/\ell^2$ norm for binary matrices (white = 1, black=0), where in the latter case the $\ell^1$ norm is taken over the rows (variables) and the $\ell^2$ norm over the columns (samples). The two matrices have the same number of nonzero entries, and thus the same $\ell^1/\ell^1$ norm, but the value of the $\ell^1/\ell^2$ norm is smaller for the matrix on the right, where the nonzero entries are positioned to fill a subset of the rows. The situation on the right is thus favored by $\ell^1/\ell^2$ regularization.

*sparsity property. A different derivative based regularizer is given by* $\frac{1}{n}\sum_{i=1}^{n}\sum_{a=1}^{d}\left|\frac{\partial f(x_i)}{\partial x^a}\right|$. *Though this penalty (which we call* $\ell^1/\ell^1$*) favors sparsity, it only forces partial derivative at points to be zero. In comparison the regularizer we propose is of the* $\ell^1/\ell^2$ *type and uses the square root to "group" the values of each partial derivative at different points hence favoring functions for which each partial derivative is small at most points. The difference between penalties is illustrated in Figure 1. Finally note that we can also consider* $\frac{1}{n}\sum_{i=1}^{n}\|\nabla f(x_i)\|$. *This regularizer, which is akin to the total variation regularizer* $\int\|\nabla f(x)\|dx$*, groups the partial derivatives differently and favors functions with localized singularities rather than selecting variables.*

**Remark 2** *As it is clear from the previous discussion, we quantify the importance of a variable based on the norm of the corresponding partial derivative. This approach makes sense only if*

$$\left\|\frac{\partial f}{\partial x_a}\right\|_{\rho_X}=0 \;\Rightarrow\; f \text{ is constant with respect to } x_a.$$

*The previous fact holds trivially if we assume the function* $f$ *to be continuously differentiable (so that the derivative is pointwise defined, and is a continuous function) and* $\mathrm{supp}\rho_X$ *to be connected. If the latter assumption is not satisfied the situation is more complicated, as the following example shows. Suppose that* $\rho_X$ *is the uniform distribution on the disjoint intervals* $[-2,-1]$ *and* $[1,2]$*, and* $\mathcal{Y}=\{-1,1\}$*. Moreover assume that* $\rho(y|x)=\delta_{-1}$*, if* $x\in[-2,-1]$ *and* $\rho(y|x)=\delta_1$*, if* $x\in[1,2]$*. Then, if we consider the regression function*

$$f(x)=\begin{cases}-1 & \text{if } x\in[-2,-1]\\ 1 & \text{if } x\in[1,2]\end{cases}$$

*we get that* $f'(x)=0$ *on the support of* $\rho_X$*, although the variable* $x$ *is relevant. To avoid such pathological situations when* $\mathrm{supp}\rho_X$ *is not connected in* $\mathbb{R}^d$ *we need to impose more stringent regularity assumptions that imply that a function which is constant on a open interval is constant everywhere. This is verified when* $f$ *belongs to the RKHS defined by a polynomial kernel, or, more generally, an analytic kernel such as the Gaussian kernel.*

### 3.2 Main Results

We summarize our main contributions.

1. Our main contribution is the analysis of the minimization of (5) and the derivation of a provably convergent iterative optimization procedure. We begin by extending the representer theorem (Wahba, 1990) and show that the minimizer of (5) has the finite dimensional representation
$$\hat{f}^{\tau}(x) = \sum_{i=1}^{n} \frac{1}{n} \alpha_i k(x_i, x) + \sum_{i=1}^{n} \sum_{a=1}^{d} \frac{1}{n} \beta_{ai} \left. \frac{\partial k(s,x)}{\partial s^a} \right|_{s=x_i},$$
with $\alpha, (\beta_{ai})_{i=1}^{n} \in \mathbb{R}^n$ for all $a = 1, \ldots, d$. Then, we show that the coefficients in the expansion can be computed using forwards-backward splitting and proximal methods (Combettes and Wajs, 2005; Beck and Teboulle, 2009). More precisely, we present a fast forward-backward splitting algorithm, in which the proximity operator does not admit a closed form and is thus computed in an approximated way. Using recent results for proximal methods with approximate proximity operators, we are able to prove convergence (and convergence rates) for the overall procedure. The resulting algorithm requires only matrix multiplications and thresholding operations and is in terms of the coefficients $\alpha$ and $\beta$ and matrices given by the kernel and its first and second derivatives evaluated at the training set points.

2. We study the consistency properties of the obtained estimator. We prove that, if the kernel we use is universal, then there exists a choice of $\tau = \tau_n$ depending on $n$ such that the algorithm is universally consistent (Steinwart and Christmann, 2008), that is
$$\lim_{n \to \infty} \mathrm{P}\big( \mathcal{E}(\hat{f}^{\tau_n}) - \mathcal{E}(f_\rho) > \varepsilon \big) = 0$$
for all $\varepsilon > 0$. Moreover, we study the selection properties of the algorithm and prove that, if $R_\rho$ is the set of relevant variables and $\hat{R}^{\tau_n}$ the set estimated by our algorithm, then the following consistency result holds
$$\lim_{n \to \infty} \mathrm{P}\big( \hat{R}^{\tau_n} \subseteq R_\rho \big) = 1.$$

3. Finally we provide an extensive empirical analysis both on simulated and benchmark data, showing that the proposed algorithm (DENOVAS) compares favorably and often outperforms other algorithms. This is particularly evident when the function to be estimated is highly non linear. The proposed method can take advantage of working in a rich, possibly infinite dimensional, hypotheses space given by a RKHS, to obtain better estimation and selection properties. This is illustrated in Figure 3.2, where the regression function is a nonlinear function of 2 of 20 possible input variables. With 100 training samples the algorithms we propose is the only one able to correctly solve the problem among different linear and non linear additive models. On real data our method outperforms other methods on several data sets. In most cases, the performance of our method and regularized least squares (RLS) are similar. However our method brings higher interpretability since it is able to select a smaller subset of relevant variable, while the estimator provided by RLS depends on all variables.

## 4. Computational Analysis

In this section we study the minimization of the functional (5).

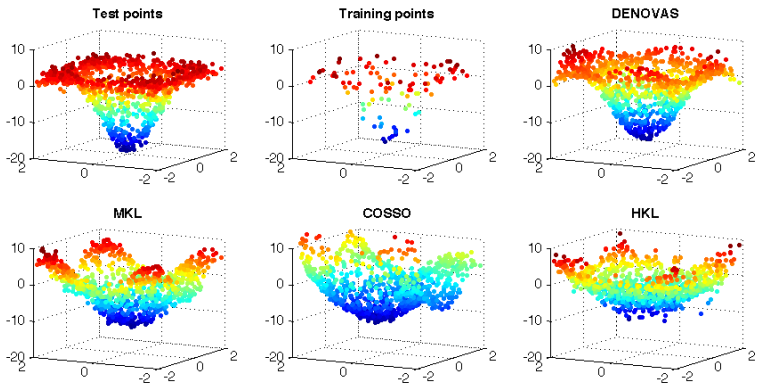



Figure 2: Comparison of predictions for a radial function of 2 out of 20 variables (the 18 irrelevant variables are not shown in the figure). In the upper left plot is depicted the value of the function on the test points (left), the noisy training points (center), the values predicted for the test points by our method (DENOVAS) (right). The bottom plots represent the values predicted for the test points by state-of-the-art algorithms based on additive models. Left: Multiple kernel learning based on additive models using kernels. Center: COSSO, which is a higher order additive model based on ANOVA decomposition (Lin and Zhang, 2006). Right: Hierarchical kernel learning (Bach, 2009).

### 4.1 Basic Assumptions

We first begin by listing some basic conditions that we assume to hold throughout the paper.

We let $\rho$ be a probability measure on $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$. A training set $\mathbf{z}_n = (\mathbf{x}, \mathbf{y}) = (x_i, y_i)_{i=1}^n$ is a sample from $\rho^n$. We consider a reproducing kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (Aronszajn, 1950) and the associated reproducing kernel Hilbert space $\mathcal{H}$. We assume $\rho$ and $k$ to satisfy the following assumptions.

**[A1]** *There exists* $\kappa_1 < \infty$ *such that* $\sup_{x \in X} \|t \mapsto k(x,t)\|_{\mathcal{H}} < \kappa_1$.

**[A2]** *The kernel* $k$ *is* $\mathcal{C}^2(\mathcal{X} \times \mathcal{X})$ *and there exists* $\kappa_2 < \infty$ *such that for all* $a = 1, \ldots, d$ *we have* $\sup_{x \in X} \|t \mapsto \frac{\partial k(s,x)}{\partial s^a}\Big|_{s=t}\|_{\mathcal{H}} < \kappa_2$.

**[A3]** *There exists* $M < \infty$ *such that* $\mathcal{Y} \subseteq [-M, M]$.

### 4.2 Computing the Regularized Solution

We start our analysis discussing how to compute efficiently a regularized solution of the functional

$$\widehat{\mathcal{E}}^{\tau}(f) := \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \tau \left( 2\widehat{\Omega}_1^D(f) + \nu \|f\|_{\mathcal{H}}^2 \right), \tag{6}$$

where $\widehat{\Omega}_1^D(f)$ is defined in (4). The term $\|f\|_{\mathcal{H}}^2$ makes the above functional coercive and strongly convex with modulus[3] $\tau\nu/2$, so that standard results (Ekeland and Temam, 1976) ensures existence and uniqueness of the minimizer $\hat{f}^\tau$, for any $\nu > 0$.

The rest of this section is divided into two parts. First we show how the theory of RKHS (Aronszajn, 1950) allows to compute derivatives of functions on high dimensional spaces and also to derive a new representer theorem leading to finite dimensional minimization problems. Second we discuss how to apply proximal methods (Combettes and Wajs, 2005; Beck and Teboulle, 2009) to derive an iterative optimization procedure for which we can prove convergence. It is possible to see that the solution of Problem (6) can be written as

$$\hat{f}^\tau(x) = \sum_{i=1}^n \frac{1}{n}\alpha_i k_{x_i}(x) + \sum_{i=1}^n \sum_{a=1}^d \frac{1}{n}\beta_{a,i}(\partial_a k)_{x_i}(x), \tag{7}$$

where $\alpha, (\beta_{a,i})_{i=1}^n \in \mathbb{R}^n$ for all $a = 1,\dots,d$ $k_x$ is the function $t \mapsto k(x,t)$, and $(\partial_a k)_x$ denotes partial derivatives of the kernel, see (19). The main outcome of our analysis is that the coefficients $\alpha$ and $\beta$ can be provably computed through an iterative procedure. To describe the algorithm we need some notation. For all $a,b = 1,\dots,d$, we define the $n \times n$ matrices $\mathrm{K}, \mathrm{Z}_a, \mathrm{L}_{a,b}$ as

$$\mathrm{K}_{i,j} = \frac{1}{n}k(x_i,x_j), \tag{8}$$

$$[\mathrm{Z}_a]_{i,j} = \frac{1}{n}\left.\frac{\partial k(s,x_j)}{\partial s^a}\right|_{s=x_i}, \tag{9}$$

and

$$[\mathrm{L}_{a,b}]_{i,j} = \frac{1}{n}\left.\frac{\partial^2 k(x,s)}{\partial x^a \partial s^b}\right|_{x=x_i, s=x_j}$$

for all $i,j = 1,\dots,n$. Clearly the above quantities can be easily computed as soon as we have an explicit expression of the kernel, see Example 1 in Appendix A. We introduce also the $n \times nd$ matrices

$$\mathrm{Z} = (\mathrm{Z}_1,\dots,\mathrm{Z}_d)$$
$$\mathrm{L}_a = (\mathrm{L}_{a,1},\dots,\mathrm{L}_{a,d}) \qquad \forall a = 1,\dots,d \tag{10}$$

and the $nd \times nd$ matrix

$$\mathrm{L} = \begin{pmatrix} \mathrm{L}_{1,1} & \dots & \mathrm{L}_{1,d} \\ \dots & \dots & \dots \\ \mathrm{L}_{d,1} & \dots & \mathrm{L}_{d,d} \end{pmatrix} = \begin{pmatrix} \mathrm{L}_a \\ \dots \\ \mathrm{L}_d \end{pmatrix}.$$

Denote with $B_n$ the unitary ball in $\mathbb{R}^n$,

$$B_n = \{v \in \mathbb{R}^n \mid \|v\|_n \le 1\}. \tag{11}$$

The coefficients in (7) are obtained through Algorithm 1, where $\beta$ is considered as a $nd$ column vector $\beta = (\beta_{1,1},\dots,\beta_{1,n},\dots,\beta_{d,1},\dots,\beta_{d,n})^T$.

3. We say that a function $\mathcal{E} : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ is:

- *coercive* if $\lim_{\|f\|\to+\infty} \mathcal{E}(f)/\|f\| = +\infty$;
- *strongly convex of modulus* $\mu$ if $\mathcal{E}(tf + (1-t)g) \le t\mathcal{E}(f) + (1-t)\mathcal{E}(g) - \frac{\mu}{2}t(1-t)\|f-g\|^2$ for all $t \in [0,1]$.

**Algorithm 1**

**Given:** parameters $\tau, \nu > 0$ and step-sizes $\sigma, \eta > 0$
**Initialize:** $\alpha^0 = \alpha^1 = 0$, $\beta^0 = \beta^1 = 0$, $s_1 = 1$, $\vec{v}^1 = 0$, $t = 1$
**while** `convergence not reached` **do**
$t = t + 1$

$$s_t = \frac{1}{2}\left(1 + \sqrt{1 + 4s_{t-1}^2}\right) \tag{12}$$

$$\tilde{\alpha}^t = \left(1 + \frac{s_{t-1} - 1}{s_t}\right)\alpha^{t-1} + \frac{1 - s_{t-1}}{s_t}\alpha^{t-2}, \quad \tilde{\beta}^t = \left(1 + \frac{s_{t-1} - 1}{s_t}\right)\beta^{t-1} + \frac{1 - s_{t-1}}{s_t}\beta^{t-2}, \tag{13}$$

$$\alpha^t = \left(1 - \frac{\tau\nu}{\sigma}\right)\tilde{\alpha}^t - \frac{1}{\sigma}\left(K\tilde{\alpha}^t + Z\tilde{\beta}^t - \mathbf{y}\right) \tag{14}$$

**set** $v^0 = \vec{v}^{t-1}$, $q = 0$
**while** `convergence not reached` **do**
$q = q + 1$
**for** $a = 1, \ldots d$ **do**

$$v_a^q = \pi_{\frac{\tau}{\sigma}B_n}\left(v_a^{q-1} - \frac{1}{\eta}\left(L_a v^{q-1} - \left(Z_a^T\alpha^t + \left(1 - \frac{\tau\nu}{\sigma}\right)L_a\tilde{\beta}^t\right)\right)\right) \tag{15}$$

**end for**
**end while**
**set** $\vec{v}^t = v^q$

$$\beta^t = \left(1 - \frac{\tau\nu}{\sigma}\right)\tilde{\beta}^t - \vec{v}^t. \tag{16}$$

**end while**
**return** $(\alpha^t, \beta^t)$

The proposed optimization algorithm consists of two nested iterations, and involves only matrix multiplications and thresholding operations. Before describing its derivation and discussing its convergence properties, we add three remarks. First, the proposed procedure requires the choice of appropriate stopping rules for the inner and outer loops, which will be discussed later, and of the step-sizes $\sigma$ and $\eta$. The simple a priori choice $\sigma = \|K\| + \tau\nu$, $\eta = \|L\|$ ensures convergence, as discussed in the Section 4.5, and is the one used in our experiments. Second, the computation of the solution for different regularization parameters can be highly accelerated by a simple warm starting procedure, as the one in Hale et al. (2008). Finally, in Section 4.6 we discuss a principled way to select variable using the norm of the coefficients $(\vec{v}_a^t)_{a=1}^d$.

### 4.3 Kernels, Partial Derivatives and Regularization

We start discussing how partial derivatives can be efficiently computed in RKHSs induced by smooth kernels and hence derive a new representer theorem. Practical computation of the derivatives for a differentiable functions is often performed via finite differences. For functions defined on a high dimensional space such a procedure becomes cumbersome and ultimately not-efficient. RKHSs provide an alternative computational scheme.

Recall that the RKHS associated to a symmetric positive definite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the unique Hilbert space $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$ such that $k_x = k(x, \cdot) \in \mathcal{H}$, for all $x \in \mathcal{X}$ and

$$f(x) = \langle f, k_x \rangle_{\mathcal{H}}, \tag{17}$$

for all $f \in \mathcal{H}, x \in \mathcal{X}$. Property (17) is called *reproducing property* and $k$ is called reproducing kernel (Aronszajn, 1950). We recall a few basic facts. The functions in $\mathcal{H}$ can be written as pointwise limits of finite linear combinations of the type $\sum_{i=1}^{p} \alpha_i k_{x_i}$, where $\alpha_i \in \mathbb{R}, x_i \in \mathcal{X}$ for all $i$. One of the most important results for kernel methods, namely the representer theorem (Wahba, 1990), shows that a large class of regularized kernel methods induce estimators that can be written as *finite* linear combinations of kernels centered at the training set points. In the following we will make use of the so called *sampling operator*, which returns the values of a function $f \in \mathcal{H}$ at a set of input points $\mathbf{x} = (x_1, \ldots, x_n)$

$$\hat{S} : \mathcal{H} \to \mathbb{R}^n, \qquad (\hat{S}f)_i = \langle f, k_{x_i} \rangle, \quad i = 1, \ldots, n. \tag{18}$$

The above operator is linear and bounded if the kernel is bounded—see Appendix A, which is true thanks to Assumption (A1).

Next, we discuss how the theory of RKHS allows efficient derivative computations. Let

$$(\partial_a k)_x := \left. \frac{\partial k(s, \cdot)}{\partial s^a} \right|_{s=x} \tag{19}$$

be the partial derivative of the kernel with respect to the first variable. Then, from Theorem 1 in Zhou (2008) we have that, if $k$ is at least a $\mathcal{C}^2(\mathcal{X} \times \mathcal{X})$, $(\partial_a k)_x$ belongs to $\mathcal{H}$ for all $x \in \mathcal{X}$ and remarkably

$$\frac{\partial f(x)}{\partial x^a} = \langle f, (\partial_a k)_x \rangle_{\mathcal{H}},$$

for $a = 1, \ldots, d$, $x \in X$. It is useful to define the analogous of the sampling operator for derivatives, which returns the values of the partial derivative of a function $f \in \mathcal{H}$ at a set of input points $\mathbf{x} = (x_1, \ldots, x_n)$,

$$\hat{D}_a : \mathcal{H} \to \mathbb{R}^n, \qquad (\hat{D}_a f)_i = \langle f, (\partial_a k)_{x_i} \rangle, \tag{20}$$

where $a = 1, \ldots, d$, $i = 1, \ldots, n$. It is also useful to define an empirical gradient operator $\hat{\nabla} : \mathcal{H} \to (\mathbb{R}^n)^d$ defined by $\hat{\nabla} f = (\hat{D}_a f)_{a=1}^d$. The above operators are linear and bounded, since assumption [A2] is satisfied. We refer to Appendix A for further details and supplementary results.

Provided with the above results we can prove a suitable generalization of the representer theorem.

**Proposition 3** *The minimizer of (6) can be written as*

$$\hat{f}^\tau = \sum_{i=1}^{n} \frac{1}{n} \alpha_i k_{x_i} + \sum_{i=1}^{n} \sum_{a=1}^{d} \frac{1}{n} \beta_{a,i} (\partial_a k)_{x_i}$$

*with* $\alpha \in \mathbb{R}$ *and* $\beta \in \mathbb{R}^{nd}$.

Proposition 3 is proved in Appendix A and shows that the regularized solution is determined by the set of $n + nd$ coefficients $\alpha \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^{nd}$. We next discuss how such coefficients can be efficiently computed.

#### 4.3.1 NOTATION

In the following, given an operator $A$ we denote by $A^*$ the corresponding adjoint operator. When $A$ is a matrix we use the standard notation for the adjoint, that is, the transpose $A^T$.

### 4.4 Computing the Solution with Proximal Methods

The functional $\widehat{\mathcal{E}}^\tau$ is not differentiable, hence its minimization cannot be done by simple gradient methods. Nonetheless it has a special structure that allows efficient computations using a forward-backward splitting algorithm (Combettes and Wajs, 2005), belonging to the class of the so called proximal methods.

Second order methods, see, for example, Chan et al. (1999), could also be used to solve similar problems. These methods typically converge quadratically and allows accurate computations. However, they usually have a high cost per iteration and hence are not suitable for large scale problems, as opposed to first order methods having much lower cost per iteration. Furthermore, in the seminal paper by Nesterov (1983) first-order methods with optimal convergence rate are proposed (Nemirovski and Yudin, 1983). First order methods have since become a popular tool to solve non-smooth problems in machine learning as well as signal and image processing, see for example FISTA —Beck and Teboulle (2009) and references therein. These methods have proved to be fast and accurate (Becker et al., 2011), both for $\ell^1$-based regularization—see Combettes and Wajs (2005), Daubechies et al. (2007), Figueiredo et al. (2007), Loris et al. (2009)—and more general regularized learning methods—see, for example, Duchi and Singer (2009), Mosci et al. (2010) and Jenatton et al. (2010).

#### 4.4.1 FORWARD-BACKWARD SPLITTING ALGORITHMS

The functional $\widehat{\mathcal{E}}^\tau$ is the sum of the two terms $F(\cdot) = \widehat{\mathcal{E}}(\cdot) + \tau\nu\|\cdot\|^2_{\mathcal{H}}$ and $2\tau\widehat{\Omega}^D_1$. The first term is strongly convex of modulus $\tau\nu$ and differentiable, while the second term is convex but not differentiable. The minimization of this class of functionals can be done iteratively using the forward-backward (FB) splitting algorithm,

$$f^t = \mathrm{prox}_{\frac{\tau}{\sigma}\widehat{\Omega}^D_1}\Big(\tilde{f}^t - \frac{1}{2\sigma}\nabla F(\tilde{f}^t)\Big), \tag{21}$$

$$\tilde{f}^t = c_{1,t} f^{t-1} + c_{2,t} f^{t-2} \tag{22}$$

where $t \geq 2$, $\tilde{f}^0 = f^0 = f^1 \in \mathcal{H}$ is an arbitrary initialization, $c_{1,t}, c_{2,t}$ are suitably chosen positive sequences, and $\mathrm{prox}_{\frac{\tau}{\sigma}\widehat{\Omega}^D_1} : \mathcal{H} \to \mathcal{H}$ is the proximity operator (Moreau, 1965) defined by,

$$\mathrm{prox}_{\frac{\tau}{\sigma}\widehat{\Omega}^D_1}(f) = \operatorname*{argmin}_{g\in\mathcal{H}} \left( \frac{\tau}{\sigma}\widehat{\Omega}^D_1(g) + \frac{1}{2}\|f-g\|^2_{\mathcal{H}} \right).$$

The above approach decouples the contribution of the differentiable and not differentiable terms. Unlike other simpler penalties used in additive models, such as the $\ell^1$ norm in the lasso, in our setting the computation of the proximity operator of $\widehat{\Omega}^D_1$ is not trivial and will be discussed in the next paragraph. Here we briefly recall the main properties of the iteration (21), (22) depending on the choice of $c_{1,t}, c_{2,t}$ and $\sigma$. The basic version of the algorithm (Combettes and Wajs, 2005), sometimes called ISTA (iterative shrinkage thresholding algorithm (Beck and Teboulle, 2009)), is obtained setting $c_{1,t} = 1$ and $c_{2,t} = 0$ for all $t > 0$, so that each step depends only on the previous

iterate. The convergence of the algorithm for both the objective function values and the minimizers is extensively studied in Combettes and Wajs (2005), but a convergence rate is not provided. In Beck and Teboulle (2009) it is shown that the convergence of the objective function values is of order $O(1/t)$ provided that the step-size $\sigma$ satisfies $\sigma \geq L$, where $L$ is the Lipschitz constant of $\nabla F/2$. An alternative choice of $c_{1,t}$ and $c_{2,t}$ leads to an accelerated version of the algorithm (21), sometimes called FISTA (fast iterative shrinkage thresholding algorithm (Tseng, 2010; Beck and Teboulle, 2009)), which is obtained by setting $s_0 = 1$,

$$s_t = \frac{1}{2}\left(1+\sqrt{1+4s_{t-1}^2}\right), \qquad c_{1,t} = 1+\frac{s_{t-1}-1}{s_t}, \text{ and } \quad c_{2,t} = \frac{1-s_{t-1}}{s_t}. \tag{23}$$

The algorithm is analyzed in Beck and Teboulle (2009) and in Tseng (2010) where it is proved that the objective values generated by such a procedure have convergence of order $O(1/t^2)$, if the step-size satisfies $\sigma \geq L$.

Computing the Lipschitz constant $L$ can be non trivial. Theorems 3.1 and 4.4 in Beck and Teboulle (2009) show that the iterative procedure (21) with an adaptive choice for the step-size, called *backtracking*, which does not require the computation of $L$, shares the same rate of convergence of the corresponding procedure with fixed step-size. Finally, it is well known that, if the functional is strongly convex with a positive modulus, the convergence rate of both the basic and accelerated scheme is indeed linear for both the function values and the minimizers (Nesterov, 1983; Mosci et al., 2010; Nesterov, 2007).

In our setting we use FISTA to tackle the minimization of $\widehat{\mathcal{E}}^\tau$ but, as we mentioned before, we have to deal with the computation of the proximity operator associated to $\widehat{\Omega}_1^D$.

### 4.4.2 COMPUTING THE PROXIMITY OPERATOR

Since $\widehat{\Omega}_1^D$ is one-homogeneous, that is, $\widehat{\Omega}_1^D(\lambda f) = \lambda \widehat{\Omega}_1^D(f)$ for $\lambda > 0$, the Moreau identity, see Combettes and Wajs (2005), gives a useful alternative formulation for the proximity operator, that is

$$\text{prox}_{\frac{\tau}{\sigma}\widehat{\Omega}_1^D} = I - \pi_{\frac{\tau}{\sigma}\mathcal{C}_n}, \tag{24}$$

where $\mathcal{C}_n = (\partial\widehat{\Omega}_1^D)(0)$ is the subdifferential[4] of $\widehat{\Omega}_1^D$ at the origin, and $\pi_{\frac{\tau}{\sigma}\mathcal{C}_n} : \mathcal{H} \to \mathcal{H}$ is the projection on $\frac{\tau}{\sigma}\mathcal{C}_n$—which is well defined since $\mathcal{C}_n$ is a closed convex subset of $\mathcal{H}$. To describe how to practically compute such a projection, we start observing that the DENOVAS penalty $\widehat{\Omega}_1^D$ is the sum of $d$ norms in $\mathbb{R}^n$. Then following Section 3.2 in Mosci et al. (2010) (see also Ekeland and Temam, 1976) we have

$$\mathcal{C}_n = \partial\widehat{\Omega}_1^D(0) = \left\{ f \in \mathcal{H} \mid f = \hat{\nabla}^* v \text{ with } v \in B_n^d \right\},$$

where $B_n^d$ is the Cartesian product of $d$ unitary balls in $\mathbb{R}^n$,

$$B_n^d = \underbrace{B_n \times \cdots \times B_n}_{d \text{ times}} = \{v = (v_1, \ldots, v_d) \,|v_a \in \mathbb{R}^n,\ \|v_a\|_n \leq 1,\ \ a = 1, \ldots, d\},$$

4. Recall that the subdifferential of a convex functional $\Omega : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ is denoted with $\partial\Omega(f)$ and is defined as the set

$$\partial\Omega(f) := \{h \in \mathcal{H} : \Omega(g) - \Omega(f) \geq \langle h, g-f\rangle_{\mathcal{H}},\ \ \forall g \in \mathcal{H}\}.$$

with $B_n$ defined in (11). Then, by definition, the projection is given by

$$\pi_{\frac{\tau}{\sigma}C_n}(f) = \hat{\nabla}^* \bar{v},$$

where

$$\bar{v} \in \operatorname*{argmin}_{v \in \frac{\tau}{\sigma}B_n^d} \|f - \hat{\nabla}^* v\|_{\mathcal{H}}^2. \tag{25}$$

Being a convex constrained problem, (25) can be seen as the sum of the smooth term $\|f - \hat{\nabla}^* v\|_{\mathcal{H}}^2$ and the indicator function of the convex set $B_n^d$. We can therefore use (21), again. In fact we can fix an arbitrary initialization $v^0 \in \mathbb{R}^{nd}$ and consider,

$$v^{q+1} = \pi_{\frac{\tau}{\sigma}B_n^d}\left(v^q - \frac{1}{\eta}\hat{\nabla}(\hat{\nabla}^* v^q - f)\right), \tag{26}$$

for a suitable choice of $\eta$. In particular, we note that $\pi_{\frac{\tau}{\sigma}B_n^d}$ can be easily computed in closed form, and corresponds to the proximity operator associated to the indicator function of $B_n^d$. Applying the results mentioned above, if $\eta \geq \|\hat{\nabla}\hat{\nabla}^*\|$, convergence of the function values of problem (25) on the sequence generated via (26) is guaranteed. Moreover, thanks to the special structure of the minimization problem in (25), it is possible to prove (see Combettes et al., 2010; Mosci et al., 2010) that

$$\|\hat{\nabla}^* v^q - \hat{\nabla}^* \bar{v}\|_{\mathcal{H}} \to 0, \quad \text{or, equivalently} \quad \|\hat{\nabla}^* v^q - \pi_{\frac{\tau}{\sigma}C_n}(f)\|_{\mathcal{H}} \to 0.$$

A similar first-order method to compute convergent approximations of $\hat{\nabla}^* \bar{v}$ has been proposed in Bect et al. (2004).

### 4.5 Overall Procedure and Convergence Analysis

To compute the minimizer of $\widehat{\mathcal{E}}^\tau$ we consider the combination of the accelerated FB-splitting algorithm (outer iteration) and the basic FB-splitting algorithm for computing the proximity operator (inner iteration). The overall procedure is given by

$$\begin{aligned}
s_t &= \frac{1}{2}\left(1 + \sqrt{1 + 4s_{t-1}^2}\right), \\
\tilde{f}^t &= \left(1 + \frac{s_{t-1} - 1}{s_t}\right) f^{t-1} + \frac{1 - s_{t-1}}{s_t} f^{t-2}, \\
f^t &= \left(1 - \frac{\tau\nu}{\sigma}\right)\tilde{f}^t - \frac{1}{\sigma}\hat{S}^*\left(\hat{S}\tilde{f}^t - \mathbf{y}\right) - \hat{\nabla}^* \bar{v}^t,
\end{aligned}$$

for $t = 2, 3, \ldots$, where $\bar{v}^t$ is computed through the iteration

$$v^q = \pi_{\frac{\tau}{\sigma}B_n^d}\left(v^{q-1} - \frac{1}{\eta}\hat{\nabla}\left(\hat{\nabla}^* v^{q-1} - \left(1 - \frac{\tau\nu}{\sigma}\right)\tilde{f}^t - \frac{1}{\sigma}\hat{S}^*\left(\hat{S}\tilde{f}^t - \mathbf{y}\right)\right)\right), \tag{27}$$

for given initializations.

The above algorithm is an *inexact* accelerated FB-splitting algorithm, in the sense that the proximal or backward step is computed only approximately. The above discussion on the convergence of FB-splitting algorithms was limited to the case where computation of the proximity operator is

done exactly (we refer to this case as the *exact* case). The convergence of the inexact FB-splitting algorithm does not follow from this analysis. For the basic—not accelerated—FB-splitting algorithm, convergence in the inexact case is still guaranteed (without a rate) (Combettes and Wajs, 2005), if the computation of the proximity operator is sufficiently accurate. The convergence of the inexact accelerated FB-splitting algorithm is studied in Villa et al. (see also Salzo and Villa, 2012) where it is shown that the same convergence rate of the exact case can be achieved, again provided that the accuracy in the computation of the proximity operator can be suitably controlled. Such result can be adapted to our setting to prove the following theorem, as shown in Appendix B.

**Theorem 4** *Let* $\varepsilon_t \sim t^{-l}$ *with* $l > 3/2$, $\sigma \geq \|\hat{S}^*\hat{S}\| + \tau\nu$, $\eta \geq \|\hat{\nabla}\hat{\nabla}^*\|$, *and* $f^t$ *given by* (27) *with* $\vec{v}^t$ *computed through algorithm* (27). *There exists* $\bar{q}$ *such that if* $\vec{v}^t = v^q$, *for* $q > \bar{q}$, *the following condition is satisfied*

$$\frac{2\tau}{\sigma}\widehat{\Omega}_1^D(f^t) - 2\langle\hat{\nabla}^*\vec{v}^t, f^t\rangle \leq \varepsilon_t^2 . \tag{28}$$

*Moreover, there exists a constant* $C > 0$ *such that*

$$\widehat{\mathcal{E}}^\tau(f^t) - \widehat{\mathcal{E}}^\tau(\hat{f}^\tau) \leq \frac{C}{t^2},$$

*and thus, if* $\nu > 0$,

$$\|f^t - \hat{f}^\tau\|_{\mathcal{H}} \leq \frac{2}{t}\sqrt{\frac{C}{\nu\tau}}. \tag{29}$$

As for the exact accelerated FB-splitting algorithm, the step-size of the outer iteration has to be greater than or equal to $L = \|\hat{S}^*\hat{S}\| + \tau\nu$. In particular, we choose $\sigma = \|\hat{S}^*\hat{S}\| + \tau\nu$ and, similarly, $\eta = \|\hat{\nabla}\hat{\nabla}^*\|$.

We add few remarks. First, as it is evident from (29), the choice of $\nu > 0$ allows to obtain convergence of $f^t$ to $\hat{f}^\tau$ with respect to the norm in $\mathcal{H}$, and positively influences the rate of convergence. This is a crucial property in variable selection, where it is necessary to accurately estimate the minimizer of the expected risk $f_\rho^\dagger$ and not only its minimum $\mathcal{E}(f_\rho^\dagger)$. Second, condition (28) represents an implementable stopping criterion for the inner iteration, once that the representer theorem is proved and is always asymptotically satisfied by sequences minimizing the dual problem (25) (see Proposition 17). Further comments on the stopping rule are given in Section 4.6. Third, we remark that for proving convergence of the inexact procedure, it is essential that the specific algorithm proposed to compute the proximal step generates a sequence belonging to $\mathcal{C}_n$. More generally, each algorithm generating a feasible sequence $\{v^q\}$ for the dual problem and minimizing the dual function (25) gives admissible approximations of the proximal point (see Villa et al.).

### 4.6 Further Algorithmic Considerations

We conclude discussing several practical aspects of the proposed method.

#### 4.6.1 THE FINITE DIMENSIONAL IMPLEMENTATION

We start by showing how the representer theorem can be used, together with the iterations described by (27) and (27), to derive Algorithm 1. This is summarized in the following proposition.

**Proposition 5** *For $\nu > 0$ and $f^0 = \frac{1}{n}\sum_i \alpha_i^0 k_{x_i} + \frac{1}{n}\sum_i \sum_a \beta_{a,i}^0 (\partial_a k)_{x_i}$ for any $\alpha^0 \in \mathbb{R}^n, \beta^0 \in \mathbb{R}^{nd}$, the solution at step t for the updating rule* (27) *is given by*

$$f^t = \frac{1}{n}\sum_{i=1}^{n} \alpha_i^t k_{x_i} + \frac{1}{n}\sum_{i=1}^{n}\sum_{a=1}^{d} \beta_{a,i}^t (\partial_a k)_{x_i} \tag{30}$$

*with $\alpha^t$ and $\beta^t$ defined by the updating rules (14-16), where $\vec{v}^t$ in* (16) *can be estimated, starting from any $v^0 \in \mathbb{R}^{nd}$, using the iterative rule* (15)*.*

The proof of the above proposition can be found in Appendix B, and is based on the observation that $\mathrm{K}, \mathrm{Z}_a, \mathrm{Z}, \mathrm{L}_a$ defined at the beginning of this Section are the matrices associated to the operators $\hat{S}\hat{S}^* : \mathbb{R}^n \to \mathbb{R}^n$, $\hat{S}\hat{D}_a^* : \mathbb{R}^n \to \mathbb{R}^n$, $\hat{S}\hat{\nabla}^* : \mathbb{R}^{nd} \to \mathbb{R}^n$ and $\hat{D}_a\hat{\nabla}^* : \mathbb{R}^{nd} \to \mathbb{R}^n$, respectively. Using the same reasoning we can make the following two further observations. First, one can compute the step sizes $\sigma$ and $\eta$ as $\sigma = \|\mathrm{K}\| + \tau\nu$, and $\eta = \|\mathrm{L}\|$. Second, since in practice we have to define suitable stopping rules, Equations (29) and (28) suggest the following choices[5]

$$\|f^t - f^{t-1}\|_{\mathcal{H}} \leq \varepsilon^{(\mathrm{ext})} \quad \text{and} \quad \frac{2\tau}{\sigma}\widehat{\Omega}_1^D(f^t) - 2\langle \hat{\nabla}^* v^q, f^t \rangle \leq \varepsilon^{(\mathrm{int})}.$$

As a direct consequence of (30) and using the definition of matrices $\mathrm{K}, \mathrm{Z}, \mathrm{L}$, these quantities can be easily computed as

$$\begin{aligned} \|f^t - f^{t-1}\|_{\mathcal{H}}^2 &= \langle \delta\alpha, \mathrm{K}\delta\alpha \rangle_n + 2\langle \delta\alpha, \mathrm{Z}\delta\beta \rangle_n + \langle \delta\beta, \mathrm{L}\delta\beta \rangle_n, \\ \frac{2\tau}{\sigma}\widehat{\Omega}_1^D(f^t) - 2\langle \hat{\nabla}^* v^q, f^t \rangle &= \sum_{a=1}^{d} \left( \frac{2\tau}{\sigma} \|Z_a\alpha^t + L_a\beta^t\|_n - 2\langle v^q, Z_a^T\alpha^t + L_a\beta^t \rangle_n \right). \end{aligned}$$

where we defined $\delta\alpha = \alpha^t - \alpha^{t-1}$ and $\delta\beta = \beta^t - \beta^{t-1}$. Also note that, according to Theorem 4, $\varepsilon^{(\mathrm{int})}$ must depend on the outer iteration as $\varepsilon^{(\mathrm{int})} = \varepsilon_t^2 \sim t^{-2l}$, $l > 3/2$.

Finally we discuss a criterion for identifying the variables selected by the algorithm.

### 4.6.2 SELECTION

Note that in the linear case $f(x) = \beta \cdot x$ the coefficients $\beta^1, \ldots, \beta^d$ coincide with the partial derivatives, and the coefficient vector $\beta$ given by $\ell^1$ regularization is sparse (in the sense that it has zero entries), so that it is easy to detect which variables are to be considered relevant. For a general non-linear function, we then expect the vector $(\|\hat{D}_a f\|_n^2)_{a=1}^d$ of the norms of the partial derivatives evaluated on the training set points, to be sparse as well. In practice since the projection $\pi_{\tau/\sigma B_n^d}$ is computed only approximately, the norms of the partial derivatives will be small but typically not zero. The following proposition elaborates on this point.

**Proposition 6** *Let $v = (v_a)_{a=1}^d \in B_n^d$ such that, for any $\sigma > 0$*

$$\hat{\nabla}^* v = -\frac{1}{\sigma}\nabla(\widehat{\mathcal{E}}(\hat{f}^\tau) + \tau\nu\|\hat{f}^\tau\|_{\mathcal{H}}^2),$$

*then*

$$\|v_a\|_n < \frac{\tau}{\sigma} \Rightarrow \|\hat{D}_a\hat{f}^\tau\|_n = 0. \tag{31}$$

5. In practice we often use a stopping rule where the tolerance is scaled with the current iterate, $\|f^t - f^{t-1}\|_{\mathcal{H}} \leq \varepsilon^{(\mathrm{ext})}\|f^t\|_{\mathcal{H}}$ and $\frac{2\tau}{\sigma}\widehat{\Omega}_1^D(f^t) - 2\langle \hat{\nabla}^* v^q, f^t \rangle \leq \varepsilon^{(\mathrm{int})}\|\hat{\nabla}^* v^q\|_{\mathcal{H}}$.

*Moreover, if $\vec{v}^t$ is given by Algorithm 1 with the inner iteration stopped when the assumptions of Theorem 4 are met, then there exists $\tilde{\varepsilon}_t > 0$ (precisely defined in* (37)*) depending on the tolerance $\varepsilon^t$ used in the inner iteration and satisfying $\lim_{t\to+\infty} \tilde{\varepsilon}_t = 0$, such that if $m := \min\{\|\hat{D}_a \hat{f}^\tau\|_n : a \in \{1,\dots,d\} \text{s.t.} \|\hat{D}_a \hat{f}^\tau\|_n > 0\}$, then*

$$\|\vec{v}_a^t\|_n \geq \frac{\tau}{\sigma} - \frac{(\tilde{\varepsilon}_t)^2}{2m} \Rightarrow \|\hat{D}_a \hat{f}^\tau\|_n = 0. \tag{32}$$

The above result, whose proof can be found in Appendix B, is a direct consequence of the Euler equation for $\widehat{\mathcal{E}}^\tau$ and of the characterization of the subdifferential of $\widehat{\Omega}_1^D$. The second part of the statement follows by observing that, as $\hat{\nabla}^* v$ belongs to the subdifferential of $\widehat{\Omega}_1^D$ at $\hat{f}^\tau$, $\hat{\nabla}^* \vec{v}^t$ belongs to the *approximate* subdifferential of $\widehat{\Omega}_1^D$ at $\hat{f}^\tau$, where the approximation of the subdifferential is controlled by the precision used in evaluating the projection. Given the pair $(f^t, \vec{v}^t)$ evaluated via Algorithm 1, we can thus consider to be irrelevant the variables such that $\|\vec{v}_a^t\|_n < \tau/\sigma - (\tilde{\varepsilon}^t)^2/(2m)$. Note that the explicit form of $\tilde{\varepsilon}^t$ is given in (37)).

## 5. Consistency for Learning and Variable Selection

In this section we study the consistency properties of our method.

### 5.1 Consistency

As we discussed in Section 3.1, though in practice we consider the regularizer $\widehat{\Omega}_1^D$ defined in (4), ideally we would be interested into $\Omega_1^{\mathrm{D}}(f) = \sum_{a=1}^d \|D_a f\|_{\rho_X}$, $f \in \mathcal{H}$. The following preliminary result shows that indeed $\widehat{\Omega}_1^D$ is a consistent estimator of $\Omega_1^{\mathrm{D}}$ when considering functions in $\mathcal{H}$ having uniformly bounded norm.

**Theorem 7** *Let $r < \infty$, then under assumption (A2)*

$$\lim_{n\to\infty} \mathrm{P}\left( \sup_{\|f\|_{\mathcal{H}} \leq r} |\widehat{\Omega}_1^D(f) - \Omega_1^D(f)| > \varepsilon \right) = 0 \qquad \forall \varepsilon > 0.$$

The restriction to functions such that $\|f\|_{\mathcal{H}} \leq r$ is natural and is required since the penalty $\widehat{\Omega}_1^D$ forces the partial derivatives to be zero only on the training set points. To guarantee that a partial derivative, which is zero on the training set, is also close to zero on the rest of the input space, we must control the smoothness of the function class where the derivatives are computed. This motivates constraining the function class by adding the (squared) norm in $\mathcal{H}$ into (5). This is in the same spirit of the manifold regularization proposed in Belkin and Niyogi (2008).

The above result on the consistency of the derivative based regularizer is at the basis of the following consistency result.

**Theorem 8** *Under assumptions (A1), (A2) and (A3), recalling that $\mathcal{E}(f) = \int (y - f(x))^2 \, d\rho(x,y)$,*

$$\lim_{n\to\infty} \mathrm{P}\left( \mathcal{E}(\hat{f}^{\tau_n}) - \inf_{f\in\mathcal{H}} \mathcal{E}(f) \geq \varepsilon \right) = 0 \qquad \forall \varepsilon > 0,$$

*for any $\tau_n$ satisfying*

$$\tau_n \to 0 \qquad (\sqrt{n}\tau_n)^{-1} \to 0.$$

The proof is given in the appendix and is based on a sample/approximation error decomposition

$$\mathcal{E}(\hat{f}^\tau) - \inf_{f\in\mathcal{H}} \mathcal{E}(f) \leq \underbrace{|\mathcal{E}(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau)|}_{\text{sample error}} + \underbrace{|\mathcal{E}^\tau(f^\tau) - \inf_{f\in\mathcal{H}} \mathcal{E}(f)|}_{\text{approximation error}},$$

where

$$\mathcal{E}^\tau(f) := \mathcal{E}(f) + \tau(2\Omega_1^{\mathrm{D}}(f) + \nu\|f\|_{\mathcal{H}}^2), \qquad f^\tau := \operatorname*{argmin}_{\mathcal{H}} \mathcal{E}^\tau.$$

The control of both terms allows to find a suitable parameter choice which gives consistency. When estimating the sample error one has typically to control only the deviation of the empirical risk from its continuous counterpart. Here we need Theorem 7 to also control the deviation of $\widehat{\Omega}_1^D$ from $\Omega_1^{\mathrm{D}}$. Note that, if the kernel is universal (Steinwart and Christmann, 2008), then $\inf_{f\in\mathcal{H}} \mathcal{E}(f) = \mathcal{E}(f_\rho)$ and Theorem 8 gives the universal consistency of the estimator $\hat{f}^{\tau_n}$.

To study the selection properties of the estimator $\hat{f}^{\tau_n}$—see next section—it useful to study the distance of $\hat{f}^{\tau_n}$ to $f_\rho$ in the $\mathcal{H}$-norm. Since in general $f_\rho$ might not belong to $\mathcal{H}$, for the sake of generality here we compare $\hat{f}^{\tau_n}$ to a minimizer of $\inf_{f\in\mathcal{H}} \mathcal{E}(f)$ which we always assume to exist. Since the minimizers might be more then one we further consider a suitable minimal norm minimizer $f_\rho^\dagger$—see below. More precisely given the set

$$\mathcal{F}_{\mathcal{H}} := \{f \in \mathcal{H} \mid \mathcal{E}(f) = \inf_{f\in\mathcal{H}} \mathcal{E}(f)\}$$

(which we assume to be not empty), we define

$$f_\rho^\dagger := \operatorname*{argmin}_{f\in\mathcal{F}_{\mathcal{H}}} \{\Omega_1^{\mathrm{D}}(f) + \nu\|f\|_{\mathcal{H}}^2\}.$$

Note that $f_\rho^\dagger$ is well defined and unique, since $\Omega_1^{\mathrm{D}}(\cdot) + \nu\|\cdot\|_{\mathcal{H}}^2$ is lower semicontinuous, strongly convex and $\mathcal{E}$ is convex and lower semi-continuous on $\mathcal{H}$, which implies that $\mathcal{F}_{\mathcal{H}}$ is closed and convex in $\mathcal{H}$. Then, we have the following result.

**Theorem 9** *Under assumptions (A1), (A2) and (A3), we have*

$$\lim_{n\to\infty} \mathrm{P}\Big( \|\hat{f}^{\tau_n} - f_\rho^\dagger\|_{\mathcal{H}} \geq \varepsilon \Big) = 0, \qquad \forall \varepsilon > 0,$$

*for any* $\tau_n$ *such that* $\tau_n \to 0$ *and* $(\sqrt{n}\tau_n^2)^{-1} \to 0$.

The proof, given in Appendix C, is based on the decomposition in sample error, $\|\hat{f}^\tau - f^\tau\|_{\mathcal{H}}$, and approximation error, $\|f^\tau - f_\rho^\dagger\|_{\mathcal{H}}$. To bound the sample error we use recent results (Villa et al., 2012) that exploit Attouch-Wets convergence (Attouch and Wets, 1991, 1993a,b) and coercivity of the penalty (ensured by the RKHS norm) to control the distance between the minimizers $\hat{f}^\tau, f^\tau$ by the distance the minima $\widehat{\mathcal{E}}^\tau(\hat{f}^\tau)$ and $\mathcal{E}^\tau(f^\tau)$. Convergence of the approximation error is again guaranteed by standard results in regularization theory (Dontchev and Zolezzi, 1993). We underline that our result is an asymptotic one, although it would be interesting to get an explicit learning rate, as we discuss in Section 5.3.

### 5.2 Selection Properties

We next consider the selection properties of our method. Following Equation (3), we start by giving the definition of relevant/irrelevant variables and sparsity in our context.

**Definition 10** *We say that a variable $a = 1,\dots,d$ is* irrelevant *with respect to $\rho$ for a differentiable function $f$, if the corresponding partial derivative $D_a f$ is zero $\rho_X$-almost everywhere, and relevant otherwise. In other words the set of relevant variables is*

$$R_f := \{a \in \{1,\dots,d\} \mid \|D_a f\|_{\rho_X} > 0\}.$$

*We say that a differentiable function $f$ is sparse if $\Omega_0^D(f) := |R_f| < d$.*

The goal of variable selection is to correctly estimate the set of relevant variables, $R_\rho := R_{f_\rho^\dagger}$. In the following we study how this can be achieved by the empirical set of relevant variables, $\hat{R}^{\tau_n}$, defined as

$$\hat{R}^{\tau_n} := \{a \in \{1,\dots,d\} | \|\hat{D}_a \hat{f}^{\tau_n}\|_n > 0\}.$$

**Theorem 11** *Under assumptions (A1), (A2) and (A3)*

$$\lim_{n\to\infty} \mathrm{P}\big(R_\rho \subseteq \hat{R}^{\tau_n}\big) = 1$$

*for any $\tau_n$ satisfying*

$$\tau_n \to 0 \qquad (\sqrt{n}\tau_n^2)^{-1} \to 0.$$

The above result shows that the proposed regularization scheme is a safe filter for variable selection, since it does not discard relevant variables, in fact, for a sufficiently large number of training samples, the set of truly relevant variables, $R_\rho$, is contained with high probability in the set of relevant variables identified by the algorithm, $\hat{R}^{\tau_n}$. The proof of the converse inclusion, giving consistency for variable selection (sometimes called sparsistency), requires further analysis that we postpone to a future work (see the discussion in the next subsection).

### 5.3 Learning Rates and Sparsity

The analysis in the previous sections is asymptotic, so it is natural to ask about the finite sample behavior of the proposed method, and in particular about the implication of the sparsity assumption. Indeed, for a variety of additive models it is possible to prove that the sample complexity (the number of samples needed to achieve a given error with a specified probability) depends linearly on the sparsity level and in a much milder way to the total number of variables, for example, logarithmically (Bühlmann and van de Geer, 2011). Proving similar results in our setting is considerably more complex and in this section we discuss the main sources of possible challenges.

Towards this end, it is interesting to contrast the form of our regularizer to that of structured sparsity penalties for which sparsity results can be derived. Inspecting the proof in Appendix C, one can see that it possible to define a suitable family of operators $V_j, \hat{V}_j : \mathcal{H} \to \mathcal{H}$, with $j = 1,\dots,d$, such that

$$\Omega_1^{\mathrm{D}}(f) = \sum_{j=1}^d \|V_j f\|_{\mathcal{H}}, \qquad \widehat{\Omega}_1^D(f) = \sum_{j=1}^d \|\hat{V}_j f\|_{\mathcal{H}}. \tag{33}$$

The operators $(V_j)_j$ are positive and self-adjoint and so are the operators $(\hat{V}_j)_j$. The latter can be shown to be stochastic approximation of the operators $(V_j)_j$, in the sense that the equalities $\mathbb{E}[\hat{V}_j^2] = V_j^2$ hold true for all $j = 1, \dots, d$.

It is interesting to compare the above expression to the one for the group lasso penalty, where for a given linear model, the coefficients are assumed to be divided in groups, only few of which are predictive. More precisely, given a collection of groups of indices $\mathcal{G} = \{G_1, \dots, G_r\}$, which forms a partition of the set $\{1, \dots, p\}$, and a linear model $f(x) = \langle \beta, x \rangle_{\mathbb{R}^p}$, the group lasso penalty is obtained by considering

$$\Omega^{GL}(\beta) = \sum_{\gamma=1}^{r} \|\beta_{|G_\gamma}\|_{\mathbb{R}^{|G_\gamma|}},$$

where, for each $\gamma$, $\beta_{|G_\gamma}$ is the $|G_\gamma|$ dimensional vector obtained restricting a vector $\beta$ to the indices in $G_\gamma$. If we let $P_\gamma$ be the orthogonal projection on the subspace of $\mathbb{R}^d$ corresponding $G_\gamma$-th group of indices, we have that $\langle P_\gamma \beta, P'_\gamma \beta' \rangle = 0$ for all $\gamma, \gamma' \in \Gamma$ and $\beta, \beta' \in \mathbb{R}^p$, since the groups form a partition of $\{1, \dots, p\}$. Then it is possible to rewrite the group lasso penalty as

$$\Omega^{GL}(\beta) = \sum_{\gamma=1}^{r} \|P_\gamma \beta\|_{\mathbb{R}^{|G_\gamma|}}.$$

The above idea can be extended to an infinite dimensional setting to obtain multiple kernel learning (MKL). Let $\mathcal{H}$ be a (reproducing kernel) Hilbert space which is the sum of $\Gamma$ disjoint (reproducing kernel) Hilbert spaces $(\mathcal{H}_\gamma, \|\cdot\|_\gamma)_{\gamma \in \Gamma}$, and $P_\gamma : \mathcal{H} \to \mathcal{H}_\gamma$ the projections of $\mathcal{H}$ onto $\mathcal{H}_\gamma$, then MKL is induced by the penalty

$$\Omega^{MKL}(f) = \sum_{\gamma \in \Gamma} \|P_\gamma f\|_\gamma.$$

When compared to our derivative based penalty, see (33), one can notice at least two source of difficulties:

1. the operators $(V_j)_j$ are not projections and no simple relation exists among their ranges,
2. in practice we have only access to the empirical estimates $(\hat{V}_j)_j$.

Indeed, structured sparsity model induced by more complex index sets have been considered, see, for example, Jenatton et al. (2010), but the penalties are still induced by operators which are orthogonal projections. Interestingly, a class of penalties induced by a (possibly countable) family of bounded operators $\mathcal{V} = \{V_\gamma\}_{\gamma \in \Gamma}$—not necessarily projections—has been considered in Maurer and Pontil (2012). This class of penalties can be written as

$$\Omega^{(\mathcal{V})}(f) = \inf\{\sum_{\gamma \in \Gamma} \|f_\gamma\| \mid f_\gamma \in \mathcal{H}, \sum_{\gamma \in \Gamma} V_\gamma f_\gamma = f\}.$$

It is easy to see that the above penalty does not include the regularizer (33) as a special case.

In conclusion, rewriting our derivative based regularizer as in (33) highlights similarity and differences with respect to previously studied sparsity methods: indeed many of these methods are induced by families of operators. On the other hand, typically, the operators are assumed to satisfy stringent assumptions which do not hold true in our case. Moreover in our case one would have to overcome the difficulties arising from the random estimation of the operators. These interesting questions are outside of the scope of this paper, will be the subject of future work.

## 6. Empirical Analysis

The content of this section is divided into three parts. First, we describe the choice of tuning parameters. Second, we study the properties of the proposed method on simulated data under different parameter settings, and third, we compare our method to related regularization methods for learning and variable selection.

When we refer to our method we always consider a two-step procedure based on variable selection via Algorithm 1 and regression on the selected variable via (kernel) Regularized Least Squares (RLS). The kernel used in both steps is the same. Where possible, we applied the same reweighting procedure to the methods we compared with.

### 6.1 Choice of Tuning Parameters

When using Algorithm 1, once the parameters $n$ and $\nu$ are fixed, we evaluate the optimal value of the regularization parameter $\tau$ via hold out validation on an independent validation set of $nval = n$ samples. The choice of the parameter $\nu$, and its influence on the estimator is discussed in the next section.

Since we use an iterative procedure to compute the solution, the output of our algorithm will not be sparse in general and a selection criterion is needed. In Section 4.6 we discussed a principled way to select variable using the norm of the coefficients $(\vec{v}_a^t)_{a=1}^d$.

When using MKL, $\ell^1$ regularization, and RLS we used hold out validation to set the regularization parameters, while for COSSO and HKL we used the choices suggested by the authors.

### 6.2 Analysis of Our Method

In this subsection we empirically investigate the sensitivity of the proposed algorithm to the model's parameters.

#### 6.2.1 ROLE OF THE SMOOTHNESS ENFORCING PENALTY $\nu\|\cdot\|_{\mathcal{H}}^2$

From a theoretical stand point we have shown that $\nu$ has to be nonzero, in order for the proposed regularization problem (5) to be well-posed. We also mentioned that the combination of the two penalties $\widehat{\Omega}_1^D$ and $\|\cdot\|_{\mathcal{H}}^2$ ensures that the regularized solution will not depend on variables that are irrelevant for two different reasons. The first is irrelevance with respect to the output. The second type of irrelevance is meant in an unsupervised sense. This happens when one or more variables are (approximately) constant with respect to the marginal distribution $\rho_X$, so that the support of the marginal distribution is (approximately) contained in a coordinate subspace. Here we present two experiments aimed at empirically assessing the role of the smoothness enforcing penalty $\|\cdot\|_{\mathcal{H}}^2$ and of the parameter $\nu$. We first present an experiment where the support of the marginal distribution approximately coincides with a coordinate subspace $x^2 = 0$. Then we systematically investigate the stabilizing effect of the smoothness enforcing penalty also when the marginal distribution is not degenerate.

*Adaption To The Marginal Distribution* We consider a toy problem in 2 dimensions, where the support of the marginal distribution $\rho_X(x^1, x^2)$ approximately coincides with the coordinate subspace $x^2 = 0$. Precisely $x^1$ is uniformly sampled from $[-1,1]$, whereas $x^2$ is drawn from a normal distribution $x^2 \sim \mathcal{N}(0, 0.05)$. The output labels are drawn from $y = (x^1)^2 + w$, where $w$ is a white noise, sampled from a normal distribution with zero mean and variance 0.1. Given a training

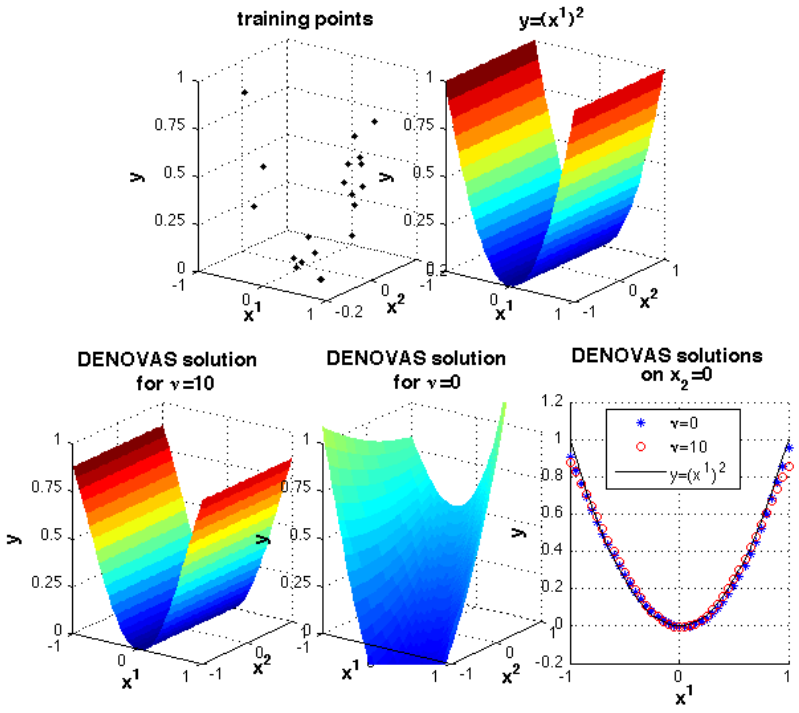



Figure 3: Effect of the combined regularization $\widehat{\Omega}_1^D(\cdot)+\nu\|\cdot\|_{\mathcal{H}}^2$ on a toy problem in $\mathbb{R}^2$ where the support of marginal distribution approximately coincides with the coordinate subspace $x^2=0$. The output labels are drawn from $y=(x^1)^2+w$, with $w\sim\mathcal{N}(0,0.1)$.

set of $n=20$ samples i.i.d. drawn from the above distribution (Figure 3 top-left), we evaluate the optimal value of the regularization parameter $\tau$ via hold out validation on an independent validation set of $n_{\text{val}}=n=20$ samples. We repeat the process for $\nu=0$ and $\nu=10$. In both cases the reconstruction accuracy on the support of $\rho_X$ is high, see Figure 3 bottom-right . However, while $\nu=10$ our method correctly selects the only relevant variable $x^1$ (Figure 3 bottom-left), when $\nu=0$ both variables are selected (Figure 3 bottom-center), since functional $\widehat{\mathcal{E}}^{\tau,0}$ is insensible to errors out of $\text{supp}(\rho_X)$, and the regularization term $\tau\widehat{\Omega}_1^D$ alone does not penalizes variations out of $\text{supp}(\rho_X)$ .

*Effect of varying* $\nu$. Here we empirically investigate the stabilizing effect of the smoothness enforcing penalty when the marginal distribution is not degenerate. The input variables $x=(x^1,\ldots,x^{20})$ are uniformly drawn from $[-1,1]^{20}$. The output labels are i.i.d. drawn from $y=\lambda\sum_{a=1}^4\sum_{b=a+1}^4 x^a x^b+w$, where $w\sim\mathcal{N}(0,1)$, and $\lambda$ is a rescaling factor that determines the signal to noise ratio to be 15:1. We extract training sets of size $n$ which varies from 50 to 120 with steps of 10. We then apply our method with polynomial kernel of degree $p=4$, letting vary $\nu$ in $\{0.5,1,5,20,100\}$. For fixed $n$ and $\nu$ we evaluate the optimal value of the regularization parameter $\tau$ via hold out validation on an independent validation set of $n_{\text{val}}=n$ samples. We measure the selection error as the mean of the false negative rate (fraction of relevant variables that were not selected) and false positive rate (fraction of irrelevant variables that were selected). Then, we

evaluate the prediction error as the root mean square error (RMSE) error of the selected model on an independent test set of $n_{\text{test}} = 500$ samples. Finally we average over 20 repetitions.

In Figure 6.2.1 we display the prediction error, selection error, and computing time, versus $n$ for different values of $\nu$. Clearly, if $\nu$ is too small, both prediction and selection are poor. For $\nu \geq 1$ the algorithm is quite stable with respect to small variations of $\nu$. However, excessive increase of the smoothness parameter leads to a slight decrease in prediction and selection performance. In terms of computing time, the higher the smoothness parameter the better the performance.

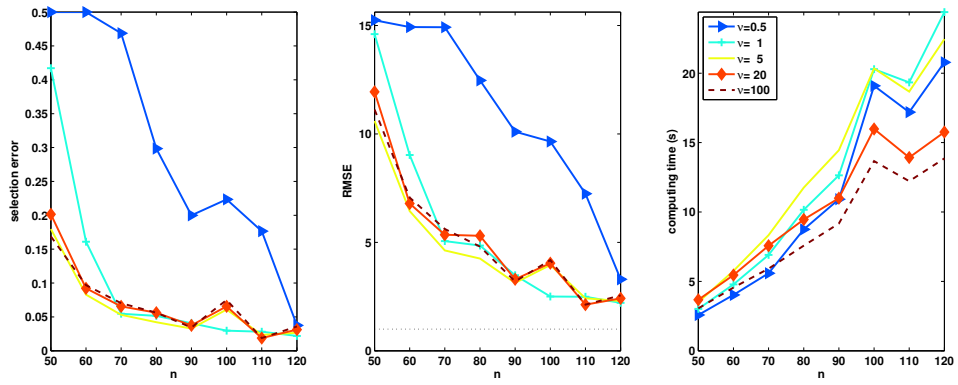



Figure 4: Selection error (left), prediction error (center), and computing time (right) versus $n$ for different values of $\nu$. The points correspond to the mean over the repetitions. The dotted line represents the white noise standard deviation. In the left figure the curves corresponding to $\nu = 5$, $\nu = 10$, and $\nu = 20$ are overlapping.

### 6.2.2 VARYING THE MODEL'S PARAMETERS

We present 3 sets of experiments where we evaluated the performance of our method (DENOVAS) when varying part of the inputs parameters and leaving the others unchanged. The parameters we take into account are the following

- $n$, training set size
- $d$, input space dimensionality
- $|R_\rho|$, number of relevant variables
- $p$, size of the hypotheses space, measured as the degree of the polynomial kernel.

In all the following experiments the input variables $x = (x^1, \ldots, x^d)$ are uniformly drawn from $[-1,1]^d$. The output labels are computed using a noise-corrupted regression function $f$ that depends nonlinearly from a set of the input variables, that is, $y = \lambda f(x) + w$, where $w$ is a white noise, sampled from a normal distribution with zero mean and variance 1, and $\lambda$ is a rescaling factor that determines the signal to noise ratio. For fixed $n, d$, and $|R_\rho|$ we evaluate the optimal value of the regularization parameter $\tau$ via hold out validation on an independent validation set of $n_{\text{val}} = n$ samples.

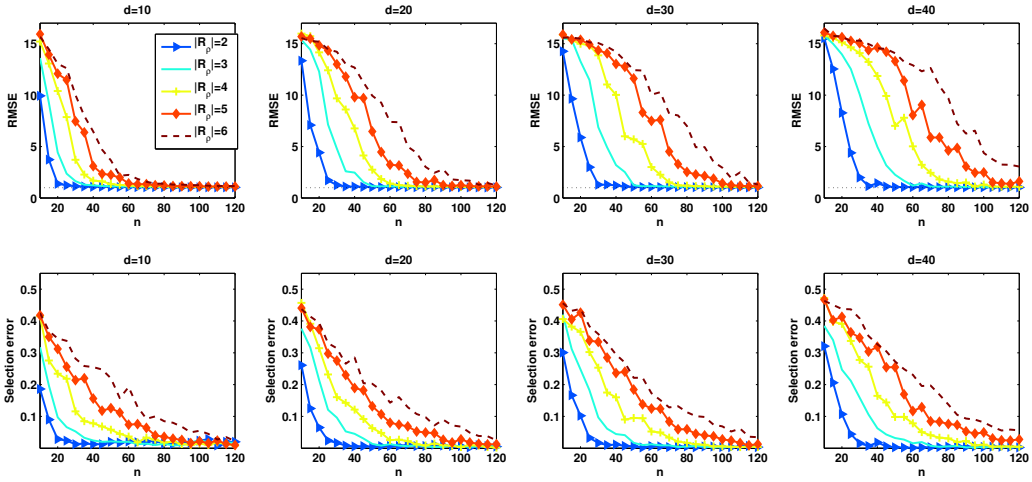



Figure 5: Prediction error (top) and selection error (bottom) versus $n$ for different values of $d$ and number of relevant variables ($|R_\rho|$). The points correspond to the means over the repetitions. The dotted line represents the white noise standard deviation.

*Varying $n, d$, and $|R_\rho|$* In this experiment we want to empirically evaluate the effect of the input space dimensionality, $d$, and the number of relevant variables, $|R_\rho|$, when the other parameters are left unchanged. In particular we use $d = 10, 20, 30, 40$ and $|R_\rho| = 2, 3, 4, 5, 6$. For each value of $|R_\rho|$ we use a different regression function, $f(x) = \lambda \sum_{a=1}^{|R_\rho|} \sum_{b=a+1}^{|R_\rho|} c_{ab} x^a x^b$, so that for fixed $|R_\rho|$ all 2-way interaction terms are present, and the polynomial degree of the regression function is always 2. The coefficients $c_{ab}$ are randomly drawn from $[.5, 1]$ And $\lambda$ is determined in order to set the signal to noise ratio as 15:1. We then apply our method with polynomial kernel of degree $p = 2$. The regression function thus always belongs to the hypotheses space.

In Figure 6.2.2, we display the selection error, and the prediction error, respectively, versus $n$ for different values of $d$ and number of relevant variables $|R_\rho|$. Both errors decrease with $n$ and increase with $d$ and $|R_\rho|$. In order to better visualize the dependence of the selection performance with respect to $d$ and $|R_\rho|$, in Figure 6.2.2 we plotted the minimum number of input points that are necessary in order to achieve 10% of average selection error. It is clear by visual inspection that $|R_\rho|$ has a higher influence than $d$ on the selection performance of our method.

*Varying $n$ and $p$* In this experiment we want to empirically evaluate the effect of the size of the hypotheses space on the performance of our method. We therefore leave unchanged the data generation setting, made exception for the number of training samples, and vary the polynomial kernel degree as $p = 1, 2, 3, 4, 5, 6$. We let $d = 20$, $R_\rho = \{1, 2\}$, and $f(x) = x^1 x^2$, and let vary $n$ from 20 to 80 with steps of 5. The signal to noise ratio is 3:1.

In Figure 6.2.2, we display the prediction and selection error, versus $n$, for different values of $p$. For $p \geq 2$, that is when the hypotheses space contains the regression function, both errors decrease with $n$ and increase with $p$. Nevertheless the effect of $p$ decreases for large $p$, in fact for $p = 4, 5$, and 6, the performance is almost the same. On the other hand, when the hypotheses space is too

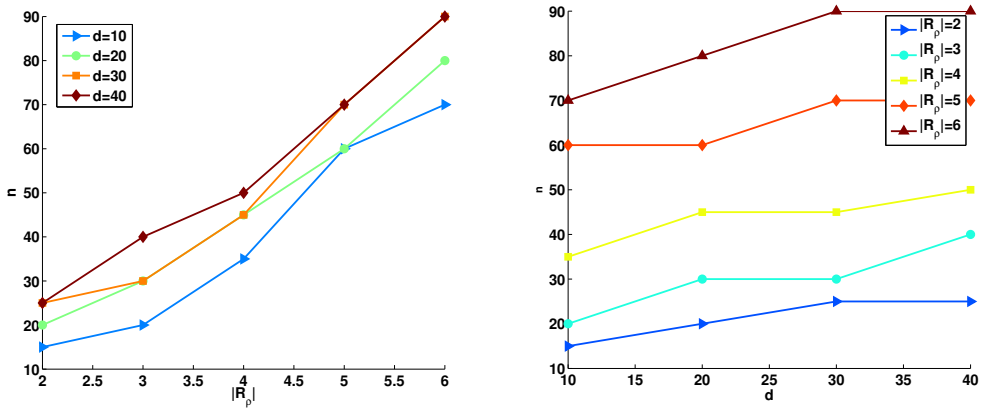



Figure 6: Minimum number of input points ($n$) necessary to achieve 10% of average selection error versus the number of relevant variables $|R_\rho|$ for different values of $d$ (left), and versus $d$ for different values of $|R_\rho|$ (right).

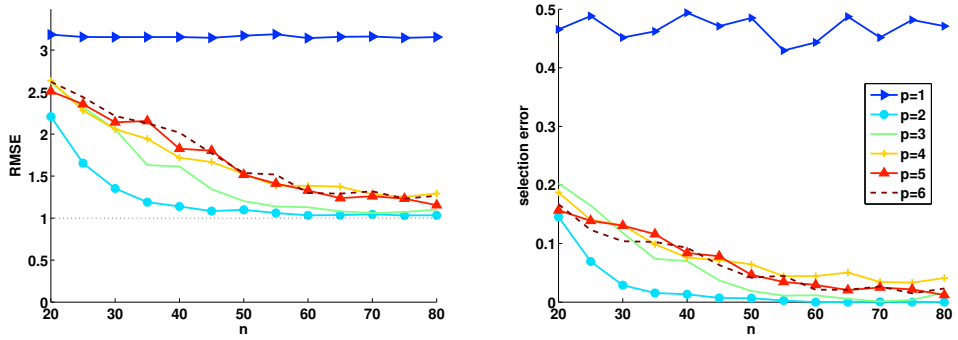



Figure 7: Prediction error (left) and selection error (right) versus $n$ for different values of the polynomial kernel degree ($p$). The points correspond to the means over the repetitions. The dotted line represents the white noise standard deviation.

small to include the regression function, as for the set of linear functions ($p = 1$), the selection error coincides with chance (0.5), and the prediction error is very high, even for large numbers of samples.

*Varying the number of relevant features, for fixed $|R_\rho|$: comparison with $\ell^1$ regularization on the feature space* In this experiment we want to empirically evaluate the effect of the number of features involved in the regression function ( that is the number of monomials constituting the polynomial) on the performance of our method when $|R_\rho|$ remains the same as well as all other input parameters. Note that while $|R_\rho|$ is the number of relevant variables, here we vary the number of relevant features (not variables!), which, in theory, has nothing to do with $|R_\rho|$. Furthermore we compare

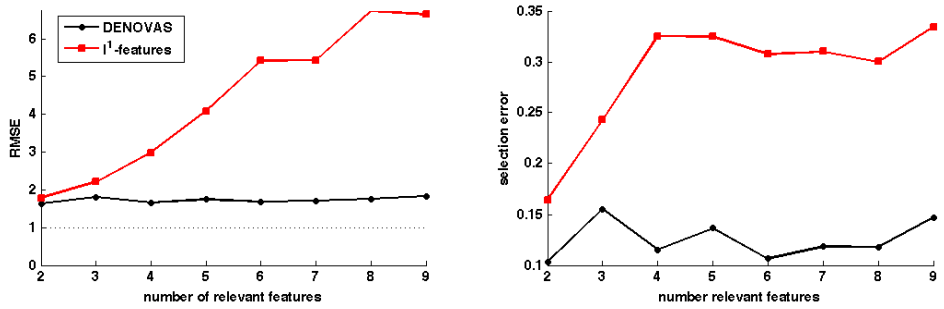



Figure 8: Prediction error (left) selection error (right) versus the number of relevant features. The points correspond to the means over the repetitions. The dotted line represents the white noise standard deviation.

the performance of our method to that of $\ell^1$ regularization on the feature space ($\ell^1$-features). We therefore leave unchanged the data generation setting, made exception for the regression function. We set $d = 10$, $R_\rho = \{1,2,3\}$, $n = 30$, and then use a polynomial kernel of degree 2. The signal to noise ratio is this time 3:1. Note that with this setting the size of the features space is 66. For fixed number of relevant features the regression function is set to be a randomly chosen linear combination of the features involving one or two of the first three variables ($x^1, (x^1)^2, x^1x^2, x^1x^3$, etc.), with the constraint that the combination must be a polynomial of degree 2, involving all 3 variables.

In Figure 6.2.2, we display the prediction and selection error, versus the number of relevant features. While the performance of $\ell^1$-features fades when the number of relevant features increases, our method presents stable performance both in terms of selection and prediction error. From our simulation it appears that, while our method depends on the number of relevant variables, it is indeed independent of the number of features.

### 6.3 Comparison with Other Methods

In this section we present numerical experiments aimed at comparing our method with state-of-the-art algorithms. In particular, since our method is a regularization method, we focus on alternative regularization approaches to the problem of nonlinear variable selection. For comparisons with more general techniques for nonlinear variable selection we refer the interested reader to Bach (2009).

#### 6.3.1 COMPARED ALGORITHMS

We consider the following regularization algorithms:

- Additive models with multiple kernels, that is Multiple Kernel Learning (MKL)
- $\ell^1$ regularization on the feature space associated to a polynomial kernel ($\ell^1$-features)

- COSSO (Lin and Zhang, 2006) with 1-way interaction (COSSO1) and 2-way interactions (COSSO2)[6]
- Hierarchical Kernel Learning (Bach, 2009) with polynomial (HKL pol.) and hermite (HKL herm.) kernel
- Regularized Least Squares (RLS).

Note that, differently from the first 4 methods, RLS is not a variable selection algorithm, however we consider it since it is typically a good benchmark for the prediction error.

For $\ell^1$-features and MKL we use our own Matlab implementation based on proximal methods (for details see Mosci et al., 2010). For COSSO we used the Matlab code available at `www.stat.wisc.edu/~yilin` or `www4.stat.ncsu.edu/~hzhang` which can deal with 1 and 2-way interactions. For HKL we used the code available online at `http://www.di.ens.fr/~fbach/hkl/index.html`. While for MKL and $\ell^1$-features we are able to identify the set of selected variables, for COSSO and HKL extracting the sparsity patterns from the available code is not straightforward. We therefore compute the selection errors only for $\ell^1$-features, MKL, and our method .

### 6.3.2 SYNTHETIC DATA

We simulated data with $d$ input variables, where each variable is uniformly sampled from [-2,2]. The output $y$ is a nonlinear function of the first 4 variables, $y = f(x^1, x^2, x^3, x^4) + \varepsilon$, where epsilon is a white noise, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, and $\sigma$ is chosen so that the signal to noise ratio is 15:1. We consider the 4 models described in Table 6.3.2.

| | **d** | **number of relevant variables** | **n** | **model ($f$)** |
|---|---|---|---|---|
| additive p=2 | 40 | 4 | 100 | $y = \sum_{a=1}^{4} (x^a)^2$ |
| 2way p=2 | 40 | 4 | 100 | $y = \sum_{a=1}^{4} \sum_{b=a+1}^{4} x^a x^b$ |
| 3way p=6 | 40 | 3 | 100 | $y = (x^1 x^2 x^3)^2$ |
| radial | 20 | 2 | 100 | $y = \frac{1}{\pi}((x^1)^2 + (x^2)^2) e^{-((x^1)^2+(x^2)^2)}$ |

Table 1: Synthetic data design

For model selection and testing we follow the same protocol described at the beginning of Section 6, with $n = 100, 100$ and 1000 for training, validation and testing, respectively. Finally we average over 20 repetitions. In the first 3 models, for MKL, HKL, RLS and our method we employed the polynomial kernel of degree $p$, where $p$ is the polynomial degree of the regression function $f$. For $\ell^1$-features we used the polynomial kernel with degree chosen as the minimum between the polynomial degree of $f$ and 3. This was due to computational reasons, in fact, with $p = 4$ and $d = 40$, the number of features is highly above $100,000$. For the last model, we used the

6. In all toy data, and in part of the real data, the following warning message was displayed:

```
Maximum number of iterations exceeded; increase options.MaxIter.
To continue solving the problem with the current solution as the starting point,
set x0 = x before calling lsqlin.
```

In those cases the algorithm did not reach convergence in a reasonable amount of time, therefore the error bars corresponding to COSSO2 were omitted.

polynomial kernel of degree 4 for MKL, $\ell^1$-features and HKL, and the Gaussian kernel with kernel parameter $\sigma = 2$ for RLS and our method.[7] COSSO2 never reached convergence. Results in terms of prediction and selection errors are reported in Figure 6.3.2.

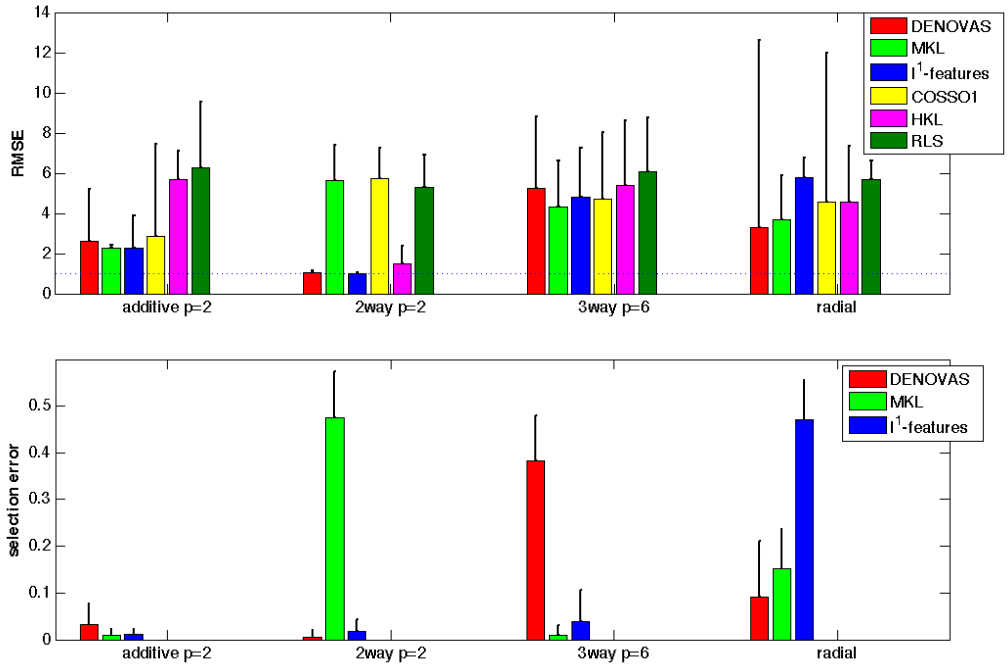



Figure 9: Prediction error (top) and fraction of selected variables (bottom) on synthetic data for the proposed method (DENOVAS), multiple kernel learning for additive models (MKL), $\ell^1$ regularization on the feature space associated to a polynomial kernel ($\ell^1$-features), COSSO with 1-way interactions (COSSO1), hierarchical kernel learning with polynomial kernel (HKL pol.), and regularized least squares (RLS). The dotted line in the upper plot corresponds to the white noise standard deviation. Selection errors for COSSO, and HKL are not reported because they are not straightforwardly computable from the available code.

When the regression function is simple (low interaction degree or low polynomial degree) more tailored algorithms, such as MKL—which is additive by design—for the experiment "additive p=2", or $\ell^1$-features for experiments "2way p=4"—in this case the dictionary size is less than 1000—compare favorably with respect to our method. However, when the nonlinearity of the regression function favors the use of a large hypotheses space, our method significantly outperforms the other methods. This is particularly evident in the experiment "radial", which was anticipated in Section 3, where we plotted in Figure 3.2 the regression function and its estimates obtained with the different algorithms for one of the 20 repetitions.

7. Note that here we are interested in evaluating the ability of our method of dealing with a general kernel like the Gaussian kernel, not in the choice of the kernel parameter itself. Nonetheless, a data driven choice for $\sigma$ will be presented in the real data experiments in Section 6.3.3.

### 6.3.3 REAL DATA

We consider the 7 benchmark data sets described in Table 6.3.3. We build training and validation

| data name | number of input variables | number of instances | source | task |
|---|---|---|---|---|
| boston housing | 13 | 506 | LIACC[8] | regression |
| census | 16 | 22784 | LIACC | regression |
| delta ailerons | 5 | 7129 | LIACC | regression |
| stock | 10 | 950 | LIACC | regression |
| image segmentation | 18 | 2310 | IDA[9] | classification |
| pole telecomm | 26[10] | 15000 | LIACC | regression |
| breast cancer | 32 | 198 | UCI[11] | regression |

Table 2: Real data sets

sets by randomly drawing $n_{\text{train}}$ and $n_{\text{val}}$ samples, and using the remaining samples for testing. For the first 6 data sets we let $n_{\text{train}} = n_{\text{val}} = 150$, whereas for breast cancer data we let $n_{\text{train}} = n_{\text{val}} = 60$. We then apply the algorithms described in Section 6.3.1. with the validation protocol described in Section 6. For our method and RLS we used the Gaussian kernel with the kernel parameter $\sigma$ chosen as the mean over the samples of the euclidean distance form the 20-th nearest neighbor. Since the other methods cannot be run with the Gaussian kernel we used a polynomial kernel of degree $p = 3$ for MKL and $\ell^1$-features. For HKL we used both the polynomial kernel and the hermite kernel, both with $p = 3$. Results in terms of prediction and selection error are reported in Figure 10.

Some of the data, such as the stock data, seem not to be variable selection problem, in fact the best performance is achieved by our method though selecting (almost) all variables, or, equivalently by RLS. Our method outperforms all other methods on several data sets. In most cases, the performance of our method and RLS are similar. Nonetheless our method brings higher interpretability since it is able to select a smaller subset of relevant variable, while the estimate provided by RLS depends on all variables.

We also run experiments on the same 7 data sets with different kernel choices for our method . We consider the polynomial kernel with degree $p = 2, 3$ and 4, and the Gaussian kernel. Comparisons among the different kernels in terms of prediction and selection accuracy are plotted in Figure 11. Interestingly the choice of the Gaussian kernel seems to be the preferable choice in most data sets.

## 7. Discussion

Sparsity based method has recently emerged as way to perform learning and variable selection from high dimensional data. So far, compared to other machine learning techniques, this class of methods suffers from strong modeling assumptions and is in fact limited to parametric or semi-parametric models (additive models). In this paper we discuss a possible way to circumvent this shortcoming and exploit sparsity ideas in a non-parametric context.

We propose to use partial derivatives of functions in a RKHS to design a new sparsity penalty and a corresponding regularization scheme. Using results from the theory of RKHS and proximal

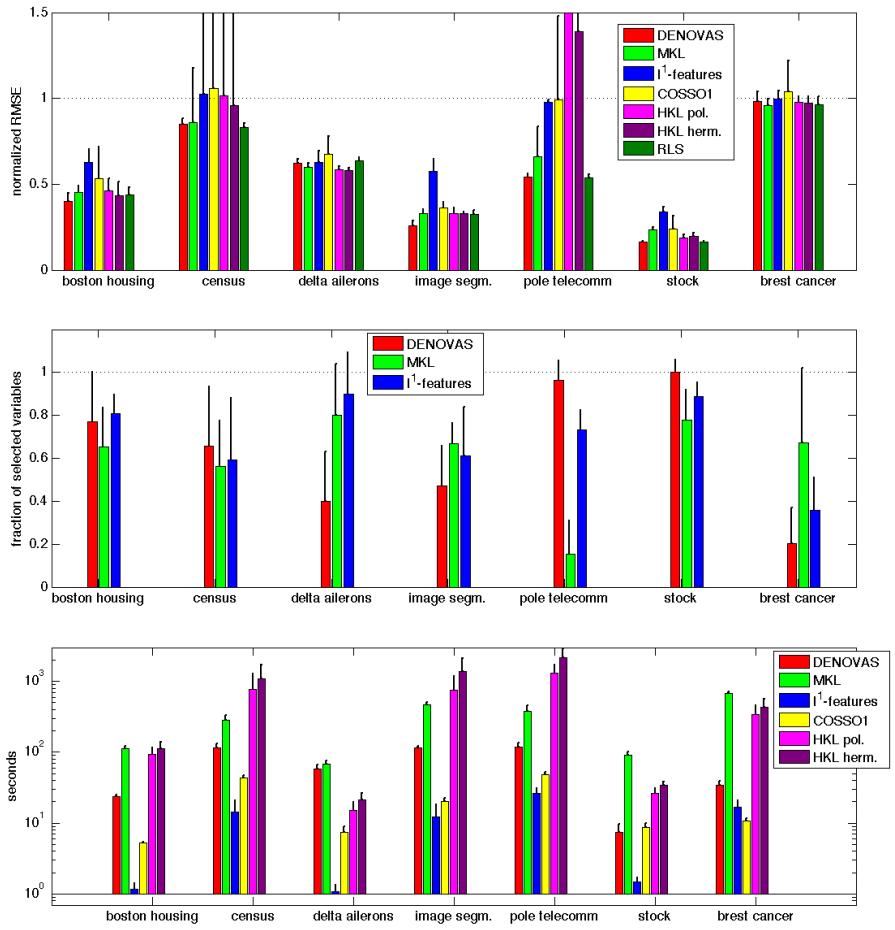



Figure 10: Prediction error (top) and fraction of selected variables (center) and computing time (bottom) on real data for the proposed method (DENOVAS), multiple kernel learning for univariate additive functions (MKL), $\ell^1$ regularization on the feature space associated to a polynomial kernel ($\ell^1$-features), COSSO with 1-way interactions (COSSO1), hierarchical kernel learning with polynomial kernel (HKL pol.), hierarchical kernel learning with hermite kernel (HKL herm.) and regularized least squares (RLS). Prediction errors for COSSO2 are not reported because it is always outperformed by COSSO1. such errors were still too large to report in the first three data sets, and were not available since the algorithm did not reach convergence for image segmentation, pole telecomm and breast cancer data. To make the prediction errors comparable among experiments, root mean squared errors (RMSE) were divided by the outputs standard deviation, which corresponds to the dotted line. Error bars are the standard deviations of the normalized RMSE. Though the largest normalized RMSE appear out of the figure axis, we preferred to display the prediction errors with the current axes limits in order to allow the reader to appreciate the difference between the smallest, and thus most significant, errors. Selection errors for COSSO, and HKL are not reported because they are not straightforwardly computable from the available code. The computing time corresponds to the entire model selection and testing protocol. Computing time for RLS is not reported because it was always negligible with respect to the other methods.

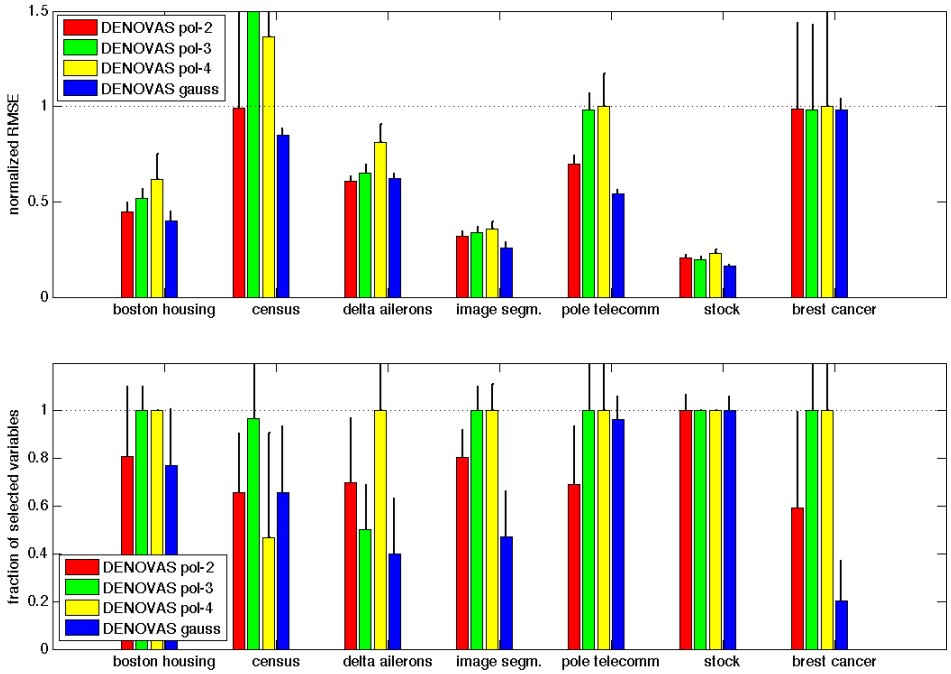



Figure 11: Prediction error (top) and fraction of selected variables (bottom) on real data for our method with different kernels: polynomial kernel of degree $p = 1, 2$ and 3 (DENOVAS pol-$p$), and Gaussian kernel (DENOVAS gauss). Error bars represent the standard deviations. In order to make the prediction errors comparable among experiments, root mean square errors were divided by the outputs standard deviation, which corresponds to the dotted line.

methods we show that the regularized estimator can be provably computed through an iterative procedure. The consistency property of the proposed estimator are studied. Exploiting the nonparametric nature of the method we can prove universal consistency. Moreover we study selection properties and show that that the proposed regularization scheme represents a safe filter for variable selection, as it does not discard relevant variables. Extensive simulations on synthetic data demonstrate the prediction and selection properties of the proposed algorithm. Finally, comparisons to state-of-the-art algorithms for nonlinear variable selection on toy data as well as on a cohort of benchmark data sets, show that our approach often leads to better prediction and selection performance.

Our work can be considered as a first step towards understanding the role of sparsity beyond additive models. It substantially differs with respect to previous approaches based on local polynomial regression (Lafferty and Wasserman, 2008; Bertin and Lecué, 2008; Miller and Hall, 2010), since it is a regularization scheme directly performing global variable selection. The RKHSs' machinery allows on the one hand to find a computationally efficient algorithmic solution, and on the other hand to consider very general probability distributions $\rho$, which are not required to have a positive

density with respect to the Lebesgue measure (differently from Comminges and Dalalyan, 2012). Several research directions are yet to be explored.

- From a theoretical point of view it would be interesting to further analyzing the sparsity property of the obtained estimator in terms of finite sample estimates for the prediction and the selection error.
- From a computational point of view the main question is whether our method can be scaled to work in very high dimensions. Current computations are limited by memory constraints. A variety of method for large scale optimization can be considered towards this end.
- A natural by product of computational improvements would be the possibility of considering a semi-supervised setting which is naturally suggested by our approach. More generally we plan to investigate the application of the RKHS representation for differential operators in unsupervised learning.
- More generally, our study begs the question of whether there are alternative/better ways to perform learning and variable selection beyond additive models and using non parametric models.

## Acknowledgments

The authors would like to thank Ernesto De Vito for many useful discussions and suggesting the proof of Lemma 4. SM and LR would like to thank Francis Bach and Guillame Obozinski for useful discussions. This paper describes a joint research work done at and at the Departments of Computer Science and Mathematics of the University of Genoa and at the IIT@MIT lab hosted in the Center for Biological and Computational Learning (within the McGovern Institute for Brain Research at MIT), at the Department of Brain and Cognitive Sciences (affiliated with the Computer Sciences and Artificial Intelligence Laboratory), The authors have been partially supported by the Integrated Project Health-e-Child IST-2004-027749 and by grants from DARPA (IPTO and DSO), National Science Foundation (NSF-0640097, NSF-0827427), and Compagnia di San Paolo, Torino. Additional support was provided by: Adobe, Honda Research Institute USA, King Abdullah University Science and Technology grant to B. DeVore, NEC, Sony and especially by the Eugene McDermott Foundation.

## Appendix A. Derivatives in RKHS and Representer Theorem

Consider $L^2(\mathcal{X},\rho_\mathcal{X}) = \{f : \mathcal{X} \to \mathbb{R} \text{ measurable}| \ \int |f(x)|^2 d\rho_\mathcal{X}(x) < \infty\}$ and $\mathbb{R}^n$ with inner product normalized by a factor $1/n$, $\|\cdot\|_n$.

The operator $I_k : \mathcal{H} \to L^2(\mathcal{X},\rho_\mathcal{X})$ defined by $(I_k f)(x) = \langle f, k_x\rangle_\mathcal{H}$, for almost all $x \in X$, is well-defined and bounded thanks to assumption (A1). The sampling operator (18) can be seen as its empirical counterpart. Similarly $D_a : \mathcal{H} \to L^2(\mathcal{X},\rho_\mathcal{X})$ defined by $(D_a f)(x) = \langle f, (\partial_a k)_x\rangle$, for almost all $x \in X$ and $a = 1,\dots d$, is well-defined and bounded thanks to assumption (A2). The operator (20) can be seen as its empirical counterpart. Several properties of such operators and related quantities are given by the following two propositions.

**Proposition 12** *If assumptions (A1) and (A2) are met, the operator $I_k$ and the continuous partial derivative $D_a$ are Hilbert-Schmidt operators from $\mathcal{H}$ to $L^2(\mathcal{X},\rho_\mathcal{X})$, and*

$$I_k^* g(t) = \int_\mathcal{X} k_x(t) g(x) d\rho_\mathcal{X}(x), \qquad I_k^* I_k f(t) = \int_\mathcal{X} k_x(t) \langle f, k_x \rangle_\mathcal{H} d\rho_\mathcal{X}(x),$$
$$D_a^* g(t) = \int_\mathcal{X} (\partial_a k)_x(t) g(x) d\rho_\mathcal{X}(x), \qquad D_a^* D_b f(t) = \int_\mathcal{X} (\partial_a k)_x(t) \langle f, (\partial_b k)_x \rangle_\mathcal{H} d\rho_\mathcal{X}(x).$$

**Proposition 13** *If assumptions (A1) and (A2) are met, the sampling operator $\hat{S}$ and the empirical partial derivative $\hat{D}_a$ are Hilbert-Schmidt operators from $\mathcal{H}$ to $\mathbb{R}^n$, and*

$$\hat{S}^* v = \frac{1}{n}\sum_{i=1}^n k_{x_i} v_i, \qquad \hat{S}^* \hat{S} f = \frac{1}{n}\sum_{i=1}^n k_{x_i} \langle f, k_{x_i} \rangle_\mathcal{H},$$
$$\hat{D}_a^* v = \frac{1}{n}\sum_{i=1}^n (\partial_a k)_{x_i} v_i, \qquad \hat{D}_a^* \hat{D}_b f = \frac{1}{n}\sum_{i=1}^n (\partial_a k)_{x_i} \langle f, (\partial_b k)_{x_i} \rangle_\mathcal{H}$$

*where $a,b = 1,\dots,d$.*

The proof can be found in De Vito et al. (2005) for $I_k$ and $\hat{S}$, where assumption (A1) is used. The proof for $D_a$ and $\hat{D}_a$ is based on the same tools and on assumption (A2). Furthermore, a similar result can be obtained for the continuous and empirical gradient

$$\nabla : \mathcal{H} \to (L^2(\mathcal{X},\rho_\mathcal{X}))^d, \qquad \nabla f = (D_a f)_{a=1}^d,$$
$$\hat{\nabla} : \mathcal{H} \to (\mathbb{R}^n)^d, \qquad \hat{\nabla} f = (\hat{D}_a f)_{a=1}^d,$$

which can be shown to be Hilbert-Schmidt operators from $\mathcal{H}$ to $(L^2(\mathcal{X},\rho_\mathcal{X}))^d$ and from $\mathcal{H}$ to $(\mathbb{R}^n)^d$, respectively.

We next restate Proposition 3 in a slightly more abstract form and give its proof.

**Proposition 3 (Extended)** *The minimizer of (6) satisfies $\hat{f}^\tau \in Range(\hat{S}^*) + Range(\hat{\nabla}^*)$. Henceforth it satisfies the following representer theorem*

$$\hat{f}^\tau = \hat{S}^*\alpha + \hat{\nabla}^*\beta = \sum_{i=1}^n \frac{1}{n}\alpha_i k_{x_i} + \sum_{i=1}^n \sum_{a=1}^d \frac{1}{n}\beta_{a,i}(\partial_a k)_{x_i} \tag{34}$$

*with $\alpha \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^{nd}$.*

**Proof** Being $\mathrm{Range}(\hat{S}^*) + \mathrm{Range}(\hat{\nabla}^*)$ a closed subspace of $\mathcal{H}$, we can write any function $f \in \mathcal{H}$ as $f = f^{//} + f^\perp$, where $f^{//} \in \mathrm{Range}(\hat{S}^*) + \mathrm{Range}(\hat{\nabla}^*)$ and $\langle f^\perp, g \rangle_\mathcal{H}$ for all $g \in \mathrm{Range}(\hat{S}^*) + \mathrm{Range}(\hat{\nabla}^*)$. Now if we plug the decomposition $f = f^{//} + f^\perp$ in the variational problem (6), we obtain

$$\hat{f}^\tau = \operatorname*{argmin}_{f\in\mathcal{H},\ f=f^{//}+f^\perp} \left\{ \widehat{\mathcal{E}}(f^{//}) + 2\tau\widehat{\Omega}_1^D(f^{//}) + \tau\nu\|f^{//}\|_\mathcal{H}^2 + \tau\nu\|f^\perp\|_\mathcal{H}^2 \right\}$$

which is clearly minimized by $f^\perp = 0$. The second equality in (34) then derives directly from definition of $\hat{S}^*$ and $\hat{\nabla}^*$. ■

We conclude with the following example on how to compute derivatives and related quantities for the Gaussian Kernel.

**Example 1** *Note that all the terms involved in* (30) *are explicitly computable. As an example we show how to compute them when* $k(x,s) = e^{-\frac{\|x-s\|^2}{2\gamma^2}}$ *is the Gaussian kernel on* $\mathbb{R}^d$*. By definition*

$$(\partial_a k)_{x_i}(x) = \langle \left.\frac{\partial k(s,\cdot)}{\partial s^a}\right|_{s=x_i}, k_x\rangle_{\mathcal{H}}.$$

*Given* $x \in \mathcal{X}$ *it holds*

$$\frac{\partial k(s,x)}{\partial s^a} = e^{-\frac{\|x-s\|^2}{2\gamma^2}} \cdot \left(-\frac{s^a - x^a}{\gamma^2}\right) \quad \Longrightarrow \quad (\partial_a k)_{x_i}(x) = e^{-\frac{\|x-x_i\|^2}{2\gamma^2}} \cdot \left(-\frac{x_i^a - x^a}{\gamma^2}\right).$$

*Moreover, as we mentioned above, the computation of* $\beta_{a,i}^t$ *and* $\alpha_i^t$ *requires the knowledge of matrices* $\mathrm{K}, \mathrm{Z}_a, \mathrm{Z}, \mathrm{L}_a$*. Also their entries are easily found starting from the kernel and the training points. We only show how the entries of* $\mathrm{Z}$ *and* $\mathrm{L}_a$ *look like. Using the previous computations we immediately get*

$$[\mathrm{Z}_a]_{i,j} = e^{-\frac{\|x_j-x_i\|^2}{2\gamma^2}} \cdot \left(-\frac{x_i^a - x_j^a}{\gamma^2}\right).$$

*In order to compute* $\mathrm{L}_a$ *we need the second partial derivatives of the kernel:*

$$\frac{\partial k(s,x)}{\partial x^b \partial s^a} = \begin{cases} -e^{-\frac{\|x-s\|^2}{2\gamma^2}} \cdot \frac{s^a - x^a}{\gamma^2} \cdot \frac{s^b - x^b}{\gamma^2} & \text{if } a \neq b \\ -e^{-\frac{\|x-s\|^2}{2\gamma^2}} \cdot \left(\frac{(s^a - x^a)^2}{\gamma^2} - \frac{1}{\gamma^2}\right) & \text{if } a = b. \end{cases}$$

*so that*

$$[\mathrm{L}_{a,b}]_{i,j} = \begin{cases} -e^{-\frac{\|x_j-x_i\|^2}{2\gamma^2}} \cdot \frac{x_i^a - x_j^a}{\gamma^2} \cdot \frac{x_i^b - x_j^b}{\gamma^2} & \text{if } a \neq b \\ -e^{-\frac{\|x_j-x_i\|^2}{2\gamma^2}} \cdot \left(\frac{(x_i^a - x_j^a)^2}{\gamma^2} - \frac{1}{\gamma^2}\right) & \text{if } a = b. \end{cases}$$

## Appendix B. Proofs of Section 4

In this appendix we collect the proofs related to the derivation of the iterative procedure given in Algorithm 1. Theorem 4 is a consequence of the general results about convergence of accelerated and inexact FB-splitting algorithms in Villa et al.. In that paper it is shown that inexact schemes converge only when the errors in the computation of the proximity operator are of a suitable type and satisfy a sufficiently fast decay condition. We first introduce the notion of admissible approximations.

**Definition 14** *Let* $\varepsilon \geq 0$ *and* $\lambda > 0$*. We say that* $h \in \mathcal{H}$ *is an* approximation of $\mathrm{prox}_{\lambda\widehat{\Omega}_1^D}(f)$ with $\varepsilon$-precision *and we write* $h \approxeq_\varepsilon \mathrm{prox}_{\lambda\widehat{\Omega}_1^D}(f)$*, if and only if*

$$\frac{f-h}{\lambda} \in \partial_{\frac{\varepsilon^2}{2\lambda}} \widehat{\Omega}_1^D(h),$$

*where* $\partial_{\frac{\varepsilon^2}{2\lambda}}$ *denotes the* $\varepsilon$*-subdifferential.*[12]

We will need the following results from Villa et al..

**Theorem 15** *Consider the following inexact version of the accelerated FB-algorithm in* (21) *with* $c_{1,t}$ *and* $c_{2,t}$ *as in* (23)

$$f^t \approxeq_{\varepsilon_t} prox_{\frac{\tau}{\sigma}\widehat{\Omega}_1^D}\Big(\Big(I-\frac{1}{2\sigma}\nabla F\Big)(c_{1,t}f^{t-1}+c_{2,t}f^{t-2})\Big). \tag{35}$$

*Then, if* $\varepsilon_t \sim 1/t^l$ *with* $l > 3/2$*, there exists a constant* $C > 0$ *such that*

$$\widehat{\mathcal{E}}^\tau(f^t) - \inf \widehat{\mathcal{E}}^\tau \leq \frac{C}{t^2}.$$

**Proposition 16** *Suppose that* $\Omega : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ *can be written as* $\Omega = \omega \circ B$*, where* $B : \mathcal{H} \to \mathcal{G}$ *is a linear and bounded operator between Hilbert spaces and* $\omega : \mathcal{G} \to \mathbb{R} \cup \{+\infty\}$ *is a one-homogeneous function such that* $\partial\omega(0)$ *is bounded. Then for any* $f \in \mathcal{H}$ *and any* $v \in \partial\omega(0)$ *such that*

$$2\lambda\omega(B(f-\lambda B^* v)) - 2\langle \lambda B^* v, f\rangle \leq \varepsilon^2 \tag{36}$$

*it holds*

$$f - \lambda B^* v \approxeq_\varepsilon \operatorname{prox}_{\lambda\Omega}(f).$$

**Proposition 17** *Under the same assumptions of Proposition 16, for any* $f \in \mathcal{H}$*,* $\varepsilon > 0$*, and sequence* $\{v^q\}$ *minimizing the dual problem*

$$\|f - \lambda B^* v\|_{\mathcal{H}} + i_S(v),$$

*where* $i_S$ *is the indicator function of S, there exists* $\bar{q}$ *such that for any* $q > \bar{q}$*,* $f - \lambda B^* v^q$ *satisfies condition* (36)*.*

**Proof** [Proof of Theorem 4] Since the the regularizer $\widehat{\Omega}_1^D$ can be written as a composition of $\omega \circ B$, with $B = \hat{\nabla}$ and $\omega : \mathbb{R}^d \to [0,+\infty)$, $\omega(v) = \sum_{a=1}^d \|v_a\|_n$ and $\{v^q\}$ is a minimizing sequence for problem (25), Proposition 17 ensures that $v^q$ satisfies condition (28) if $q$ is big enough. Therefore, the sequence $\hat{\nabla}^* v^q$ obtained from $v^q$ generates, via (27), admissible approximations of $\operatorname{prox}_{\frac{\tau}{\sigma}\widehat{\Omega}_1^D}$.

Therefore, if $\varepsilon_t$ is such that $\varepsilon_t \sim 1/t^l$ with $l > 3/2$, Theorem 15 implies that the inexact version of the FB-splitting algorithm in (27) shares the $1/t^2$ convergence rate. Equation (29) directly follows from the definition of strong convexity,

$$\frac{\tau\nu}{8}\|f^t - \hat{f}^\tau\|^2 \leq \widehat{\mathcal{E}}^\tau(f^t)/2 + \widehat{\mathcal{E}}^\tau(\hat{f}^\tau)/2 - \widehat{\mathcal{E}}^\tau(f^t/2 + \hat{f}^\tau/2) \leq \frac{1}{2}(\widehat{\mathcal{E}}^\tau(f^t) - \widehat{\mathcal{E}}^\tau(\hat{f}^\tau)).$$

■

12. Recall that the $\varepsilon$-subdifferential $\partial_\varepsilon$ of a convex functional $\Omega : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ is defined as the set

$$\partial_\varepsilon\Omega(f) := \{h \in \mathcal{H} : \Omega(g) - \Omega(f) \geq \langle h, g-f\rangle_{\mathcal{H}} - \varepsilon,\ \forall g \in \mathcal{H}\}, \qquad \forall f \in \mathcal{H}.$$

**Proof** [Proof of Proposition 5] We first show that the matrices $\mathrm{K}, \mathrm{Z}_a, \mathrm{L}_a$ defined in (8), (9), and (10), are the matrices associated to the operators $\hat{S}\hat{S}^* : \mathbb{R}^n \to \mathbb{R}^n$, $\hat{S}\hat{D}_a^* : \mathbb{R}^n \to \mathbb{R}^n$ and $\hat{D}_a\hat{\nabla}^* : \mathbb{R}^{nd} \to \mathbb{R}^n$, respectively. For K, the proof is trivial and derives directly from the definition of adjoint of $\hat{S}$—see Proposition 13. For $\mathrm{Z}_a$ and Z, from the definition of $\hat{D}_a^*$ we have that

$$\left(\hat{S}\hat{D}_a^*\alpha\right)_i = \frac{1}{n}\sum_{j=1}^n \alpha_j(\partial_a k)_{x_j}(x_i) = \sum_{j=1}^n (\mathrm{Z}_a)_{i,j}\alpha_j = (\mathrm{Z}_a\alpha)_i,$$

so that $\hat{S}\hat{\nabla}^*\beta = \sum_{a=1}^d \hat{S}\hat{D}_a^*(\beta_{a,i})_{i=1}^n = \sum_{a=1}^d \mathrm{Z}_a(\beta_{a,i})_{i=1}^n = \mathrm{Z}\beta$. For $\mathrm{L}_a$, we first observe that

$$\langle(\partial_a k)_x, (\partial_b k)_{x'}\rangle_{\mathcal{H}} = \left.\frac{\partial(\partial_b k)_{x'}(t)}{\partial t^a}\right|_{t=x} = \left.\frac{\partial^2 k(s,t)}{\partial t^a \partial s^b}\right|_{t=x,s=x'},$$

so that operator $\hat{D}_a\hat{D}_b^* : \mathbb{R}^n \to \mathbb{R}^n$ is given by

$$\left(\left(\hat{D}_a\hat{D}_b^*\right)v\right)_i = \langle(\partial_a k)_{x_i}, \hat{D}_b^* v\rangle_{\mathcal{H}} = \frac{1}{n}\sum_{j=1}^n \langle(\partial_a k)_{x_i}, (\partial_b k)_{x_j}\rangle_{\mathcal{H}} v_j = \left(\mathrm{L}_{a,b}\right)_{i,j} v_j$$

for $i = 1, \ldots, n$, for all $v \in \mathbb{R}^n$. Then, since $\hat{D}_a\hat{\nabla}^*\beta = \sum_{a=1}^d \hat{D}_a\hat{D}_b^*(\beta_{a,i})_{i=1}^n$, we have that $\mathrm{L}_a$ is the matrix associated to the operator $\hat{D}_a\hat{\nabla}^* : \mathbb{R}^{nd} \to \mathbb{R}^n$, that is

$$(\hat{D}_a\hat{\nabla}^*\beta)_i = \sum_{j=1}^n \sum_{b=1}^d (\mathrm{L}_{a,b})_{i,j}\beta_{b,j},$$

for $i = 1, \ldots, n$, for all $\beta \in \mathbb{R}^{nd}$. To prove equation (30), first note that, as we have done in Proposition 3 extended, (30) can be equivalently rewritten as $f^t = \hat{S}^*\alpha^t + \hat{\nabla}^*\beta^t$. We now proceed by induction. The base case, namely the representation for $t = 0$ and $t = 1$, is clear. Then, by the inductive hypothesis we have that $f^{t-1} = \hat{S}^*\alpha^{t-1} + \hat{\nabla}^*\beta^{t-1}$, and $f^{t-2} = \hat{S}^*\alpha^{t-2} + \hat{\nabla}^*\beta^{t-2}$ so that $\tilde{f}^t = \hat{S}^*\tilde{\alpha}^t + \hat{\nabla}^*\tilde{\beta}^t$ with $\tilde{\alpha}^t$ and $\tilde{\beta}^t$ defined by (12) and (13). Therefore, using (21), (27), (24) it follows that $f^t$ can be expressed as:

$$\left(I - \pi_{\frac{\tau}{\sigma}C_n}\right)\left(\hat{S}^*\left(\left(1 - \frac{\tau\nu}{\sigma}\right)\tilde{\alpha}^t - \frac{1}{\sigma}\left(\mathrm{K}\tilde{\alpha}^t + \mathrm{Z}\tilde{\beta}^t - \mathrm{y}\right)\right) + \left(1 - \frac{\tau\nu}{\sigma}\right)\hat{\nabla}^*\tilde{\beta}^t\right)$$

and the proposition is proved, letting $\tilde{\alpha}^t$, $\tilde{\beta}^t$ and $\bar{v}^t$ as in Equations (14), (16) and (25).

For the projection, we first observe that operator $\hat{D}_a\hat{S}^* : \mathbb{R}^n \to \mathbb{R}^n$ is given by

$$(\hat{D}_a\hat{S}^*\alpha)_i = (\langle\hat{S}^*\alpha, (\partial_a k)_{x_i}\rangle_{\mathcal{H}}) = \frac{1}{n}\sum_{j=1}^n \alpha_j(\partial_a k)_{x_i}(x_j) = \sum_{j=1}^n \alpha_j(\mathrm{Z}_a)_{j,i} = \mathrm{Z}_a^T\alpha.$$

Then, we can plug the representation (34) in (27) to obtain (15). ■

**Proof** [Proof of Proposition 6] Since $\hat{f}^\tau$ is the unique minimizer of the functional $\widehat{\mathcal{E}}^\tau$, it satisfies the Euler equation for $\widehat{\mathcal{E}}^\tau$

$$0 \in \partial(\widehat{\mathcal{E}}(\hat{f}^\tau) + 2\tau\widehat{\Omega}_1^D(\hat{f}^\tau) + \tau\nu\|\hat{f}^\tau\|_{\mathcal{H}}^2).$$

where, for an arbitrary $\lambda > 0$, the subdifferential of $\lambda\widehat{\Omega}_1^D$ at $f$ is given by

$$\partial\lambda\widehat{\Omega}_1^D(f) = \{\hat{\nabla}^* v, v = (v_a)_{a=1}^d \in (\mathbb{R}^n)^d \mid v_a = \lambda \hat{D}_a f / \|\hat{D}_a f\|_n \text{ if } \|\hat{D}_a f\|_n > 0, \\ \text{and } \|v_a\|_n \leq \lambda \text{ otherwise}, \forall a = 1, \ldots, d\}.$$

Using the above characterization and the fact that $\widehat{\mathcal{E}} + \tau\nu\|\cdot\|_{\mathcal{H}}^2$ is differentiable, the Euler equation is equivalent to

$$\hat{\nabla}^* v = -\frac{1}{2\sigma}\nabla(\widehat{\mathcal{E}} + \tau\nu\|\cdot\|_{\mathcal{H}}^2)(\hat{f}^\tau),$$

for any $\sigma > 0$ , and for some $v = (v_a)_{a=1}^d \in (\mathbb{R}^n)^d$ with $v = (v_a)_{a=1}^d \in (\mathbb{R}^n)^d$ such that

$$\begin{aligned} v_a &= \frac{\tau}{\sigma}\frac{\hat{D}_a \hat{f}^\tau}{\|\hat{D}_a \hat{f}^\tau\|_n} \quad \text{if} \quad \|\hat{D}_a \hat{f}^\tau\|_n > 0, \\ v_a &\in \frac{\tau}{\sigma} B_n \qquad\qquad \text{otherwise.} \end{aligned}$$

In order to prove (31), we proceed by contradiction and assume that $\|\hat{D}_a \hat{f}^\tau\|_n > 0$. This would imply $\|v_a\|_n = \tau/\sigma$, which contradicts the assumption, hence $\|\hat{D}_a \hat{f}^\tau\|_n = 0$.

We now prove (32). First, according to Definition 14 (see also Theorem 4.3 in Villa et al. and Beck and Teboulle (2009) for the case when the proximity operator is evaluated exactly), the algorithm generates by construction sequences $\tilde{f}^t$ and $f^t$ such that

$$\tilde{f}^t - f^t - \frac{1}{2\sigma}\nabla F(\tilde{f}^t) \in \frac{1}{2\sigma}\partial_{\sigma(\varepsilon_t)^2} 2\tau\widehat{\Omega}_1^D(f^t) = \frac{\tau}{\sigma}\partial_{\frac{\sigma}{2\tau}(\varepsilon_t)^2}\widehat{\Omega}_1^D(f^t).$$

where $\partial_\varepsilon$ denotes the $\varepsilon$-subdifferential.[13] Plugging the definition of $f^t$ from (27) in the above equation, we obtain $\hat{\nabla}^*\vec{v}^t \in \frac{\tau}{\sigma}\partial_{\frac{\sigma}{2\tau}(\varepsilon_t)^2}\widehat{\Omega}_1^D(f^t)$. Now, we can use a kind of *transportation formula* (Hiriart-Urruty and Lemaréchal, 1993) for the $\varepsilon$-subdifferential to find $\tilde{\varepsilon}_t$ such that $\hat{\nabla}^*\vec{v}^t \in \frac{\tau}{\sigma}\partial_{\frac{\sigma}{2\tau}(\tilde{\varepsilon}_t)^2}\widehat{\Omega}_1^D(f^\tau)$. By definition of $\varepsilon$-subdifferential:

$$\widehat{\Omega}_1^D(f) - \widehat{\Omega}_1^D(f^t) \geq \langle\frac{\sigma}{\tau}\hat{\nabla}^*\vec{v}^t, f - f^t\rangle_{\mathcal{H}} - \frac{\sigma}{2\tau}(\varepsilon_t)^2, \qquad \forall f \in \mathcal{H}.$$

Adding and subtracting $\widehat{\Omega}_1^D(\hat{f}^\tau)$ and $\langle\frac{\sigma}{\tau}\hat{\nabla}^* v^t, \hat{f}^\tau\rangle$ to the previous inequality we obtain

$$\widehat{\Omega}_1^D(f) - \widehat{\Omega}_1^D(\hat{f}^\tau) \geq \langle\frac{\sigma}{\tau}\hat{\nabla}^*\vec{v}^t, f - \hat{f}^\tau\rangle_{\mathcal{H}} - \frac{\sigma}{2\tau}(\tilde{\varepsilon}_t)^2,$$

with

$$(\tilde{\varepsilon}_t)^2 = (\varepsilon_t)^2 + \frac{2\tau}{\sigma}\Big(\widehat{\Omega}_1^D(\hat{f}^\tau) - \widehat{\Omega}_1^D(f^t) - \langle 2\hat{\nabla}^*\vec{v}^t, \hat{f}^\tau - f^t\rangle_{\mathcal{H}}\Big).$$

From the previous equation, using (29) we have

$$(\tilde{\varepsilon}_t)^2 \leq (\varepsilon_t)^2 + \sqrt{\frac{C}{\nu\tau}}\left(\frac{\tau}{\sigma}\sum_a \sqrt{\|\hat{D}_a^*\hat{D}\|} + 1\right)\frac{4}{t}, \tag{37}$$

13. Recall that the $\varepsilon$-subdifferential, $\partial_\varepsilon$, of a convex functional $\Omega : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ is defined as the set

$$\partial_\varepsilon\Omega(f) := \{h \in \mathcal{H} : \Omega(g) - \Omega(f) \geq \langle h, g - f\rangle_{\mathcal{H}} - \varepsilon, \ \forall g \in \mathcal{H}\}, \qquad \forall f \in \mathcal{H}.$$

which implies $\frac{\sigma}{\tau}\hat{\nabla}^* \vec{v}^t \in \partial_{\sigma(\tilde{\varepsilon}_t)^2/2\tau}\widehat{\Omega}_1^D(\hat{f}^\tau)$. Now, relying on the structure of $\widehat{\Omega}_1^D$, it is easy to see that

$$\partial_\varepsilon \widehat{\Omega}_1^D(f) \subseteq \{\hat{\nabla}^* v, v=(v_a)_{a=1}^d \in (\mathbb{R}^n)^d \mid \|v_a\|_n \geq 1-\varepsilon/\|\hat{D}_a f\|_n \text{ if } \|\hat{D}_a f\|_n > 0\}.$$

Thus, if $\|\hat{D}_a \hat{f}^\tau\|_n > 0$ we have $\|\vec{v}^t\|_n \geq \frac{\tau}{\sigma}(1 - \frac{(\tilde{\varepsilon}_t)^2}{2\|\hat{D}_a \hat{f}^\tau\|_n})$. ■

## Appendix C. Proofs of Section 5

We start proving the following preliminary probabilistic inequalities.

**Lemma 18** *For $0 < \eta_1, \eta_2, \eta_3, \eta_4 \leq 1$, $n \in \mathbb{N}$, it holds*

*(1)* $\mathrm{P}\Big(\Big|\|\mathbf{y}\|_n^2 - \int_{\mathcal{X}\times\mathcal{Y}} y^2 d\rho(x,y)\Big| \leq \varepsilon(n,\eta_1)\Big) \geq 1-\eta_1$ *with* $\varepsilon(n,\eta_1) = \dfrac{2\sqrt{2}}{\sqrt{n}} M^2 \log\dfrac{2}{\eta_1}$,

*(2)* $\mathrm{P}\big(\|\hat{S}^*\mathbf{y} - I_k^* f_\rho\|_{\mathcal{H}} \leq \varepsilon(n,\eta_2)\big) \geq 1-\eta_2$ *with* $\varepsilon(n,\eta_2) = \dfrac{2\sqrt{2}}{\sqrt{n}} \kappa_1 M \log\dfrac{2}{\eta_2}$,

*(3)* $\mathrm{P}\big(\|\hat{S}^*\hat{S} - I_k^* I_k\| \leq \varepsilon(n,\eta_3)\big) \geq 1-\eta_3$ *with* $\varepsilon(n,\eta_3) = \dfrac{2\sqrt{2}}{\sqrt{n}} \kappa_1^2 \log\dfrac{2}{\eta_3}$,

*(4)* $\mathrm{P}\big(\|\hat{D}_a^*\hat{D}_a - D_a^* D_a\| \leq \varepsilon(n,\eta_4)\big) \geq 1-\eta_4$ *with* $\varepsilon(n,\eta_4) = \dfrac{2\sqrt{2}}{\sqrt{n}} \kappa_2^2 \log\dfrac{2}{\eta_4}$.

**Proof** From standard concentration inequalities for Hilbert space valued random variables—see, for example, Pinelis and Sakhanenko (1985)—we have that, if $\xi$ is a random variable with values in a Hilbert space $\mathcal{H}$ bounded by $L$ and $\xi_1, \ldots, \xi_n$ are $n$ i.i.d. samples, then

$$\|\frac{1}{n}\sum_{i=1}^n \xi_i - \mathbb{E}(\xi)\| \leq \varepsilon(n,\eta) = \frac{2\sqrt{2}}{\sqrt{n}} L \log\frac{2}{\eta}$$

with probability at least $1-\eta$, $\eta \in [0,1]$. The proof is a direct application of the above inequalities to the random variables,

*(1)* $\xi = y^2$ $\quad \xi \in \mathbb{R}$ $\quad$ with $\quad \sup_{\mathbf{z}_n} \|\xi\| \leq M^2$,
*(2)* $\xi = k_x y$ $\quad \xi \in \mathcal{H}\otimes\mathbb{R}$ $\quad$ with $\quad \sup_{\mathbf{z}_n} \|\xi\| \leq \kappa_1 M$,
*(3)* $\xi = \langle\cdot, k_x\rangle_{\mathcal{H}} k_x$ $\quad \xi \in \mathcal{H}S(\mathcal{H})$ $\quad$ with $\quad \sup_{\mathbf{z}_n} \|\xi\|_{\mathcal{H}S}(\mathcal{H}) \leq \kappa_1^2$,
*(4)* $\xi = \langle\cdot, (\partial_a k)_x\rangle_{\mathcal{H}} (\partial_a k)_x$ $\quad \xi \in \mathcal{H}S(\mathcal{H})$ $\quad$ with $\quad \sup_{\mathbf{z}_n} \|\xi\|_{\mathcal{H}S}(\mathcal{H}) \leq \kappa_2^2$.

where $\mathcal{H}S(\mathcal{H}), \|\cdot\|_{\mathcal{H}S(\mathcal{H})}$ are the space of Hilbert-Schmidt operators on $\mathcal{H}$ and the corresponding norm, respectively (note that in the final bound we upper-bound the operator norm by the Hilbert-Schmidt norm).

■

### C.1 Proofs of the Consistency of the Regularizer

We restate Theorem 7 in an extended form.

**Theorem 7 (Extended)** *Let $r < \infty$, then under assumption (A2), for any $\eta > 0$,*

$$\mathrm{P}\left(\sup_{\|f\|_{\mathcal{H}} \leq r} |\widehat{\Omega}_1^D(f) - \Omega_1^D(f)| \geq rd\frac{2\sqrt{2}}{(n)^{1/4}}\kappa_2\sqrt{\log\frac{2d}{\eta}}\right) < \eta. \tag{38}$$

*Consequently*

$$\lim_{n\to\infty} \mathrm{P}\left( \sup_{\|f\|_{\mathcal{H}}\leq r} |\widehat{\Omega}_1^D(f) - \Omega_1^D(f)| > \varepsilon \right) = 0, \qquad \forall \varepsilon > 0.$$

**Proof**

For $f \in \mathcal{H}$ consider the following chain of inequalities,

$$\begin{aligned}
|\widehat{\Omega}_1^D(f) - \Omega_1^{\mathrm{D}}(f)| &\leq \sum_{a=1}^{d} \left| \|\hat{D}_a f\|_n - \|D_a f\|_{\rho_X} \right| \\
&\leq \sum_{a=1}^{d} \left( \left| \|\hat{D}_a f\|_n^2 - \|D_a f\|_{\rho_X}^2 \right| \right)^{1/2} \\
&= \sum_{a=1}^{d} \left( \left| \langle f, (\hat{D}_a^* \hat{D}_a - {D_a}^* D_a) f \rangle_{\mathcal{H}} \right| \right)^{1/2} \\
&\leq \sum_{a=1}^{d} \|\hat{D}_a^* \hat{D}_a - {D_a}^* D_a\|^{1/2} \|f\|_{\mathcal{H}},
\end{aligned}$$

that follows from $|\sqrt{x} - \sqrt{y}| \leq \sqrt{|x-y|}$, the definition of $\hat{D}_a, D_a$ and basic inequalities. Then, using $d$ times inequality (d) in Lemma 18 with $\eta/d$ in place of $\eta_4$, and taking the supremum on $f \in \mathcal{H}$ such that $\|f\|_{\mathcal{H}} \leq r$, we have with probability $1-\eta$,

$$\sup_{\|f\|_{\mathcal{H}}\leq r} |\widehat{\Omega}_1^D(f) - \Omega_1^{\mathrm{D}}(f)| \leq rd \frac{2\sqrt{2}}{(n)^{1/4}} \kappa_2 \sqrt{\log \frac{2d}{\eta}}.$$

The last statement of the theorem easily follows. ■

### C.1.1 CONSISTENCY PROOFS

To prove Theorem 8, we need the following lemma.

**Lemma 19** *Let* $\eta \in (0,1]$. *Under assumptions (A1) and (A3), we have*

$$\sup_{\|f\|_{\mathcal{H}}\leq r} |\widehat{\mathcal{E}}(f) - \mathcal{E}(f)| \leq \frac{2\sqrt{2}}{\sqrt{n}} \left( \kappa_1^2 r^2 + 2\kappa_1 M r + M^2 \right) \log \frac{6}{\eta},$$

*with probability* $1-\eta$.

**Proof** Recalling the definition of $I_k$ we have that,

$$\begin{aligned}
\mathcal{E}(f) &= \int_{\mathcal{X}\times\mathcal{Y}} (I_k f(x) - y)^2 d\rho(x,y) \\
&= \int_{\mathcal{X}} (I_k f(x))^2 d\rho_{\mathcal{X}}(x) + \int_{\mathcal{X}\times\mathcal{Y}} y^2 d\rho(x,y) - 2\int_{\mathcal{X}\times\mathcal{Y}} I_k f(x) y d\rho(x,y) \\
&= \int_{\mathcal{X}} (I_k f(x))^2 d\rho_{\mathcal{X}}(x) + \int_{\mathcal{X}\times\mathcal{Y}} y^2 d\rho(x,y) - 2\int_{\mathcal{X}} I_k f(x) f_\rho(x) d\rho_{\mathcal{X}}(x) \\
&= \langle f, I_k^* I_k f \rangle_{\mathcal{H}} + \int_{\mathcal{X}\times\mathcal{Y}} y^2 d\rho(x,y) - 2\langle f, I_k^* f_\rho \rangle_{\mathcal{H}}.
\end{aligned}$$

Similarly $\widehat{\mathcal{E}}(f) = \langle f, \hat{S}^*\hat{S}f\rangle_{\mathcal{H}} + \|\mathbf{y}\|_n^2 - 2\langle f, \hat{S}^* f_\rho\rangle_{\mathcal{H}}$. Then, for all $f \in \mathcal{H}$, we have the bound

$$|\widehat{\mathcal{E}}(f) - \mathcal{E}(f)| \leq \|\hat{S}^*\hat{S} - I_k^* I_k\| \|f\|_{\mathcal{H}}^2 + 2\|\hat{S}^*\mathbf{y} - I_k^* f_\rho\|_{\mathcal{H}} \|f\|_{\mathcal{H}} + \left| \|\mathbf{y}\|_n^2 - \int_{X\times Y} y^2 d\rho(x,y) \right|$$

The proof follows applying Lemma 18 with probabilities $\eta_1 = \eta_2 = \eta_3 = \eta/3$. ■

We now prove Theorem 8. We use the following standard result in regularization theory (see, for example, Dontchev and Zolezzi, 1993) to control the the approximation error.

**Proposition 20** *Let $\tau_n \to 0$, be a positive sequence. Then we have that*

$$\mathcal{E}^{\tau_n}(f^{\tau_n}) - \inf_{f\in\mathcal{H}} \mathcal{E}(f) \to 0.$$

**Proof** [Proof of Theorem 8] We recall the standard sample/approximation error decomposition

$$\mathcal{E}(\hat{f}^\tau) - \inf_{f\in\mathcal{H}} \mathcal{E}(f) \leq |\mathcal{E}(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau)| + |\mathcal{E}^\tau(f^\tau) - \inf_{f\in\mathcal{H}} \mathcal{E}(f)| \tag{39}$$

where $\mathcal{E}^\tau(f) = \mathcal{E}(f) + 2\tau\Omega_1^{\mathrm{D}}(f) + \tau\nu\|f\|_{\mathcal{H}}^2$.

We first consider the sample error. Toward this end, we note that

$$\tau\nu\|\hat{f}^\tau\|_{\mathcal{H}}^2 \leq \widehat{\mathcal{E}}^\tau(\hat{f}^\tau) \leq \widehat{\mathcal{E}}^\tau(0) = \|\mathbf{y}\|_n^2 \Longrightarrow \|\hat{f}^\tau\|_{\mathcal{H}} \leq \frac{\|\mathbf{y}\|_n}{\sqrt{\tau\nu}} \leq \frac{M}{\sqrt{\tau\nu}},$$

and similarly $\|f^\tau\|_{\mathcal{H}} \leq (\int_X y^2 d\rho)^{1/2}/\sqrt{\tau\nu} \leq \frac{M}{\sqrt{\tau\nu}}$.

We have the following bound,

$$\begin{aligned}
\mathcal{E}(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau) &\leq (\mathcal{E}(\hat{f}^\tau) - \widehat{\mathcal{E}}(\hat{f}^\tau)) + \widehat{\mathcal{E}}(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau) \\
&\leq (\mathcal{E}(\hat{f}^\tau) - \widehat{\mathcal{E}}(\hat{f}^\tau)) + \widehat{\mathcal{E}}^\tau(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau) \\
&\leq (\mathcal{E}(\hat{f}^\tau) - \widehat{\mathcal{E}}(\hat{f}^\tau)) + \widehat{\mathcal{E}}^\tau(f^\tau) - \mathcal{E}^\tau(f^\tau) \\
&\leq (\mathcal{E}(\hat{f}^\tau) - \widehat{\mathcal{E}}(\hat{f}^\tau)) + (\widehat{\mathcal{E}}(f^\tau) - \mathcal{E}(f^\tau)) + \tau(\widehat{\Omega}_1^D(f^\tau) - \Omega_1^{\mathrm{D}}(f^\tau)) \\
&\leq 2 \sup_{\|f\|_{\mathcal{H}} \leq \frac{M}{\sqrt{\tau\nu}}} |\widehat{\mathcal{E}}(f) - \mathcal{E}(f)| + \tau \sup_{\|f\|_{\mathcal{H}} \leq \frac{M}{\sqrt{\tau\nu}}} |\widehat{\Omega}_1^D(f) - \Omega_1^{\mathrm{D}}(f)|.
\end{aligned}$$

Let $\eta' \in (0,1]$. Using Lemma 19 with probability $\eta = 3\eta'/(3+d)$, and inequality (38) with $\eta = d\eta'/(3+d)$, and if $\eta'$ is sufficiently small we obtain

$$\mathcal{E}(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau) \leq \frac{4\sqrt{2}}{\sqrt{n}} M^2 \left( \frac{\kappa_1^2}{\tau\nu} + \frac{2\kappa_1}{\sqrt{\tau\nu}} + 1 \right) \log\frac{6+2d}{\eta'} + \tau \frac{2\sqrt{2}}{(n)^{1/4}} d \frac{M}{\sqrt{\tau\nu}} \kappa_2 \sqrt{\log\frac{6+2d}{\eta'}}.$$

with probability $1-\eta'$. Furthermore, we have the bound

$$\mathcal{E}(\hat{f}^\tau) - \mathcal{E}^\tau(f^\tau) \leq c \left( \frac{M\kappa_1^2}{n^{1/2}\tau\nu} + \frac{\tau^{1/2} d\kappa_2}{n^{1/4}\sqrt{\nu}} \right) \log\frac{6+2d}{\eta'} \tag{40}$$

where $c$ does not depend on $n, \tau, \nu, d$. The proof follows, if we plug (40) in (39) and take $\tau = \tau_n$ such that $\tau_n \to 0$ and $(\tau_n \sqrt{n})^{-1} \to 0$, since the approximation error goes to zero (using Proposition 20) and the sample error goes to zero in probability as $n \to \infty$ by (40).

■

We next consider convergence in the RKHS norm. The following result on the convergence of the approximation error is standard (Dontchev and Zolezzi, 1993).

**Proposition 21** *Let $\tau_n \to 0$, be a positive sequence. Then we have that*

$$\|f_\rho^\dagger - f^{\tau_n}\|_{\mathcal{H}} \to 0.$$

We can now prove Theorem 9. The main difficulty is to control the sample error in the $\mathcal{H}$-norm. This requires showing that controlling the distance between the minima of two functionals, we can control the distance between their minimizers. Towards this end it is critical to use the results in Villa et al. (2012) based on Attouch-Wets convergence. We need to recall some useful quantities. Given two subsets $A$ and $B$ in a metric space $(\mathcal{H}, d)$, the excess of $A$ on $B$ is defined as $e(A,B) := \sup_{f \in A} d(f,B)$, with the convention that $e(\emptyset, B) = 0$ for every $B$. Localizing the definition of the excess we get the quantity $e_r(A,B) : e(A \cap B(0,r), B)$ for each ball $B(0,r)$ of radius $r$ centered at the origin. The *r-epi-distance* between two subsets $A$ and $B$ of $\mathcal{H}$, is denoted by $d_r(A,B)$ and is defined as

$$d_r(A,B) := \max\{e_r(A,B), e_r(B,A)\}.$$

The notion of epi-distance can be extended to any two functionals $F, G : \mathcal{H} \to \mathbb{R}$ by

$$d_r(G,F) := d_r(\mathrm{epi}(G), \mathrm{epi}(F)),$$

where for any $F : \mathcal{H} \to \mathbb{R}$, epi(F) denotes the epigraph of $F$ defined as

$$\mathrm{epi}(F) := \{(f, \alpha), F(f) \leq \alpha\}.$$

We are now ready to prove Theorem 9, which we present here in an extended form.

**Theorem 9 (Extended)** *Under assumptions (A1), (A2) and (A3),*

$$\mathrm{P}\Big( \|\hat{f}^\tau - f_\rho^\dagger\|_{\mathcal{H}} \geq A(n,\tau)^{1/2} + \|f^\tau - f_\rho^\dagger\|_{\mathcal{H}}, \Big) < \eta$$

*where*

$$A(n,\tau) = 4\sqrt{2} M \left( \frac{4\kappa_1^2 M}{\sqrt{n}\tau^2\nu^2} + \frac{4\kappa_1}{\sqrt{n}\tau\nu\sqrt{\tau\nu}} + \frac{1}{\sqrt{n}\tau\nu} + \frac{2d\kappa_2}{n^{1/4}\nu\sqrt{\tau\nu}} \right)$$

*for $0 < \eta \leq 1$. Moreover,*

$$\lim_{n\to\infty} \mathrm{P}\Big( \|\hat{f}^{\tau_n} - f_\rho^\dagger\|_{\mathcal{H}} \geq \varepsilon \Big) = 0, \qquad \forall \varepsilon > 0,$$

*for any $\tau_n$ such that $\tau_n \to 0$ and $(\sqrt{n}\tau_n^2)^{-1} \to 0$.*

**Proof** [Proof of Theorem 9]

We consider the decomposition of $\|\hat{f}^\tau - f_\rho^\dagger\|_{\mathcal{H}}$ into a sample and approximation term,

$$\|\hat{f}^\tau - f_\rho^\dagger\|_{\mathcal{H}} \leq \|\hat{f}^\tau - f^\tau\|_{\mathcal{H}} + \|f^\tau - f_\rho^\dagger\|_{\mathcal{H}}. \tag{41}$$

From Theorem 2.6 in Villa et al. (2012) we have that

$$\psi^{\diamond}_{\tau\nu}(\|\hat{f}^{\tau} - f^{\tau}\|_{\mathcal{H}}) \leq 4 d_{M/\sqrt{\tau\nu}}(t_{\mathcal{E}^{\tau}}\mathcal{E}^{\tau}, t_{\mathcal{E}^{\tau}}\widehat{\mathcal{E}}^{\tau})$$

where $\psi^{\diamond}_{\tau\nu}(t) := \inf\{\frac{\tau\nu}{2}s^2 + |t-s| : s \in [0,+\infty)\}$, and $t_{\mathcal{E}^{\tau}}$ is the translation map defined as

$$t_{\mathcal{E}^{\tau}} G(f) = G(f + f^{\tau}) - \mathcal{E}^{\tau}(f^{\tau})$$

for all $G : \mathcal{H} \to \mathbb{R}$.

From Theorem 2.7 in Villa et al. (2012), we have that

$$d_{M/\sqrt{\tau\nu}}(t_{\mathcal{E}^{\tau}}\mathcal{E}^{\tau}, t_{\mathcal{E}^{\tau}}\widehat{\mathcal{E}}^{\tau}) \leq \sup_{\|f\|_{\mathcal{H}} \leq M/\sqrt{\tau\nu}} |t_{\mathcal{E}^{\tau}}\mathcal{E}^{\tau}(f) - t_{\mathcal{E}^{\tau}}\widehat{\mathcal{E}}^{\tau}(f)|.$$

We have the bound,

$$\begin{aligned}
\sup_{\|f\|_{\mathcal{H}} \leq M/\sqrt{\tau\nu}} |t_{\mathcal{E}^{\tau}}\mathcal{E}^{\tau}(f) - t_{\mathcal{E}^{\tau}}\widehat{\mathcal{E}}^{\tau}(f)| &\leq \sup_{\|f\|_{\mathcal{H}} \leq M/\sqrt{\tau\nu} + \|f^{\tau}\|_{\mathcal{H}}} |\mathcal{E}^{\tau}(f) - \widehat{\mathcal{E}}^{\tau}(f)| \\
&\leq \sup_{\|f\|_{\mathcal{H}} \leq 2M/\sqrt{\tau\nu}} |\mathcal{E}(f) - \widehat{\mathcal{E}}(f)| + \tau \sup_{\|f\|_{\mathcal{H}} \leq 2M/\sqrt{\tau\nu}} |\Omega_1^{\mathrm{D}}(f) - \widehat{\Omega}_1^{D}(f)|.
\end{aligned}$$

Using Theorem 7 (equation (38)) and Lemma 19 we obtain with probability $1-\eta'$, if $\eta'$ is small enough,

$$\begin{aligned}
d_{M/\sqrt{\tau\nu}}(t_{\mathcal{E}^{\tau}}\mathcal{E}^{\tau}, t_{\mathcal{E}^{\tau}}\widehat{\mathcal{E}}^{\tau}) &\leq \frac{2\sqrt{2}}{\sqrt{n}} \left( \kappa_1^2 \frac{4M^2}{\tau\nu} + 4\kappa_1 \frac{M^2}{\sqrt{\tau\nu}} + M^2 \right) \log \frac{6+2d}{\eta'} + \tau \frac{2M}{\sqrt{\tau\nu}} d \frac{2\sqrt{2}}{n^{1/4}} \kappa_2 \sqrt{\log \frac{6+2d}{\eta'}} \\
&\leq 2\sqrt{2}M \left( \frac{4\kappa_1^2 M}{\sqrt{n}\tau\nu} + \frac{4\kappa_1 M}{\sqrt{n}\sqrt{\tau\nu}} + \frac{M}{\sqrt{n}} + \tau \frac{2d\kappa_2}{n^{1/4}\sqrt{\tau\nu}} \right) \log \frac{6+2d}{\eta'}. \qquad (42)
\end{aligned}$$

From the definition of $\psi^{\diamond}_{\tau\nu}$ it is possible to see that we can write explicitly $(\psi^{\diamond}_{\tau\nu})^{-1}$ as

$$(\psi^{\diamond}_{\tau\nu})^{-1}(y) = \begin{cases} \sqrt{\frac{2y}{\tau\nu}} & \text{if } y < \frac{1}{2\tau\nu} \\ y + \frac{1}{2\tau} & \text{otherwise.} \end{cases}$$

Since $\tau = \tau_n \to 0$ by assumption, for sufficiently large $n$, the bound in (42) is smaller than $1/2\tau\nu$, and we obtain that with probability $1-\eta'$,

$$\|\hat{f}^{\tau} - f^{\tau}\|_{\mathcal{H}} \leq \left( 4\sqrt{2}M \left( \frac{4\kappa_1^2 M}{\sqrt{n}\tau^2\nu^2} + \frac{4\kappa_1}{\sqrt{n}\tau\nu\sqrt{\tau\nu}} + \frac{1}{\sqrt{n}\tau\nu} + \frac{2d\kappa_2}{n^{1/4}\nu\sqrt{\tau\nu}} \right) \right)^{1/2} \sqrt{\log \frac{6+2d}{\eta'}}. \qquad (43)$$

If we now plug (43) in (41) we obtain the first part of the proof. The rest of the proof follows by taking the limit $n \to \infty$, and by observing that, if one chooses $\tau = \tau_n$ such that $\tau_n \to 0$ and $(\tau_n^2\sqrt{n})^{-1} \to 0$, the assumption of Proposition 21 is satisfied and the bound in (43) goes to 0, so that the limit of the sum of the sample and approximation terms goes to 0.

■

### C.1.2 PROOFS OF THE SELECTION PROPERTIES

In order to prove our main selection result, we will need the following lemma.

**Lemma 22** *Under assumptions (A1), (A2) and (A3) and defining $A(n,\tau)$ as in Theorem 9 extended, we have, for all $a = 1,\dots,d$ and for all $\varepsilon > 0$,*

$$\mathrm{P}\Big(\Big|\|\hat{D}_a\hat{f}^\tau\|_n^2 - \|D_a f_\rho^\dagger\|_{\rho_X}^2\Big| \geq \varepsilon\Big) < (6+2d)\exp\left(-\frac{\varepsilon - b(\tau)}{a(n,\tau)}\right),$$

*where $a(n,\tau) = 2\max\{\frac{2\sqrt{2}M^2\kappa_2^2}{\sqrt{n}\tau\nu}, 2\kappa_2^2 A(n,\tau)\}$ and $\lim_{\tau\to 0} b(\tau) = 0$.*

**Proof** We have the following set of inequalities

$$\begin{aligned}
\Big|\|\hat{D}_a\hat{f}^\tau\|_n^2 - \|D_a f_\rho^\dagger\|_{\rho_X}^2\Big| &= |\langle \hat{f}^\tau, \hat{D}_a^*\hat{D}_a\hat{f}^\tau\rangle_{\mathcal{H}} - \langle f_\rho^\dagger, D_a^*D_a f_\rho^\dagger\rangle_{\mathcal{H}} + \\
&\quad \langle \hat{f}^\tau, D_a^*D_a\hat{f}^\tau\rangle_{\mathcal{H}} - \langle \hat{f}^\tau, D_a^*D_a\hat{f}^\tau\rangle_{\mathcal{H}} + \\
&\quad \langle f_\rho^\dagger, D_a^*D_a\hat{f}^\tau\rangle_{\mathcal{H}} - \langle f_\rho^\dagger, D_a^*D_a\hat{f}^\tau\rangle_{\mathcal{H}}| \\
&= \Big|\langle \hat{f}^\tau, (\hat{D}_a^*\hat{D}_a - D_a^*D_a)\hat{f}^\tau\rangle_{\mathcal{H}} + \langle \hat{f}^\tau - f_\rho^\dagger, D_a^*D_a(\hat{f}^\tau - f_\rho^\dagger)\rangle_{\mathcal{H}}\Big| \\
&\leq \|\hat{D}_a^*\hat{D}_a - D_a^*D_a\|\frac{M^2}{\tau\nu} + \kappa_2^2\|\hat{f}^\tau - f_\rho^\dagger\|_{\mathcal{H}}^2 \\
&\leq \|\hat{D}_a^*\hat{D}_a - D_a^*D_a\|\frac{M^2}{\tau\nu} + 2\kappa_2^2\|\hat{f}^\tau - f^\tau\|_{\mathcal{H}}^2 + 2\kappa_2^2\|f^\tau - f_\rho^\dagger\|_{\mathcal{H}}^2.
\end{aligned}$$

Using Theorem 9 extended, equation (43), and Lemma 18 with probability $\eta_4 = \eta/(3+d)$, we obtain with probability $1-\eta$

$$\Big|\|\hat{D}_a\hat{f}^\tau\|_n^2 - \|D_a f_\rho^\dagger\|_{\rho_X}^2\Big| \leq \frac{2\sqrt{2}M^2\kappa_2^2}{\sqrt{n}\tau\nu}\log\frac{6+2d}{\eta} + 2\kappa_2^2 A(n,\tau)\log\frac{6+2d}{\eta} + 2\kappa_2^2\|f^\tau - f_\rho^\dagger\|_{\mathcal{H}}^2.$$

We can further write

$$\Big|\|\hat{D}_a\hat{f}^\tau\|_n^2 - \|D_a f_\rho^\dagger\|_{\rho_X}^2\Big| \leq a(n,\tau)\log\frac{6+2d}{\eta} + b(\tau),$$

where $a(n,\tau) = 2\max\{\frac{2\sqrt{2}M^2\kappa_2^2}{\sqrt{n}\tau\nu}, 2\kappa_2^2 A(n,\tau)\}$ and $\lim_{\tau\to 0} b(\tau) = 0$ according to Proposition 21. The proof follows by writing $\varepsilon = a(n,\tau)\log\frac{6+2d}{\eta} + b(\tau)$ and inverting it with respect to $\eta$.

■

Finally we can prove Theorem 11.

**Proof** [Proof of Theorem 11] We have

$$\mathrm{P}\big(R_\rho \subseteq \hat{R}^\tau\big) = 1 - \mathrm{P}\big(R_\rho \not\subseteq \hat{R}^\tau\big) = 1 - \mathrm{P}\left(\bigcup_{a\in R_\rho}\{a \notin \hat{R}^\tau\}\right) \geq 1 - \sum_{a\in R_\rho}\mathrm{P}\big(a \notin \hat{R}^\tau\big)$$

Let us now estimate $\mathrm{P}\big(a \notin \hat{R}^\tau\big)$ or equivalently $\mathrm{P}\big(a \in \hat{R}^\tau\big) = \mathrm{P}\big(\|\hat{D}_a\hat{f}^\tau\|_n^2 > 0\big)$, for $a \in R_\rho$. Let $C < \min_{a\in R_\rho}\|D_a f_\rho^\dagger\|_{\rho_X}^2$. From Lemma 22, there exist $a(n,\tau)$ and $b(\tau)$ satisfying $\lim_{\tau\to 0} b(\tau) = 0$, such that

$$\Big|\|D_a f_\rho^\dagger\|_{\rho_X}^2 - \|\hat{D}_a\hat{f}^\tau\|_n^2\Big| \leq \varepsilon$$

with probability $1-(6+2d)\exp\left(-\frac{\varepsilon-b(\tau)}{a(n,\tau)}\right)$, for all $a=1,\ldots,d$. Therefore, for $\varepsilon=C$, for $a\in R_\rho$, it holds

$$\|\hat{D}_a\hat{f}^\tau\|_n^2 \geq \|D_a f_\rho^\dagger\|_{\rho_X}^2 - C > 0.$$

with probability $1-(6+2d)\exp\left(-\frac{C-b(\tau)}{a(n,\tau)}\right)$. We than have

$$\mathrm{P}\big(a\in\hat{R}^\tau\big) = \mathrm{P}\big(\|\hat{D}_a\hat{f}^\tau\|_n^2>0\big) \geq 1-(6+2d)\exp\left(-\frac{C-b(\tau)}{a(n,\tau)}\right),$$

so that $\mathrm{P}\big(a\notin\hat{R}^\tau\big)\leq(6+2d)\exp\left(-\frac{C-b(\tau)}{a(n,\tau)}\right)$. Finally, if we let $\tau=\tau_n$ satisfying the assumption, we have $\lim_n b(\tau_n)\to 0$, $\lim_n a(n,\tau_n)\to 0$, so that

$$\begin{aligned}\lim_{n\to\infty}\mathrm{P}\big(R_\rho\subseteq\hat{R}^{\tau_n}\big) &\geq \lim_{n\to\infty}\left[1-|R_\rho|(6+2d)\exp\left(-\frac{C-c(\tau_n)}{a(n,\tau_n)}\right)\right]\\ &= 1-|R_\rho|(6+2d)\lim_{n\to\infty}\exp\left(-\frac{C-b(\tau_n)}{a(n,\tau_n)}\right)\\ &= 1.\end{aligned}$$

■

## References


N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

H. Attouch and R. Wets. Quantitative stability of variational systems. I. The epigraphical distance. *Transactions of the American Mathematical Society*, 328(2):695–729, 1991.

H. Attouch and R. Wets. Quantitative stability of variational systems. II. A framework for nonlinear conditioning. *SIAM Journal on Optimization*, 3(2):359–381, 1993a.

H. Attouch and R. Wets. Quantitative stability of variational systems. III. ε-approximate solutions. *Mathematical Programming*, 61(2, Ser. A):197–214, 1993b.

F. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

F. Bach. High-dimensional non-linear variable selection through hierarchical kernel learning. Technical Report HAL 00413473, INRIA, 2009.

F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the 21st International Machine Learning Conference*, Banff, Alberta, Canada, 2004.

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.

| | | |
|---|---|---|
| Spaces and distributions | $\mathcal{X} \subseteq \mathbb{R}^d$ | input space |
| | $\mathcal{Y} \subseteq \mathbb{R}$ | output space |
| | $\rho$ | probability distribution on $\mathcal{X} \times \mathcal{Y}$ |
| | $\rho_{\mathcal{X}}$ | marginal distribution of $\rho$ |
| | $L^2(\mathcal{X}, \rho_{\mathcal{X}})$ | $\{f : \mathcal{X} \to \mathbb{R} : \text{measurable and s.t. } \int_{\mathcal{X}} f(x)^2 \, d\rho_{\mathcal{X}}(x) < +\infty\}$ |
| | $\mathcal{H}$ | RKHS $\subseteq \{f : \mathcal{X} \to \mathcal{Y}\}$ |
| Norms and scalar products | $\|\cdot\|_n$ and $\langle \cdot, \cdot \rangle_n$ | $\frac{1}{\sqrt{n}} \cdot$ euclidean norm and scalar product |
| | $\|\cdot\|_{\rho_{\mathcal{X}}}$ and $\langle \cdot, \cdot \rangle_{\rho_{\mathcal{X}}}$ | norm and scalar product in $L^2(\mathcal{X}, \rho_{\mathcal{X}})$ |
| | $\|\cdot\|_{\mathcal{H}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ | norm and scalar product in $\mathcal{H}$ |
| Functionals and Operators | $\Omega_1^{\mathrm{D}} : \mathcal{H} \to [0, +\infty)$ | $\Omega_1^{\mathrm{D}}(f) = \sum_{a=1}^{d} \sqrt{\int_{\mathcal{X}} \left( \frac{\partial f(x)}{\partial x^a} \, d\rho_{\mathcal{X}}(x) \right)^2}$ |
| | $\widehat{\Omega}_1^{D} : \mathcal{H} \to [0, +\infty)$ | $\widehat{\Omega}_1^{D}(f) = \sum_{a=1}^{d} \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left( \frac{\partial f(x_i)}{\partial x^a} \right)^2}$ |
| | $\mathcal{E} : \mathcal{H} \to [0, +\infty)$ | $\mathcal{E}(f) = \int_{\mathcal{X}} (f(x) - y)^2 \, d\rho(x, y)$ |
| | $\mathcal{E}^{\tau} : \mathcal{H} \to [0, +\infty)$ | $\mathcal{E}^{\tau}(f) = \int_{\mathcal{X}} (f(x) - y)^2 \, d\rho(x, y) + \tau(2\Omega_1^{\mathrm{D}}(f) + \nu \|f\|_{\mathcal{H}}^2)$ |
| | $\hat{\mathcal{E}} : \mathcal{H} \to [0, +\infty)$ | $\hat{\mathcal{E}}(f) = \sum_{i=1}^{n} \frac{1}{n} (f(x_i) - y_i)^2$ |
| | $\hat{\mathcal{E}}^{\tau} : \mathcal{H} \to [0, +\infty)$ | $\hat{\mathcal{E}}^{\tau}(f) = \sum_{i=1}^{n} \frac{1}{n} (f(x_i) - y_i)^2 + \tau(2\widehat{\Omega}_1^{D}(f) + \nu \|f\|_{\mathcal{H}}^2)$ |
| | $I_k : \mathcal{H} \to L^2(\mathcal{X}, \rho_{\mathcal{X}})$ | $(I_k f)(x) = \langle f, k_x \rangle_{\mathcal{H}}$ |
| | $\hat{S} : \mathcal{H} \to \mathbb{R}^n$ | $\hat{S} f = (f(x_1), \ldots, f(x_n))$ |
| | $D_a : \mathcal{H} \to L^2(\mathcal{X}, \rho_{\mathcal{X}})$ | $(D_a f)(x) = \langle f, (\partial_a k)_x \rangle$ |
| | $\hat{D}_a : \mathcal{H} \to \mathbb{R}^n$ | $\hat{D}_a(f) = \left( \frac{\partial f}{\partial x^a}(x_1), \ldots, \frac{\partial f}{\partial x^a}(x_n) \right)$ |
| | $\nabla : \mathcal{H} \to (L^2(\mathcal{X}, \rho_{\mathcal{X}}))^d$ | $\nabla f = (D_a f)_{a=1}^d$ |
| | $\hat{\nabla} : \mathcal{H} \to (\mathbb{R}^n)^d$ | $\hat{\nabla} f = (\hat{D}_a f)_{a=1}^d$ |
| Functions | $k_x : \mathcal{X} \to \mathbb{R}$ | $t \mapsto k(x, t)$ |
| | $f_{\rho}^{\dagger}$ | $\operatorname{argmin}_{f \in \operatorname{argmin} \mathcal{E}} \{\Omega_1^D(f) + \nu \|f\|_{\mathcal{H}}^2\}$ |
| | $(\partial_a k)_x : \mathcal{X} \to \mathbb{R}$ | $t \mapsto \left. \frac{\partial k(s,t)}{\partial s^a} \right\|_{s=x}$ |
| | $f^{\tau}$ | the minimizer in $\mathcal{H}$ of $\mathcal{E}^{\tau}$ |
| | $\hat{f}^{\tau}$ | the minimizer in $\mathcal{H}$ of $\hat{\mathcal{E}}^{\tau}$ |
| Sets | $R_{\rho}$ | $\{a \in \{1, \ldots d\} : \frac{\partial f_{\rho}^{\dagger}}{\partial x^a} \neq 0\}$ |
| | $\hat{R}^{\tau}$ | $\{a \in \{1, \ldots d\} : \frac{\partial f^{\tau}}{\partial x^a} \neq 0\}$ |
| | $B_n$ | $\{v \in \mathbb{R}^n : \|v\|_n \leq 1\}$ |
| | $B_n^d$ | $\{v \in \mathbb{R}^n : \|v\|_n \leq 1\}^d$ |

Table 3: List of symbols and notations

S. Becker, J. Bobin, and E. Candès. NESTA: a fast and accurate first-order method for sparse recovery. *SIAM Journal on Imaging Sciences*, 4(1):1–39, 2011.

J. Bect, L. Blanc-Féraud, G. Aubert, and A. Chambolle. A $\ell^1$-unified variational framework for image restoration. In *Proceedings of the 8th European Conference on Computer Vision, Part IV*, pages 1–13, Prague, Czech Republic, 2004.

M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2008.

K. Bertin and G. Lecué. Selection of variables and dimension reduction in high-dimensional non-parametric regression. *Electronic Journal of Statistics*, 2:1224–1241, 2008.

P. Bühlmann and S. van de Geer. *Statistics for High-Dimensional Data*. Springer, Berlin, 2011.

J. Candès, E.and Romberg and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

T. Chan, G. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM Journal on Scientific Computing*, 20, 1999.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

P. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200 (electronic), 2005.

P. L. Combettes, D. Dũng, and B. C. Vũ. Dualization of signal recovery problems. *Set-Valued and Variational Analysis*, 18(3-4):373–404, 2010.

L. Comminges and A. Dalalyan. Tight conditions for consistency of variable selection in the context of high dimensionality. *Annals of Statistics*, 40(5):2667–2696, 2012.

I. Daubechies, G. Teschke, and L. Vese. Iteratively solving linear inverse problems under general convex constraints. *Inverse Problems and Imaging*, 1(1):29–46, 2007.

E. De Vito, L. Rosasco, A. Caponnetto, U. De Giovannini, and F. Odone. Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6:883–904, 2005.

R. DeVore, G. Petrova, and P. Wojtaszczyk. Approximation of functions of few variables in high dimensions. *Constructive Approximation. An International Journal for Approximations and Expansions*, 33(1):125–143, 2011.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, New York, 1996.

D. Donoho. Compressed sensing. *IEEE. Transactions on Information Theory*, 52(4):1289–1306, 2006.

A. L. Dontchev and T. Zolezzi. *Well-posed Optimization Problems*. Springer-Verlag, Berlin, 1993.

J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, December 2009.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32: 407–499, 2004.

I. Ekeland and R. Temam. *Convex Analysis and Variational Problems*. North-Holland Publishing Co., Amsterdam, 1976.

M. Figueiredo, R. Nowak, and S. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):586–597, 2007.

C. Gu. *Smoothing Spline ANOVA Models*. Springer, New York, 2002.

E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for l1-minimization: Methodology and convergence. *SIAM Journal of Optimization*, 19(3):1107–1130, 2008.

T. Hastie and R. Tibshirani. *Generalized Additive Models*. Chapman and Hall, London, 1990.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms: Part I: Fundamentals*. Springer, Berlin, 1993.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

V. Koltchinskii and M. Yuan. Sparsity in multiple kernel learning. *Annals of Statistics*, 38(6): 3660–3695, 2010.

J. Lafferty and L. Wasserman. Rodeo: sparse, greedy nonparametric regression. *Annals of Statistics*, 36(1):28–63, 2008.

Y. Lin and H. H. Zhang. Component selection and smoothing in multivariate nonparametric regression. *Annals of Statistics*, 34:2272, 2006.

I. Loris. On the performance of algorithms for the minimization of $l_1$-penalized functionals. *Inverse Problems*, 25(3):035008, 16, 2009.

I. Loris, M. Bertero, C. De Mol, R. Zanella, and L. Zanni. Accelerating gradient projection methods for $l_1$-constrained signal recovery by steplength selection rules. *Applied and Computational Harmonic Analysis*, 27(2):247–254, 2009.

A. Maurer and M. Pontil. Structured sparsity and generalization. *Journal Machine Learning Research*, 13:671–690, 2012.

H. Miller and P. Hall. Local polynomial regression and variable selection. In J.O. Berger, T.T. Cai, and I.M. Johnstone, editors, *Borrowing Strength: Theory Powering Applications—a Festschrift for Lawrence D. Brown*, volume 6, pages 216–233. Institute of Mathematical Statistics, 2010.

J.-J. Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la Société Mathématique de France*, 93:273–299, 1965.

S. Mosci, L. Rosasco, M. Santoro, A. Verri, and S. Villa. Solving structured sparsity regularization with proximal methods. In J.L. Balcàzar, F. Bonchi, A. Gionis, and M. Sebag, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 6322 of *LNCS*, pages 418–433. Springer Berlin Heidelberg, 2010.

A.S. Nemirovski and D.B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. John Wiley & Sons, New York, 1983.

Y. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. *Doklady AN SSSR*, 269(3):543–547, 1983.

Y. Nesterov. Gradient methods for minimizing composite objective function. CORE Discussion Paper 2007/76, Catholic University of Louvain, September 2007.

I. F. Pinelis and A. I. Sakhanenko. Remarks on inequalities for probabilities of large deviations. *Theory of Probability and its Applications*, 30(1):143–148, 1985.

P. Ravikumar, H. Liu, J. Lafferty, and L. Wasserman. Spam: Sparse additive models. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*. NIPS Foundation, 2008.

L. Rosasco, S. Mosci, M. S. Santoro, A. Verri, and S. Villa. A regularization approach to nonlinear variable selection. In *Proceedings of the 13 International Conference on Artificial Intelligence and Statistics*, 2010.

S. Salzo and S. Villa. Inexact and accelerated proximal point algorithms. *Journal of Convex Analysis*, 19(4), 2012.

M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 286–297, 2007.

I. Steinwart and A. Christmann. *Support Vector Machines*. Information Science and Statistics. Springer, New York, 2008.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

J. Tropp and A. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53:4655–4666, 2007.

P. Tseng. Approximation accuracy, gradient methods, and error bound for structured convex optimization. *Mathematical Programming. A Publication of the Mathematical Programming Society*, 125(2, Ser. B):263–295, 2010.

S. Villa, S. Salzo, L. Baldassarre, and A. Verri. *SIOPT*.

S. Villa, L. Rosasco, S. Mosci, and A. Verri. Consistency of learning algorithms using Attouch-Wets convergence. *Optimization*, 61(3):287–305, 2012.

G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), 1990.

G. Wahba, Y. Wang, C. Gu, R. Klein, and B. Klein. Smoothing spline ANOVA for exponential families, with application to the Wisconsin epidemiological study of diabetic retinopathy. *Annals of Statistics*, 23:1865–1895, 1995.

D.-X. Zhou. Derivative reproducing properties for kernel methods in learning theory. *Journal of Computational and Applied Mathematics*, 220:456–463, 2008.

Journal of Machine Learning Research 14 (2013) 1715-1746 Submitted 1/12; Revised 11/12; Published 7/13

# Similarity-based Clustering by Left-Stochastic Matrix Factorization

**Raman Arora** ARORA@TTIC.EDU
*Toyota Technological Institute*
*6045 S. Kenwood Ave*
*Chicago, IL 60637, USA*

**Maya R. Gupta** MAYAGUPTA@GOOGLE.COM
*Google*
*1225 Charleston Rd*
*Mountain View, CA 94301, USA*

**Amol Kapila** AKAPILA@U.WASHINGTON.EDU
**Maryam Fazel** MFAZEL@U.WASHINGTON.EDU
*Department of Electrical Engineering*
*University of Washington*
*Seattle, WA 98195, USA*

**Editor:** Inderjit Dhillon

## Abstract

For similarity-based clustering, we propose modeling the entries of a given similarity matrix as the inner products of the unknown cluster probabilities. To estimate the cluster probabilities from the given similarity matrix, we introduce a left-stochastic non-negative matrix factorization problem. A rotation-based algorithm is proposed for the matrix factorization. Conditions for unique matrix factorizations and clusterings are given, and an error bound is provided. The algorithm is particularly efficient for the case of two clusters, which motivates a hierarchical variant for cases where the number of desired clusters is large. Experiments show that the proposed left-stochastic decomposition clustering model produces relatively high within-cluster similarity on most data sets and can match given class labels, and that the efficient hierarchical variant performs surprisingly well.

**Keywords:** clustering, non-negative matrix factorization, rotation, indefinite kernel, similarity, completely positive

## 1. Introduction

Clustering is important in a broad range of applications, from segmenting customers for more effective advertising, to building codebooks for data compression. Many clustering methods can be interpreted in terms of a matrix factorization problem. For example, the popular $k$-means clustering algorithm attempts to solve the *k-means problem*: produce a clustering such that the sum of squared error between samples and the mean of their cluster is small (Hastie et al., 2009). For $n$ feature vectors gathered as the $d$-dimensional columns of a matrix $X \in \mathbb{R}^{d\times n}$, the $k$-means problem can be written as a matrix factorization:

$$\begin{aligned} \underset{\substack{F\in\mathbb{R}^{d\times k}\\ G\in\mathbb{R}^{k\times n}}}{\text{minimize}} \quad & \|X-FG\|_F^2 \\ \text{subject to} \quad & G\in\{0,1\}^{k\times n}, G^T\mathbf{1}_k=\mathbf{1}_n, \end{aligned} \tag{1}$$

©2013 Raman Arora, Maya R. Gupta, Amol Kapila and Maryam Fazel.

where $\|\cdot\|_F$ is the Frobenius norm and $\mathbf{1}_n$ is an $n \times 1$ vector of 1's. The matrix $F$ can be interpreted as a matrix whose columns are the $k$ cluster centroids. The combined constraints $G \in \{0,1\}^{k\times n}$ and $G^T\mathbf{1}_k = \mathbf{1}_n$ force each column of $G$ to contain all zeros except for one element, which is a 1, whose location corresponds to the cluster assignment. That is, $G_{ij} = 1$ if sample $j$ belongs in cluster $i$, and $G_{ij} = 0$ otherwise. The $k$-means clustering problem is not convex. The $k$-means algorithm is traditionally used to seek a local optimum to (1) (Hastie et al., 2009; Selim and Ismail, 1984).

The *soft* $k$-means problem instead solves for a *cluster probability matrix* $P$ that specifies the probability of each sample belonging to each of the clusters:

$$\begin{aligned} \underset{\substack{F\in\mathbb{R}^{d\times k}\\ P\in\mathbb{R}^{k\times n}}}{\text{minimize}} \quad & \|X - FP\|_F^2 \\ \text{subject to} \quad & P \geq 0, P^T\mathbf{1}_k = \mathbf{1}_n, \end{aligned} \tag{2}$$

where the inequality $P \geq 0$ is entrywise. The constraints $P \geq 0$ and $P^T\mathbf{1}_k = \mathbf{1}_n$ together force each column of $P$ to be non-negative and to sum to 1, making $P$ left-stochastic. Hence, each column of $P$ is a probability mass function encoding probabilistic assignments of points to clusters: we interpret $P_{ij}$ to express the probability that sample $j$ belongs in cluster $i$.

Interpreting non-negative factors of matrices as describing a data clustering was proposed by Paatero and Tapper (1994) and Lee and Seung (1999). Following their work, other researchers have posed different matrix factorization models, and attempted to explicitly solve the resulting matrix factorization problems to form clusterings. Zha et al. (2001) proposed relaxing the constraints on $G$ in the $k$-means optimization problem to an orthogonality constraint:

$$\begin{aligned} \underset{\substack{F\in\mathbb{R}^{d\times k}\\ G\in\mathbb{R}^{k\times n}}}{\text{minimize}} \quad & \|X - FG\|_F^2 \\ \text{subject to} \quad & GG^T = I_k, G \geq 0, \end{aligned}$$

where $I_k$ is a $k \times k$ identity matrix. Ding et al. (2005) considered the kernelized clustering objective,

$$\begin{aligned} \underset{G\in\mathbb{R}^{k\times n}}{\text{minimize}} \quad & \|K - G^TG\|_F^2 \\ \text{subject to} \quad & GG^T = I_k, G \geq 0. \end{aligned}$$

Recently, Ding et al. (2010) considered changing the constraints on $G$ in the $k$-means problem given in (1) to only require that $G$ be positive. They explored a number of approximations to the $k$-means problem that imposed different constraints on $F$. One such variant that they deemed particularly worthy of further investigation was *convex NMF*. Convex NMF restricts the columns of $F$ (the cluster centroids) to be convex combinations of the columns of $X$:

$$\underset{\substack{W\in\mathbb{R}_+^{n\times k}\\ G\in\mathbb{R}_+^{k\times n}}}{\text{minimize}} \quad \|X - XWG\|_F^2.$$

Convex NMF is amenable to the kernel trick (Ding et al., 2010); for an input kernel matrix $K$, the kernel convex NMF solves

$$\underset{\substack{W\in\mathbb{R}_+^{n\times k}\\ G\in\mathbb{R}_+^{k\times n}}}{\text{minimize}} \quad \operatorname{tr}\left(K - 2G^TW + W^TKWG^TG\right).$$

In this paper, we propose a new non-negative matrix factorization (NNMF) model for clustering from pairwise similarities between the samples.[1] First, we introduce the proposed *left-stochastic decomposition* (LSD) in Section 1.1. Then in Section 2, we provide a theoretical foundation for LSD and motivate our rotation-based algorithm, which is given in Section 3. For $k > 2$ clusters, we show that there may be multiple matrix factorization solutions related by a rotation, and provide conditions for the uniqueness of the resulting clustering. For $k = 2$ clusters, our approach to solving the LSD problem provides a simple unique solution, which motivates a fast binary hierarchical LSD clustering, described in Section 3.3. Experimental results are presented in Section 5, and show that the LSD clustering model performs well in terms of both within-class similarity and misclassification rate. The paper concludes with a discussion and some notes on how the proposed rotation-based algorithmic approach may be useful in other contexts.

### 1.1 Left-Stochastic Matrix Decomposition

Cristianini et al. (2001) defined the *ideal kernel* $K^*$ to have $K^*_{ij} = 1$ if and only if the $i^{th}$ and $j^{th}$ sample are from the same class or cluster. We propose a clustering model that relaxes this notion of an ideal kernel. Suppose $P \in \{0,1\}^{k\times n}$ is a cluster assignment matrix with the $P_{ij} = 1$ if and only if the $j^{th}$ sample belongs to the $i^{th}$ cluster. Then $K^* = P^T P$ would be such an ideal kernel. Relax the ideal kernel's assumption that each sample belongs to only one class, and let $P \in [0,1]^{k\times n}$ be interpreted as a soft cluster assignment matrix for $n$ samples and $k$ clusters. We further constrain $P$ to be left-stochastic, so that each column sums to one, allowing $P_{ij}$ to be interpreted as the probability that sample $j$ belongs to cluster $i$. In practice, the range of the entries of a given matrix $K$ may preclude it from being well approximated by $P^T P$ for a left-stochastic matrix $P$. For example, if the elements of $K$ are very large, then it will be impossible to approximate them by the inner products of probability vectors. For this reason, we also include a scaling factor $c \in \mathbb{R}$, producing what we term the *left-stochastic decomposition* (LSD) model: $cK \approx P^T P$.

We use the LSD model for clustering by solving a non-negative matrix factorization problem for the best cluster assignment matrix $P$. That is, given a matrix $K$, we propose finding a scaling factor $c$ and a cluster probability matrix $P$ that best solve

$$\begin{aligned} \underset{\substack{c\in\mathbb{R}_+ \\ P\in[0,1]^{k\times n}}}{\text{minimize}} \quad & \|K - \frac{1}{c}P^T P\|_F^2 \\ \text{subject to} \quad & P^T \mathbf{1}_k = \mathbf{1}_n. \end{aligned} \tag{3}$$

*LSD Clustering:* Given a solution $\hat{P}$ to (3) an LSD clustering is an assignment of samples to clusters such that if the $j^{th}$ sample is assigned to cluster $i^*$, then $\hat{P}_{i^*j} \geq \hat{P}_{ij}$ for all $i$.

*Input $K$*: While the LSD problem stated in (3) does not in itself require any conditions on the input matrix $K$, we restrict our attention to symmetric input matrices whose top $k$ eigenvalues are positive, and refer to such matrices in this paper as *similarity matrices*. If a user's input matrix does not have $k$ positive eigenvalues, its spectrum can be easily modified (Chen et al., 2009a,b).

*LSDable: $K$* is *LSDable* if the minimum of (3) is zero, that is $cK = P^T P$ for some feasible $c$ and $P$.

1. A preliminary version of this paper appeared as Arora et al. (2011). This paper includes new theoretical results, detailed proofs, new algorithmic contributions, and more exhaustive experimental comparisons.

### 1.2 Equivalence of LSD Clustering and Soft $k$-Means

A key question for LSD clustering is, "How does it relate to the soft $k$-means problem?" We can show that for LSDable $K$, any solution to the LSD problem also solves the soft $k$-means problem (2):

**Proposition 1 (Soft $k$-means Equivalence)** *Suppose $K$ is LSDable. Consider any feature mapping $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ such that $K_{ij} = \phi(X_i)^T \phi(X_j)$, and let $\Phi(X) = [\phi(X_1), \phi(X_2), \ldots, \phi(X_n)] \in \mathbb{R}^{d' \times n}$. Then the minimizer $P^*$ of the LSD problem (3) also minimizes the following soft k-means problem:*

$$\begin{aligned} \underset{\substack{F \in \mathbb{R}^{d' \times k} \\ P \geq 0,}}{\textit{minimize}} \quad & \|\Phi(X) - FP\|_F^2 \\ \textit{subject to} \quad & P^T \mathbf{1}_k = \mathbf{1}_n. \end{aligned} \tag{4}$$

In practice, a given similarity matrix $K$ will not be LSDable, and it is an open question how the LSD solution and soft $k$-means solution will be related.

## 2. Theory

In this section we provide a theoretical foundation for LSD clustering and our rotation-based LSD algorithm. All of the proofs are in the appendix.

### 2.1 Decomposition of the LSD Objective Error

To analyze LSD and motivate our algorithm, we consider the error introduced by the various constraints on $P$ separately. Let $M \in \mathbb{R}^{k \times n}$ be any rank-$k$ matrix (this implies $M^T M$ has rank $k$). Let $Z \in \mathbb{R}^{k \times n}$ be any matrix such that $Z^T Z$ is LSDable. The LSD objective given in (3) can be re-written as:

$$\begin{aligned} \|K - \frac{1}{c} P^T P\|_F^2 &= \|K - M^T M + M^T M - Z^T Z + Z^T Z - \frac{1}{c} P^T P\|_F^2 \\ &= \|K - M^T M\|_F^2 + \|M^T M - Z^T Z\|_F^2 + \|Z^T Z - \frac{1}{c} P^T P\|_F^2 && (5) \\ &\quad + \text{trace}\left(2(K - M^T M)^T (Z^T Z - \frac{1}{c} P^T P)\right) && (6) \\ &\quad + \text{trace}\left(2(K - M^T M)^T (M^T M - Z^T Z)\right) && (7) \\ &\quad + \text{trace}\left(2(M^T M - Z^T Z)(Z^T Z - \frac{1}{c} P^T P)\right), && (8) \end{aligned}$$

where the second equality follows from the definition of the Frobenius norm. Thus minimizing the LSD objective requires minimizing the sum of the above six terms.

First, consider the third term on line (5): $\|Z^T Z - \frac{1}{c} P^T P\|_F^2$ for some LSDable $Z^T Z$. For any LSDable $Z^T Z$, by definition, we can choose a $c$ and a $P$ such that $Z^T Z = \frac{1}{c} P^T P$. Such a choice will zero-out the third, fourth and sixth terms on lines (5), (6) and (8) respectively. Proposition 2 gives a closed-form solution for such an optimal $c$, and Theorem 3 states that given such a $c$, the optimal matrix $P$ is a rotation of the matrix $\sqrt{c} Z$.

The optimal choice of a rank $k$ matrix $M^T M$ depends on the first and second terms on line (5), and the fifth term on line (7). As an approximation, in our algorithm we will choose $M$ to minimize

only the first term, $\|K - M^T M\|_F^2$. Given the eigendecomposition $K = V\Lambda V^T = \sum_{i=1}^n \lambda_i v_i v_i^T$ such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ (with $\lambda_k > 0$ by our assumption that $K$ is a similarity matrix), the best rank-$k$ matrix approximation in Frobenius norm (or in general any unitarily invariant norm) to matrix $K$ is given as $K^{(k)} = \sum_{i=1}^k \lambda_i v_i v_i^T$ (Eckart and Young, 1936). Setting $M = \left[\sqrt{\lambda_1} v_1 \ \sqrt{\lambda_2} v_2 \ \cdots \sqrt{\lambda_k} v_k\right]$ will thus minimize the first term. If $K^{(k)}$ is LSDable, then setting $Z = M$ will zero-out the second and fifth terms, and this strategy will be optimal, achieving the lowerbound error $\|K - K^{(k)}\|_F^2$ on the LSD objective.

If $K^{(k)}$ is not LSDable, then it remains to choose $Z$. The remaining error would be minimized by choosing the $Z$ that minimizes the second and fifth terms, subject to $Z^T Z$ being LSDable:

$$\|K^{(k)} - Z^T Z\|_F^2 + \text{trace}\left(2(K - K^{(k)})^T (K^{(k)} - Z^T Z)\right). \tag{9}$$

As a heuristic to efficiently approximately minimizing (9), we take advantage of the fact that if $Z^T Z$ is LSDable then the columns of $Z$ must lie on a hyperplane (because the columns of $P$ add to one, and so they lie on a hyperplane, and by Theorem 3, $Z$ is simply a scaled rotation of $P$, and thus its columns must also lie on a hyperplane). So we set the columns of $Z$ to be the least-squares hyperplane fit to the columns of $M$. This in itself is not enough to make $Z^T Z$ LSDable—the convex hull of its columns must fit in the probability simplex, which is achieved by projecting the columns on to the probability simplex. In fact we learn an appropriate rotation and projection via an alternating minimization approach. See Section 3.4 for more details.

See Section 3.2 for complete details on the proposed LSD algorithm, and Section 3.1 for alternative algorithmic approaches.

### 2.2 Optimal Scaling Factor for LSD

The following proposition shows that given any LSDable matrix $Z^T Z$, the optimal $c$ has a closed-form solution and is independent of the optimal $P$. Thus, without loss of generality, other results in this paper assume that $c^* = 1$.

**Proposition 2 (Scaling Factor)** *Let $Z \in \mathbb{R}^{k \times n}$ be a rank-k matrix such that $Z^T Z$ is LSDable. Then, the scaling factor $c^*$ that solves*

$$\begin{aligned} \underset{\substack{c \in \mathbb{R}_+ \\ P \in [0,1]^{k \times n}}}{\textit{minimize}} \quad & \|Z^T Z - \frac{1}{c} P^T P\|_F^2 \\ \textit{subject to} \quad & P^T \mathbf{1}_k = \mathbf{1}_n \end{aligned}$$

*is*

$$c^* = \frac{\|(ZZ^T)^{-1} Z \mathbf{1}_n\|_2^2}{k}. \tag{10}$$

*Further, let $Q \in \mathbb{R}^{k \times n}$ be some matrix such that $Q^T Q = Z^T Z$, then $c^*$ can equivalently be written in terms of $Q$ rather than $Z$.*

Note that the matrix $ZZ^T$ in (10) is invertible because $Z$ is a full rank matrix.

### 2.3 Family of LSD Factorizations

For a given similarity matrix $K$, there will be multiple left-stochastic factors that are related by rotations about the normal to the probability simplex (which includes permuting the rows, that is, changing the cluster labels):

**Theorem 3 (Factors of $K$ Related by Rotation)** *Suppose $K$ is* LSDable *such that $K = P^T P$. Then,* (a) *for any matrix $M \in \mathbb{R}^{k\times n}$ s.t. $K = M^T M$, there is a unique orthogonal $k \times k$ matrix $R$ s.t. $M = RP$.* (b) *for any matrix $\hat{P} \in \mathbb{R}^{k\times n}$ s.t. $\hat{P} \geq 0, \hat{P}^T \mathbf{1}_k = \mathbf{1}_n$ and $K = \hat{P}^T \hat{P}$, there is a unique orthogonal $k \times k$ matrix $R_u$ s.t. $\hat{P} = R_u P$ and $R_u u = u$, where $u = \frac{1}{k}[1,\ldots,1]^T$ is normal to the plane containing the probability simplex.*

### 2.4 Uniqueness of an LSD Clustering

While there may be many left-stochastic decompositions of a given $K$, they may all result in the same LSD clustering. The theorem below gives sufficient conditions for the uniqueness of an LSD clustering. For $k = 2$ clusters, the theorem implies that the LSD clustering is unique. For $k = 3$ clusters, the theorem's condition may be restated as follows (see Figure 1 for an illustration): Let $\alpha_c$ ($\alpha_{ac}$) be the maximum angle of clockwise (anti-clockwise) rotation about the normal to the simplex that does not rotate any of the columns of $P$ off the simplex; let $\beta_c$ ($\beta_{ac}$) be the minimum angle of clockwise (anti-clockwise) rotation that changes the clustering. Then, if $\alpha_c < \beta_c$ and $\alpha_{ac} < \beta_{ac}$, the LSD clustering is unique, up to a permutation of labels. This condition for $k = 3$ can be checked in linear time.

**Theorem 4 (Unique LSD Clustering)** *Let $H_u$ be the subgroup of the rotation group $SO(k)$ in $\mathbb{R}^k$ that leaves the vector $u = \frac{1}{k}[1,\ldots,1]^T \in \mathbb{R}^k$ invariant. Consider the subset*

$$G(P) = \{R \in H_u \mid (RP)_j \in \Delta^k, j = 1,\ldots,n\},$$

*of rotation matrices that rotate all columns of a given LSD factor $P$ into the probability simplex $\Delta^k \subset \mathbb{R}^k$. Consider two partitions of $G(P)$: the first partition $G(P) = \cup_i G_i^{(LSD)}(P)$ such that $R', R'' \in G_i^{(LSD)}(P)$ if and only if $R', R''$ result in the same LSD clustering; the second partition $G(P) = \cup_i G_i^{(conn)}(P)$ such that $R', R'' \in G_i^{(conn)}(P)$ if and only if there is a continuous connected path between $R'$ and $R''$ that is contained in $G_i^{(conn)}(P)$. Let $\mu(\cdot)$ denote the Haar measure on the rotation group $SO(k-1)$ and define*

$$\begin{aligned} \alpha &= \sup_{i \in I^{(conn)}} \mu(G_i^{(conn)}(P)), \\ \beta &= \inf_{i \in I^{(LSD)}} \mu(G_i^{(LSD)}(P)). \end{aligned}$$

*Then, if $\alpha < \beta$, the LSD clustering is unique, up to a permutation of labels.*

### 2.5 Error Bounds on LSD

Our next result uses perturbation theory (Stewart, 1998) to give an error bound on how much an LSD factor is perturbed when an LSDable matrix K is additively perturbed.

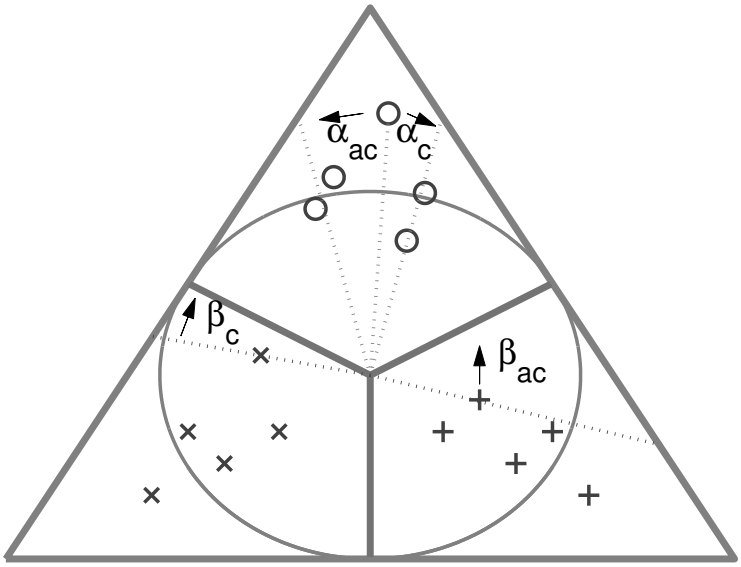



Figure 1: Illustration of conditions for uniqueness of the LSD clustering for the case $k = 3$ and for an LSDable $K$. The columns of $P^*$ are shown as points on the probability simplex for $n = 15$ samples. The Y-shaped thick gray lines show the clustering decision regions, and separate the 15 points into three clusters marked by 'o's, '+'s, and 'x's. One can rotate the columns of $P^*$ about the center of the simplex $u$ to form a different probability matrix $\hat{P} = R_u P^*$, but the inner product does not change $\hat{P}^T \hat{P} = P^{*T} P^* = K$, thus any such rotation is another LSD solution. One can rotate clockwise by at most angle $\beta_c$ (and anti-clockwise $\beta_{ac}$) before a point crosses a cluster decision boundary, which changes the clustering. Note that rotating $P^*$ by more than $\alpha_c$ degrees clockwise (or $\alpha_{ac}$ degrees anti-clockwise) pushes the points out of the probability simplex - no longer a legitimate LSD; but rotating by 120 degrees is a legitimate LSD that corresponds to a re-labeling of the clusters. Theorem 4 applied to the special case of $k = 3$ states that a sufficient condition for the LSD clustering to be unique (up to re-labeling) is if $\alpha_c < \beta_c$ and $\alpha_{ac} < \beta_{ac}$. That is, if one cannot rotate any point across a cluster boundary without pushing another point out of the probability simplex, then the LSD clustering is unique.

**Theorem 5 (Perturbation Error Bound)** *Suppose $K$ is* LSDable *and let $\tilde{K} = K + W$, where $W$ is a symmetric matrix with bounded Frobenius norm, $\|W\|_F \leq \varepsilon$. Then $\|K - \hat{P}^T \hat{P}\|_F \leq 2\varepsilon$, where $\hat{P}$ minimizes (3) for $\tilde{K}$. Furthermore, if $\|W\|_F$ is $o(\tilde{\lambda}_k)$, where $\tilde{\lambda}_k$ is the $k^{th}$ largest eigenvalue of $\tilde{K}$, then there exists an orthogonal matrix $R$ and a constant $C_1$ such that*

$$\|P - R\hat{P}\|_F \leq \varepsilon \left( 1 + C_1 \frac{\sqrt{k}}{|\tilde{\lambda}_k|} \left( \sqrt{\mathrm{tr}(K)} + \varepsilon \right) \right). \tag{11}$$

The error bound in (11) involves three terms: the first term captures the perturbation of the eigenvalues and scales linearly with $\varepsilon$; the second term involves $\|K^{\frac{1}{2}}\|_F = \sqrt{\mathrm{tr}(K)}$ due to the coupling between the eigenvectors of the original matrix $K$ and the perturbed eigenvectors as well as

perturbed eigenvalues; and the third term proportional to $\varepsilon^2$ is due to the perturbed eigenvectors and perturbed eigenvalues. As expected, $\|P - R\hat{P}\|_F \to 0$ as $\varepsilon \to 0$, relating the LSD to the true factor with a rotation, which is consistent with Theorem 3.

## 3. LSD Algorithms

The LSD problem (3) is a nonconvex optimization problem in $P$. Standard NNMF techniques could be adapted to optimize it, as we discuss in Section 3.1. In Section 3.2, we propose a new rotation-based iterative algorithm that exploits the invariance of inner products to rotations to solve (3). For the special case of $k = 2$ clusters, the proposed algorithm requires no iterations. The simple $k = 2$ clustering can be used to produce a fast hierarchical binary-tree clustering, which we discuss in Section 3.3.

### 3.1 NNMF Algorithms to Solve LSD

The LSD problem stated in (3) is a *completely positive* (CP) matrix factorization with an additional left-stochastic constraint (Berman and Shaked-Monderer, 2003). CP problems are a subset of non-negative matrix factorization (NNMF) problems where a matrix $A$ is factorized as $A = B^T B$ and $B$ has non-negative entries. We are not aware of any NNMF algorithms specifically designed to solve problems equivalent to (3). A related NNMF problem has been studied in Ho (2008), where the matrix $K$ to be factorized had fixed column sums, rather than our problem where the constraint is that the matrix factor $P$ has fixed column sums.

A survey of NNMF algorithms can be found in Berry et al. (2007). Standard NNMF approaches can be used to solve the LSD problem with appropriate modifications. We adapted the multiplicative update method of Lee and Seung (2000) for the LSD problem constraints by adding a projection onto the feasible set at each iteration. In Section 5, we show experimentally that both the proposed rotation-based approach and the multiplicative update approach can produce good solutions to (3), but that the rotation based algorithm finds a better solution to the LSD problem, in terms of the LSD objective, than the multiplicative update algorithm.

The LSD problem can also be approximated such that an alternating minimization approach may be used, as in Paatero and Tapper (1994); Paatero (1997). For example, we attempted to minimize $\|K - P^T Q\|_F + \lambda \|P - Q\|_F$ with the LSD constraints by alternately minimizing in $P$ and $Q$, but the convergence was very poor. Other possibilities may be a greedy method using rank-one downdates (Biggs et al., 2008), or gradient descent (Paatero, 1999; Hoyer, 2004; Berry et al., 2007).

### 3.2 Rotation-based LSD Algorithm

We propose a rotation-based algorithm for solving (3) that we refer to as the *LSD algorithm*. The algorithm comprises three main steps: (i) initialize with an eigenvalue factorization (see Section 3.2.1), (ii) rotate the matrix factor to enforce the left-stochastic constraint, which puts each column of the matrix factor in the same plane as the probability simplex (see Section 3.2.2) and (iii) rotate again to enforce non-negativity constraints (see Section 3.2.3) which puts each column of the matrix factor inside the probability simplex. The algorithm is motivated by Theorem 3 which states that all matrix factors are related by a rotation. The complete algorithm is given in Algorithm 1 and Subroutines 1 and 2. Figure 2 summarizes the LSD algorithm for $k = 3$. The notation used in this section is summarized in Table 1.

| Symbols | Description |
|---|---|
| $\|\cdot\|_2$ | $\ell_2$ norm |
| $\|\cdot\|_F$ | Frobenius norm |
| $q_i \geq 0$ | entry-wise inequality: $q_{ij} \geq 0$ for $j = 1, \ldots, k$ |
| $K'$ | scaled similarity matrix (see Section 3.2.1) |
| $\mathcal{H}$ | a hyperplane (see Section 3.2.2) |
| $(Q)_\Delta$ | matrix comprising columns of $Q$ projected onto the probability simplex (see Section 3.2.2) |
| $\mathrm{SO}(k)$ | group of $k \times k$ rotation matrices |
| $\psi_u$ | embedding of $\mathrm{SO}(k-1)$ into $\mathrm{SO}(k)$ as the isotropy subgroup of $u \in \mathbb{R}^k$ (see Section 3.2.3) |
| $\pi$ | a projection from $\mathbb{R}^{k\times k}$ to $\mathbb{R}^{(k-1)\times(k-1)}$; see (13) |
| $g^{(t)}$ | rotation estimate after $t^{th}$ iteration |

Table 1: Notation used in Section 3.2

### 3.2.1 STEP 1: INITIALIZE WITH EIGENVALUE FACTORIZATION

A given similarity matrix is first scaled to get $K' = c^* K$ where $c^*$ is given in Equation (10). Consider the eigendecomposition of the scaled similarity matrix, $K' = \sum_{i=1}^{n} \lambda_i v_i v_i^T$ where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k > 0$. The scaled similarity matrix is factorized as $K' \approx M^T M$, where $M = [\sqrt{\lambda_1} v_1 \ \sqrt{\lambda_2} v_2 \ \cdots \sqrt{\lambda_k} v_k]^T$ is a $k \times n$ real matrix comprising scaled eigenvectors corresponding to the top $k$ eigenvalues. Note that this is where we require that the top $k$ eigenvalues of the similarity matrix be positive. Furthermore, the matrix $M^T M$ is the best rank-$k$ approximation of $K'$ in terms of the Frobenius norm.

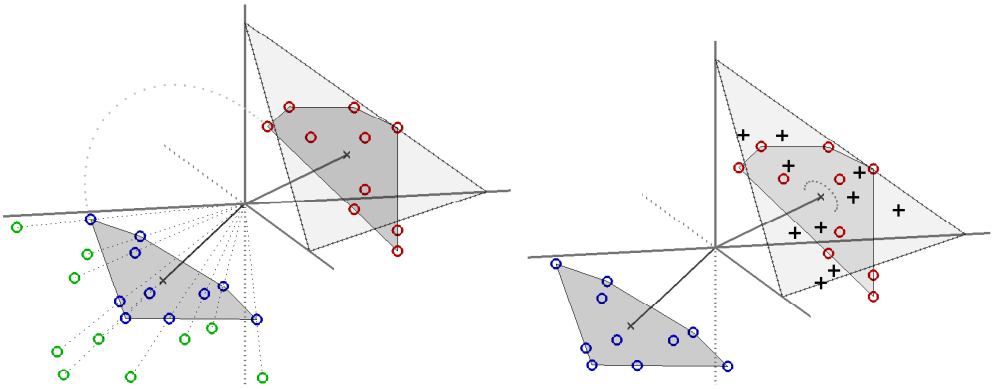
### 3.2.2 STEP 2: ENFORCE LEFT-STOCHASTIC CONSTRAINT

The $n$ columns of $M$ can be seen as points in $k$-dimensional Euclidean space, and the objective of the LSD algorithm is to rotate these points to lie inside the probability simplex in $\mathbb{R}^k$. However, the columns of $M$ may not lie on a hyperplane in $\mathbb{R}^k$. Therefore, the next step is to project the columns of $M$ onto the best fitting (least-squares) hyperplane, $\mathcal{H}$. Let $m$ denote the normal to the hyperplane $\mathcal{H}$, and let $\tilde{M}$ denote the matrix obtained after projection of columns of $M$ onto a hyperplane perpendicular to $m$ and $1/\sqrt{k}$ units away from the origin (see Figure 2(a) for an illustration and Algorithm 1 for details).

Next the columns of $\tilde{M}$ are rotated by a rotation matrix $R_s$ that rotates the unit vector $\frac{m}{\|m\|_2} \in \mathbb{R}^k$ about the origin to coincide with the unit vector $u = \frac{1}{\sqrt{k}}[1, \ldots, 1]^T$, which is normal to the probability simplex; that is, $R_s \frac{m}{\|m\|_2} = \frac{u}{\|u\|_2}$ (see Figure 2(a) for an illustration for the case where $k = 3$). The rotation matrix $R_s$ is computed from a Givens rotation, as described in Subroutine 1. This rotation matrix acts on $\tilde{M}$ to give $Q = R_s \tilde{M}$, whose columns are points in $\mathbb{R}^k$ that lie on the hyperplane containing the probability simplex.

*Special case ($k = 2$):* For the special case of binary clustering with $k = 2$, the rotation matrix $R_s = U R_G U^T$, where $R_G$ is the Givens rotation matrix

$$R_G = \begin{bmatrix} u^T m & -1 + (u^T m)^2 \\ 1 - (u^T m)^2 & u^T m \end{bmatrix},$$



(a) Rotate $M$ to the simplex hyperplane

(b) Rotate about the unit vector until all points are inside the probability simplex

Figure 2: The proposed rotation-based LSD algorithm for a toy example with an LSDable $K$ and $k = 3$. Figure 2a shows the columns of $M$ from Step 2, where $K' = M^T M$. The columns of $M$ correspond to points in $\mathbb{R}^k$, shown here as green circles in the negative orthant. If $K$ is LSDable, the green circles would lie on a hyperplane. We scale each column of $M$ so that the least-squares fit of a hyperplane to columns of $M$ is $1/\sqrt{k}$ units away from the origin (i.e., the distance of the probability simplex from the origin). We then project columns of $M$ onto this hyperplane, mapping the green circles to the blue circles. The normal to this best-fit hyperplane is first rotated to the vector $u = \frac{1}{k}[1,\ldots,1]^T$ (which is normal to the probability simplex), mapping the blue circles to the red circles, which are the columns of $Q$ in Step 3. Then, as shown in Figure 2b, we rotate the columns of $Q$ about the normal $u$ to best fit the points inside the probability simplex (some projection onto the simplex may be needed), mapping the red circles to black crosses. The black crosses are the columns of the solution $P^*$.

with $u = \frac{1}{\sqrt{2}}[1 \quad 1]^T$ and the unitary matrix $U = \begin{bmatrix} \frac{m}{\|m\|_2} & \frac{v}{\|v\|_2} \end{bmatrix}$ with $v = u - \frac{u^T m}{\|m\|_2^2}\, m$. We can then simply satisfy all LSD constraints for the matrix factor $Q = R_s \tilde{M}$ by projecting each column of $Q$ onto the probability simplex (Michelot, 1986), that is, $\hat{P} = (Q)_\Delta$; we use the notation $(Q)_\Delta$ to denote the matrix obtained from a matrix $Q$ by projecting each column of $Q$ onto the probability simplex. For $k = 2$, we do not need Step 3 which finds the rotation about the normal to the simplex that rotates each column of the matrix factor into the simplex. In this special case, there are only two possible rotations that leave the normal to the probability simplex in $\mathbb{R}^2$ invariant, corresponding to permuting the labels.

#### 3.2.3 STEP 3: ENFORCE NON-NEGATIVITY CONSTRAINT

For $k > 2$, a final step is needed in which we rotate the columns of $Q$ about $u$ in the hyperplane containing the simplex to fit each column of $Q$ into the simplex (see Figure 2(b) for an illustration). This requires estimating a rotation matrix, denoted $R_u$, that simultaneously rotates all points (i.e., columns of $Q$) into the probability simplex, thereby satisfying all LSD constraints. The matrix $R_u$

**Algorithm 1: Rotation-based LSD Algorithm**

**Input:** Similarity matrix $K \in \mathbb{R}^{n\times n}$, number of clusters $k$, number of iterations ITER

Compute $K' = c^* K$ where $c^*$ is given in Equation (10)

Compute rank-$k$ eigendecomposition $K'^{(k)} = \sum_{i=1}^{n} \lambda_i v_i v_i^T$ where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k > 0$

Compute $M = [\sqrt{\lambda_1} v_1 \ \sqrt{\lambda_2} v_2 \ \cdots \sqrt{\lambda_k} v_k]^T \ \in \mathbb{R}^{k\times n}$

Compute $m = (MM^T)^{-1} M\mathbf{1}_n$ (normal to least-squares hyperplane fit to columns of $M$)

Compute $\tilde{M} = \left(I - \frac{mm^T}{\|m\|_2}\right) M$ (project columns of $M$ onto the hyperplane normal to $m$ that passes through the origin)

Compute $\tilde{M} = \tilde{M} + \frac{1}{\sqrt{k}\|m\|_2} [m \ldots m]$ (shift columns $1/\sqrt{k}$ units in direction of $m$ )

Compute a rotation $R_s$=`Rotate_Givens`$(m,u)$ (see Subroutine 1 or if $k = 2$ the formula in Section 3.2.2)

Compute matrix $Q = R_s \tilde{M}$

If $k > 2$, compute $R_u =$ `Rotate_Simplex`$(K, Q, ITER)$ (see Subroutine 2),
else set $R_u = I$

Compute the (column-wise) Euclidean projection onto the simplex: $\hat{P} = (R_u Q)_\Delta$

**Output:** Cluster probability matrix $\hat{P}$

is learned using an incremental batch algorithm (adapted from Arora 2009b; Arora and Sethares 2010), as described in Subroutine 2. Note that it may not be possible to rotate each point into the simplex and therefore in such cases we would require a projection step onto the simplex after the algorithm for learning $R_u$ has converged. Learning $R_u$ is the most computationally challenging part of the algorithm, and we provide more detail on this step next.

Following Theorem 3(b), Subroutine 2 tries to find the rotation that best fits the columns of the matrix $Q$ inside the probability simplex by solving the following problem

$$\begin{aligned} \underset{R\in \mathrm{SO}(k)}{\text{minimize}} \quad & \|K' - (RQ)_\Delta^T (RQ)_\Delta\|_F^2 \\ \text{subject to} \quad & Ru = u. \end{aligned} \tag{12}$$

The objective defined in (12) captures the LSD objective attained by the matrix factor $(RQ)_\Delta$. The optimization is over all $k \times k$ rotation matrices $R$ that leave $u$ invariant (the isotropy subgroup of $u$) ensuring that the columns of $RQ$ stay on the hyperplane containing the probability simplex. This invariance constraint can be made implicit in the optimization problem by considering the following map from the set of $(k-1)\times(k-1)$ rotation matrices to the set of $k \times k$ rotation matrices (Arora,

2009a),

$$\begin{aligned} \psi_u : \mathrm{SO}(k-1) &\rightarrow \mathrm{SO}(k) \\ g &\mapsto R_{ue}^T \begin{bmatrix} g & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} R_{ue}, \end{aligned}$$

where $\mathbf{0}$ is a $k \times 1$ column vector of all zeros and $R_{ue}$ is a rotation matrix that rotates $u$ to $e = [0, \ldots, 0, 1]^T$, that is, $R_{ue}u = e$. The matrix $R_{ue}$ can be computed using Subroutine 1 with inputs $u$ and $e$. For notational convenience, we also consider the following map:

$$\begin{aligned} \pi : \quad \mathbb{R}^{k \times k} &\rightarrow \mathbb{R}^{(k-1)\times(k-1)} \\ \begin{bmatrix} A & b \\ c^T & d \end{bmatrix} &\mapsto A, \end{aligned} \tag{13}$$

where $b, c^T \in \mathbb{R}^k$ and $d$ is a scalar.

It is easy to check that any rotation matrix $R$ that leaves a vector $u \in \mathbb{R}^k$ invariant can be written as $\psi_u(g)$ for some rotation matrix $g \in \mathrm{SO}(k-1)$. We have thus exploited the invariance property of the isotropy subgroup to reduce our search space to the set of all $(k-1) \times (k-1)$ rotation matrices. The optimization problem in (12) is therefore equivalent to solving:

$$\underset{g \in \mathrm{SO}(k-1)}{\text{minimize}} \quad \|K' - (\psi_u(g)Q)_\Delta^T (\psi_u(g)Q)_\Delta\|_F^2 \quad . \tag{14}$$

We now discuss our iterative method for solving (14). Let $g^{(t)}$ denote the estimate of the optimal rotation matrix at iteration $t$.

Define matrices $X = \psi(g^{(t)})Q = [x_1 \ \ldots \ x_n]$ and $Y = (\psi(g^{(t)})Q)_\Delta = [y_1 \ \ldots \ y_n]$. Define $D \in \mathbb{R}^{n \times n}$ such that

$$D_{ij} = \begin{cases} 1, & q_i \geq 0, \ \mathbf{1}^T q_i = 1 \\ 0, & \text{otherwise.} \end{cases}$$

Note that $x_i$ represents the column $q_i$ after rotation by the current estimate $\psi(g^{(t)})$ and $y_i$ is the projection of $x_i$ onto the probability simplex. We aim to seek the rotation that simultaneously rotates all $q_i$ into the probability simplex. However, it may not be feasible to rotate all $x_i$ into the probability simplex. Therefore, we update our estimate by solving the following problem:

$$g^{(t+1)} = \arg \min_{g \in \mathrm{SO}(k-1)} \sum_{i=1}^{n} D_{ii} \|y_i - \psi(g)x_i\|_2^2.$$

This is the classical orthogonal Procrustes problem and can be solved globally by singular value decomposition (SVD). Define matrix $T = \pi(YDX^T)$ and consider its SVD, $T = U\Sigma V^T$. Then the next iterate that solves (15) is given as $g^{(t+1)} = UV^T$.

Note that the sequence of rotation matrices generated by Subroutine 2 tries to minimize $\tilde{J}(R) = \|Q - (RQ)_\Delta\|_F$, rather than directly minimizing (12). This is a sensible heuristic because the two problems are equivalent when the similarity matrix is LSDable; that is, when the minimum value of $J(R)$ in (12) over all possible rotations is zero. Furthermore, in the non-LSDable case, the last step, projection onto the simplex, contributes towards the final objective $J(R)$, and minimizing $\tilde{J}(R)$ precisely reduces the total projection error that is accumulated over the columns of $Q$.

**Subroutine 1: `Rotate_Givens` (Subroutine to Rotate a Unit Vector onto Another)**

**Input:** vectors $m, u \in \mathbb{R}^k$

Normalize the input vectors, $m = \frac{m}{\|m\|_2}$, $u = \frac{u}{\|u\|_2}$.

Compute $v = u - \left(u^T m\right)\ m$. Normalize $v = \frac{v}{\|v\|_2}$.

Extend $\{m, v\}$ to a basis $U \in \mathbb{R}^{k \times k}$ for $\mathbb{R}^k$ using Gram-Schmidt orthogonalization.

Initialize $R_G$ to be a $k \times k$ identity matrix.

Form the Givens rotation matrix by setting:

$$(R_G)_{11} = (R_G)_{22} = u^T m,$$
$$(R_G)_{21} = -(R_G)_{12} = u^T v.$$

Compute $R_s = U R_G U^T$.

**Output:** $R_s$ (a rotation matrix such that $R_s \frac{m}{\|m\|_2} = \frac{u}{\|u\|_2}$).

**Subroutine 2: `Rotate_Simplex` (Subroutine to Rotate Into the Simplex)**

**Input:** Similarity matrix $K \in \mathbb{R}^{n \times n}$; matrix $Q = [q_1 \dots q_n] \in \mathbb{R}^{k \times n}$ with columns lying in the probability simplex hyperplane; maximum number of batch iterations $ITER$

Initialize $\psi(g_0), \hat{R}_u$ as $k \times k$ identity matrices.

Compute rotation matrix $R_{ue} = \texttt{Rotate\_Givens}(u, e) \in \mathbb{R}^k$ where $u = \frac{1}{\sqrt{k}}[1, \dots, 1]^T$ and $e = [0, \dots, 0, 1]^T \in \mathbb{R}^k$.

For $t = 1, 2, \dots, ITER$:

Compute matrices $X = R_{ue}\psi(g_{t-1})Q$, $Y = R_{ue}(\psi(g_{t-1})Q)_\Delta$.

Compute the diagonal matrix with $D_{ii} = 1 \iff x_i$ lies inside the simplex.

If $\text{trace}(D) = 0$, **return** $\hat{R}_u$.

Compute $T = \pi(YDX^T)$ where $\pi$ is given in (13).

Compute the SVD, $T = U\Sigma V^T$.

Update $g^{(t)} = UV^T$.

If $J(\psi(g^{(t)})) < J(\hat{R}_u)$, update $\hat{R}_u = \psi(g^{(t)})$.

**Output:** Rotation matrix $\hat{R}_u$.

### 3.3 Hierarchical LSD Clustering

For the special case of $k = 2$ clusters, the LSD algorithm described in Section 3.2 does not require any iterations. The simplicity and efficiency of the $k = 2$ case motivated us to explore a hierarchical binary-splitting variant of LSD clustering, as follows.

Start with all $n$ samples as the root of the cluster tree. Split the $n$ samples into two clusters using the LSD algorithm for $k = 2$, forming two leaves. Calculate the average within-cluster similarity of

the two new leaves, where for the $m$-th leaf cluster $C_m$ the within-cluster similarity is

$$W(C_m) = \frac{1}{n_m(n_m+1)} \sum_{\substack{i,j \in C_m, \\ i \le j}} K_{ij},$$

where $C_m$ is the set of points belonging to the $m$-th leaf cluster, and $n_m$ is the number of points in that cluster. Then, choose the leaf in the tree with the smallest average within-cluster similarity. Create a new similarity matrix composed of only the entries of $K$ corresponding to samples in that leaf's cluster. Split the leaf's cluster into two clusters. Iterate until the desired $k$ clusters are produced.

This top-down *hierarchical LSD clustering* requires running LSD algorithm for $k = 2$ a total of $k-1$ times. It produces $k$ clusters, but does not produce an optimal $n \times k$ cluster probability matrix $P$. In Section 5, we show experimentally that for large $k$ the runtime of hierarchical LSD clustering may be orders of magnitude faster than other clustering algorithms that produce similar within-cluster similarities.

### 3.4 An Alternating Minimization View of the LSD Algorithm

The rotation-based LSD algorithm proposed in Section 3.2 may be viewed as an "alternating minimization" algorithm. We can view the algorithm as aiming to solve the following optimization problem (in a slightly general setting),

$$\begin{array}{ll} \text{minimize}_{P,R} & \|XR - P\|_F^2 \\ \text{subject to} & R^T R = I \\ & P \in \mathcal{C}, \end{array} \tag{15}$$

where $P \in \mathbb{R}^{d \times k}$ and $R \in \mathbb{R}^{k \times k}$ are the optimization variables, $R$ is an orthogonal matrix, $X \in \mathbb{R}^{d \times k}$ is the given data, and $\mathcal{C}$ is any convex set (for example in the LSD problem, it is the unit simplex). Geometrically, in this general problem the goal is to find an orthogonal transform that maps the rows of $X$ into the set $\mathcal{C}$.

Unfortunately this problem is not jointly convex in $P$ and $R$. A heuristic approach is to alternately fix one variable and minimize over the other variable, and iterate. This gives a general algorithm that includes LSD as a special case. The algorithm can be described as follows: At iteration $k$, fix $P_k$ and solve the following problem for $R$,

$$\begin{array}{ll} \text{minimize}_{R} & \|XR - P_k\|_F^2 \\ \text{subject to} & R^T R = I, \end{array}$$

which is the well-known orthogonal Procrustes problem. The optimal solution is $R = UV^T$, where $U, V$ are from the SVD of the matrix $X^T P$, that is, $X^T P = U \Sigma V^T$ (note that $UV^T$ is also known as the "sign" matrix corresponding to $X^T P$). Then fix $R = R_k$ and solve for $P$,

$$\begin{array}{ll} \text{minimize}_{P} & \|XR_k - P\|_F^2 \\ \text{subject to} & P \in \mathcal{C}, \end{array}$$

where the optimal $P$ is the Euclidean projection of $XR_k$ onto the set $\mathcal{C}$. Update $P$ as

$$P_{k+1} = \text{Proj}_{\mathcal{C}}(XR_k),$$

and repeat.

Computationally, the first step of the algorithm described above requires an SVD of a $k \times k$ matrix. The second step requires projecting $d$ vectors of length $k$ onto the set $\mathcal{C}$. In cases where this projection is easy to carry out, the above approach gives a simple and efficient heuristic for problem (15). Note that in the LSD algorithm, $R$ is forced to be a rotation matrix (which is easy to do with a small variation of the first step). Also, at the end of each iteration $k$, the data $X$ is also updated as $X_{k+1} = X_k R_k$, which means the rotation is applied to the data, and we look for further rotation to move our data points into $\mathcal{C}$. This unifying view shows how the LSD algorithm could be extended to other problems with a similar structure but with other constraint sets $\mathcal{C}$.

### 3.5 Related Clustering Algorithms

The proposed rotation-based LSD has some similarities to spectral clustering (von Luxburg, 2006; Ng et al., 2002) and to Perron cluster analysis (Weber and Kube, 2005; Deuflhard and Weber, 2005). Our algorithm begins with an eigenvalue decomposition, and then we work with the eigenvalue-scaled eigenvectors of the similarity matrix $K$. Spectral clustering instead acts on the eigenvectors of the graph Laplacian of $K$ (normalized spectral clustering acts on the eigenvectors of the *normalized* graph Laplacian of $K$). Perron cluster analysis acts on the eigenvectors of row-normalized $K$; however, it is straightforward to show that these are the same eigenvectors as the normalized graph Laplacian eigenvectors and thus for the $k = 2$ cluster case, Perron cluster analysis and normalized spectral clustering are the same (Weber et al., 2004).

For $k > 2$ clusters, Perron cluster analysis *linearly maps* their $n \times k$ eigenvector matrix to the probability simplex to form a soft cluster assignment. In a somewhat similar step, we *rotate* a $n \times k$ matrix factorization to the probability simplex. Our algorithm is motivated by the model $K = P^T P$ and produces an exact solution if $K$ is LSDable. In contrast, we were not able to interpret the Perron cluster analysis as solving a non-negative matrix factorization.

## 4. Experiments

We compared the LSD and hierarchical LSD clustering to nine other clustering algorithms: kernel convex NMF (Ding et al., 2010), unnormalized and normalized spectral clustering (Ng et al., 2002), $k$-means and kernel $k$-means, three common agglomerative linkage methods (Hastie et al., 2009), and the classic DIANA hierarchical clustering method (MacNaughton-Smith et al., 1964). In addition, we compared with hierarchical variants of the other clustering algorithms using the same splitting strategy as used in the proposed hierarchical LSD. We also explored how the proposed LSD algorithm compares against the multiplicative update approach adapted to minimize the LSD objective.

Details of how algorithm parameters were set for all experiments are given in Section 4.1. Clustering metrics are discussed in Section 4.2. The thirteen data sets used are described in Section 4.3.

### 4.1 Algorithm Details for the Experiments

For the LSD algorithms we used a convergence criterion of absolute change in LSD objective dropping below a threshold of $10^{-6}$, that is, the LSD algorithms terminate if the absolute change in the LSD objective at two successive iterations is less than the threshold.

Kernel $k$-means implements $k$-means in the implicit feature space corresponding to the kernel. Recall that $k$-means clustering iteratively assigns each of the samples to the cluster whose mean is the closest. This only requires being able to calculate the distance between any sample $i$ and the mean of some set of samples $\mathcal{J}$, and this can be computed directly from the kernel matrix as follows. Let $\phi_i$ be the (unavailable) implicit feature vector for sample $i$, and suppose we are not given $\phi_i$, but do have access to $K_{ij} = \phi_i^T \phi_j$ for any $i, j$. Then $k$-means on the $\phi$ features can be implemented directly from the kernel using:

$$\begin{aligned}\|\phi_i - \frac{1}{|\mathcal{J}|}\sum_{j\in\mathcal{J}}\phi_j\|_2^2 &= \left(\phi_i - \frac{1}{|\mathcal{J}|}\sum_{j\in\mathcal{J}}\phi_j\right)^T\left(\phi_i - \frac{1}{|\mathcal{J}|}\sum_{j\in\mathcal{J}}\phi_j\right)\\ &= K_{ii} - \frac{2}{|\mathcal{J}|}\sum_{j\in\mathcal{J}}K_{ji} + \frac{1}{|\mathcal{J}|^2}\sum_{j,\ell\in\mathcal{J}}K_{j\ell}.\end{aligned}$$

For each run of kernel $k$-means, we used 100 random starts and chose the result that performed the best with respect to the kernel $k$-means problem.

Similarly, when running the $k$-means algorithm as a subroutine of the spectral clustering variants, we used Matlab's `kmeans` function with 100 random starts and chose the solution that best optimized the $k$-means objective, that is, within-cluster scatter. For each random initialization, `kmeans` was run for a maximum of 200 iterations. For normalized spectral clustering, we used the Ng-Jordan-Weiss version (Ng et al., 2002).

For *kernel convex NMF* (Ding et al., 2010) we used the NMF Matlab Toolbox (Li and Ngom, 2011) with its default parameters. It initializes the NMF by running k-means on the matrix $K$, which treats the similarities as features (Chen et al., 2009a).

The top-down clustering method DIANA (DIvisive ANAlysis) (MacNaughton-Smith et al., 1964; Kaufman and Rousseeuw, 1990) was designed to take a dissimilarity matrix as input. We modified it to take a similarity matrix instead, as follows: At each iteration, we split the cluster with the smallest average within-cluster similarity. The process of splitting a cluster $C$ into two occurs iteratively. First, we choose the point $x_1 \in C$ that has the smallest average similarity to all the other points in the cluster and place it in a new cluster $C_{new}$ and set $C_{old} = C \backslash \{x_1\}$. Then, we choose the point in $C_{old}$ that maximizes the difference in average similarity to the new cluster, as compared to the old; that is, the point $x$ that maximizes

$$\frac{1}{|C_{new}|}\sum_{y\in C_{new}} K_{xy} - \frac{1}{|C_{old}|-1}\sum_{y\neq x, y\in C_{old}} K_{xy}. \tag{16}$$

We place this point in the new cluster and remove it from the old one. That is, we set $C_{new} = C_{new} \cup \{x\}$ and $C_{old} = C_{old} \backslash \{x\}$, where $x$ is the point that maximizes (16). We continue this process until the expression in (16) is non-positive for all $y \in C_{old}$; that is, until there are no points in the old cluster that have a larger average similarity to points in the new cluster, as compared to remaining points in the old cluster.

Many of the similarity matrices used in our experiments are not positive semidefinite. Kernel k-means, LSD methods and kernel convex NMF theoretically require the input matrix to be a positive semidefinite (PSD) matrix, and so we clipped any negative eigenvalues, which produces the closest

(in terms of the Frobenius norm) PSD matrix to the original similarity matrix[2] (see Chen et al. 2009a for more details on clipping eigenvalues in similarity-based learning).

## 4.2 Metrics

There is no single approach to judge whether a clustering is "good," as the goodness of the clustering depends on what one is looking for. We report results for four common metrics: within-cluster similarity, misclassification rate, perplexity, and runtime.

### 4.2.1 AVERAGE WITHIN-CLUSTER SIMILARITY

One common goal of clustering algorithms is to maximize the similarity between points within the same cluster, or equivalently, to minimize the similarity between points lying in different clusters. For example, the classic $k$-means algorithm seeks to minimize within-cluster scatter (or dissimilarity), unnormalized spectral clustering solves a relaxed version of the RatioCut problem, and the Shi-Malik version of spectral clustering solves a relaxed version of the NCut problem (von Luxburg, 2006). Here, we judge clusterings on how well they maximize the average of the within-cluster similarities:

$$\frac{1}{\sum_{m=1}^{k} n_m^2} \sum_{m=1}^{k} \left( \sum_{\substack{i,j \in C_m, \\ i \neq j}} K_{ij} + \sum_{i \in C_m} K_{ii} \right), \tag{17}$$

where $n_m$ is the number of points in cluster $C_m$. This is equivalent to the min-cut problem.

### 4.2.2 MISCLASSIFICATION RATE

As in Ding et al. (2010), the misclassification rate for a clustering is defined to be the smallest misclassification rate over all permutations of cluster labels. Färber et al. (2010) recently argued that such external evaluations are "the best way for fair evaluations" for clustering, but cautioned, "Using classification data for the purpose of evaluating clustering results, however, encounters several problems since the class labels do not necessarily correspond to natural clusters." For example, consider the Amazon-47 data set (see Section 4.3 for details), where the given similarity between two samples (books) A and B is the (symmetrized) percentage of people who buy A after viewing B on Amazon. The given class labels are the 47 authors who wrote the 204 books. A clustering method asked to find 47 clusters might not pick up on the author-clustering, but might instead produce a clustering indicative of sub-genre. This is particularly dangerous for the divisive methods that make top-down binary decisions - early clustering decisions might reasonably separate fiction and non-fiction, or hardcover and paperback. Despite these issues, we agree with other researchers that misclassification rate considered over a number of data sets is a useful way to compare clustering algorithms. We use Kuhn's bipartite matching algorithm for computing the misclassification rate (Kuhn, 1955).

### 4.2.3 PERPLEXITY

An alternate metric for evaluating a clustering given known class labels is conditional perplexity. The conditional perplexity of the conditional distribution $P(L|C)$, of label $L$ given cluster $C$, with

2. Experimental results with the kernel convex NMF code were generally not as good with the full similarity matrix as with the nearest PSD matrix, as suggested by the theory.

conditional entropy $H(L|C)$, is defined to be $2^{H(L|C)}$. Conditional perplexity measures the average number of classes that fall in each cluster, thus the lower the perplexity the better the clustering. Arguably, conditional perplexity is a better metric than misclassification rate because it makes "soft" assignments of labels to the clusters.

#### 4.2.4 RUNTIME

For runtime comparisons, all methods were run on machines with two Intel Xeon E5630 CPUs and 24G of memory. To make the comparisons as fair as possible, all algorithms were programmed in Matlab and used as much of the same code as possible. To compute eigenvectors for LSD and spectral clustering, we used the `eigs` function in Matlab, which computes only the $k$ eigenvalues and eigenvectors needed for those methods.

#### 4.2.5 OPTIMIZATION OF THE LSD OBJECTIVE

Because both our LSD algorithm and the multiplicative update algorithm seek to solve the LSD minimization problem (3), we compare them in terms of (3).

### 4.3 Data Sets

The proposed method acts on a similarity matrix, and thus most of the data sets used are specified as similarity matrices as described in the next subsection. However, to compare with standard k-means, we also considered two popular Euclidean data sets, described in the following subsection. Most of these data sets are publicly available from the cited sources or from `idl.ee.washington.edu/similaritylearning`.

#### 4.3.1 NATIVELY SIMILARITY DATA SETS

We compared the clustering methods on eleven similarity data sets. Each data set provides a pairwise similarity matrix $K$ and class labels for all samples, which were used as the ground truth to compute the misclassification rate and perplexity.

*Amazon-47:* The samples are 204 books, and the classes are the 47 corresponding authors. The similarity measures the symmetrized percent of people who buy one book after viewing another book on Amazon.com. For details see Chen et al. (2009a).

*Aural Sonar:* The samples are 100 sonar signals, and the classes are target or clutter. The similarity is the average of two humans' judgement of the similarity between two sonar signals, on a scale of 1 to 5. For details see Philips et al. (2006).

*Face Rec:* The samples are 945 faces, and the classes are the 139 corresponding people. The similarity is a cosine similarity between the integral invariant signatures of the surface curves of the 945 sample faces. For details see Feng et al. (2007).

*Internet Ads:* The samples are 2359 webpages (we used only the subset of webpages that were not missing features), and the classes are advertising or not-advertising. The similarity is the Tversky similarity of 1556 binary features describing a webpage, which is negative for many pairs of webpages. For details see the UCI Machine Learning Repository and Cazzanti et al. (2009).

*MIREX:* The samples are 3090 pieces of music, and the classes are ten different musical genres. The similarity is the average of three humans' fine-grained judgement of the audio similarity of a

pair of samples. For details see the Music Information Retrieval Evaluation eXchange (MIREX) 2007.

*MSIFT:* The samples are 477 images, class labels are nine scene types, as labeled by humans. The similarity is calculated from the multi-spectral scale-invariant feature transform (MSIFT) descriptors (Brown and Susstrunk, 2011) of two images by taking the average distance $\overline{d}$ between all pairs of descriptors for the two images, and setting the similarity to $e^{-\overline{d}}$. For details see Brown and Susstrunk (2011).

*Patrol:* The samples are 241 people, and the class labels are the eight units they belong to. The binary similarity measures the symmetrized event that one person identifies another person as being in their patrol unit. For details see Driskell and McDonald (2008).

*Protein:* The samples are 213 proteins, and four biochemically relevant classes. The similarity is a sequence-alignment score. We used the pre-processed version detailed in Chen et al. (2009a).

*Rhetoric:* The samples are 1924 documents, and the class labels are the eight terrorist groups that published the documents. The similarity measures KL divergence between normalized histograms of 173 keywords for each pair of documents. Data set courtesy of Michael Gabbay.

*Voting:* The samples are 435 politicians from the United States House of Representatives, and the class label is their political party. The similarity measures the Hamming similarity of sixteen votes in 1984 between any two politicians. For details see the UCI Machine Learning Repository.

*Yeast:* The samples are 2222 genes that have only one of 13 biochemically relevant class labels. The similarity is the Smith-Waterman similarity between different genes. For details see Lanckriet et al. (2004).

#### 4.3.2 NATIVELY EUCLIDEAN DATA SETS

In order to also compare similarity-based clustering algorithms to the standard $k$-means clustering algorithm (Hastie et al., 2009), we used the standard MNIST and USPS benchmark data sets, which each natively consist of ten clusters corresponding to the ten handwritten digits 0-9. We subsampled the data sets to 600 samples from each of the ten classes, for a total of 6000 samples. We compute the locally translation-invariant features proposed by Bruna and Mallat (2011) for each digit image. The $k$-means algorithm computes the cluster means and cluster assignments using the features directly, whereas the similarity-based clustering algorithms use the RBF (radial basis function) kernel to infer the similarities between a pair of data points. The bandwidth of the RBF kernel was tuned for the average within cluster similarity on a small held-out set. We tuned the kernel bandwidth for the kernel $k$-means algorithm and the used the same bandwidth for all similarity-based algorithms. Note that different bandwidths yield different similarity matrices and the resulting average within cluster similarities (computed using Equation (17)) are not directly comparable for two different values of bandwidths. Therefore, we picked the kernel bandwidth that maximized the average within-cluster-similarity in the original feature space (Hastie et al., 2009).

## 5. Results

Results were averaged over 100 different runs,[3] and are reported in Table 2 (LSD objective minimization), Table 3 (average within-cluster similarity), Table 4 (perplexity), Table 5 (misclassifica-

3. For some of the clustering algorithms, such as the linkages, there is no algorithmic randomness, but ties in the linkage values were broken arbitrarily.

tion rate). The results on the MNIST and the USPS data sets are reported in Table 6. Runtimes are reported in Table 7 (runtimes), and while averaged over 100 different runs, the runtime results were highly variable, and so we consider them comparable only in terms of order of magnitude.

### 5.1 Comparison of Rotation-based and Multiplicative Update Approaches to Minimize the LSD Objective

We compared the proposed rotation-based algorithm to the multiplicative-update approach for solving for the LSD optimization problem given in (3), as shown in Table 2. The resulting objective function values for the two algorithms were generally similar, with the LSD algorithm finding a better solution for ten of the eleven data sets.

In terms of within-cluster similarity, the multiplicative update approach was slightly better or tied with the rotation-based algorithm on all eleven similarity data sets. However, in terms of misclassification rate and perplexity, the rotation-based algorithm was almost always better, implying that the rotation-based method is picking up legitimate structure in the data sets even if it is not doing as well at maximizing the within-cluster similarity.

| | LSD (Rotation-based algorithm) | LSD (Multiplicative-update algorithm) |
|---|---|---|
| Amazon47 | **818** | 882 |
| AuralSonar | **17** | 18 |
| FaceRec | **18** | 475 |
| InternetAds | 60093 | **58093** |
| Mirex07 | **632** | 766 |
| MSIFTavg | **14** | **14** |
| Patrol | **19** | 22 |
| Protein | **41** | 43 |
| Rhetoric | **23554** | 25355 |
| Voting | **27** | 28 |
| YeastSW13Class | **559** | **559** |

Table 2: Comparison of optimized LSD objective values.

### 5.2 Comparison of LSD to Other Clustering Algorithms

The LSD model for clustering performed well. Of the 15 clustering methods compared on within-cluster similarity, LSD using the multiplicative-update minimization was the best (or tied for best) on nine of the eleven data sets. LSD using the rotation-based minimization had the best misclassification rate and perplexity most often of all the methods over the total of thirteen data sets.

The runtimes for the LSD algorithms were similar to the kernel convex NMF and spectral clustering methods.

### 5.3 Results for Hierarchical LSD

The most surprising result of our experiments was the good performance of the hierarchical methods in terms of all metrics considered. We implemented the hierarchical LSD to take advantage of

| | Amaz47 | AuralSon | FaceRec | Int Ads | Mirex07 | MSIFT | Patrol | Protein | Rhetoric | Voting | Yeast |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *# classes* | 47 | 2 | 139 | 2 | 10 | 9 | 8 | 4 | 8 | 2 | 13 |
| *# points* | 204 | 100 | 945 | 2359 | 3090 | 477 | 241 | 213 | 1924 | 435 | 2222 |
| LSD (M-Upd.) | **18.83** | **0.44** | 0.88 | -11.31 | **0.05** | **0.38** | **0.09** | **0.51** | **36.13** | **0.79** | **0.66** |
| LSD (Rotation) | 14.56 | **0.44** | 0.86 | **-9.60** | 0.03 | **0.38** | 0.08 | 0.49 | **36.13** | **0.79** | **0.66** |
| Hier. LSD | 16.20 | **0.44** | **1.00** | -12.25 | 0.04 | **0.38** | 0.07 | 0.46 | 33.93 | **0.79** | 0.58 |
| Kernel Conv. NMF | 13.74 | **0.44** | **1.00** | -10.19 | 0.03 | **0.38** | 0.08 | 0.49 | 35.08 | **0.79** | 0.64 |
| Hier. Kernel Conv. NMF | 15.48 | **0.44** | **1.00** | -10.16 | 0.04 | **0.38** | 0.08 | **0.51** | 35.99 | **0.79** | 0.64 |
| Kernel k-Means | 1.15 | 0.43 | **1.00** | -9.97 | 0.01 | 0.37 | 0.02 | 0.41 | 35.03 | **0.79** | 0.52 |
| Hier. Kernel k-Means | 3.04 | 0.43 | **1.00** | -9.97 | 0.01 | 0.37 | 0.03 | 0.38 | 34.76 | **0.79** | 0.53 |
| Unnorm. Spec. | 14.35 | 0.34 | 0.92 | -11.38 | 0.01 | 0.37 | 0.08 | 0.30 | 34.89 | **0.79** | 0.52 |
| Hier. Unnorm. Spec. | 14.97 | 0.34 | **1.00** | -11.38 | 0.01 | 0.37 | 0.08 | 0.30 | 34.89 | **0.79** | 0.52 |
| Norm. Spec | 15.38 | **0.44** | 0.89 | **-9.60** | 0.01 | **0.38** | 0.08 | **0.51** | 35.00 | **0.79** | **0.66** |
| Hier. Norm. Spec. | 15.79 | **0.44** | **1.00** | -9.70 | 0.02 | **0.38** | 0.07 | **0.51** | 35.67 | **0.79** | 0.62 |
| Sing. Link. | 10.71 | 0.33 | **1.00** | -11.32 | 0.01 | 0.37 | 0.08 | 0.29 | 34.87 | 0.55 | 0.52 |
| Comp. Link. | 1.41 | 0.33 | **1.00** | -11.12 | 0.01 | 0.37 | 0.01 | 0.35 | 34.84 | 0.77 | 0.56 |
| Avg. Link. | 14.74 | 0.36 | **1.00** | -11.23 | 0.01 | 0.37 | 0.08 | 0.31 | 34.89 | 0.77 | 0.56 |
| DIANA | 7.89 | **0.44** | 0.98 | -10.61 | 0.01 | 0.37 | 0.08 | 0.35 | 34.91 | **0.79** | 0.62 |

Table 3: Average within-cluster similarity.

the simplicity and efficiency of the $k = 2$ case of the rotation-based LSD algorithm. Because the other clustering algorithms do not have fast special cases for $k = 2$, hierarchical variants of these methods do not offer increased efficiency, but we compared to them for completeness. Surprisingly, the hierarchical variants generally did not degrade performance. For example, the hierarchical normalized spectral clustering is as good or tied with normalized spectral clustering on eight of the eleven similarity data sets.

The hierarchical LSD method performed consistently fast, and achieved good results in terms of within-cluster similarity and misclassification rate. For example, for the Face Recognition data set (with $k = 139$) the runtime of hierarchical LSD is an order of magnitude faster than the LSD rotation-based algorithm, and achieves the highest average within-cluster similarity.

## 6. Conclusions, Further Discussion, and Open Questions

A number of NNMF models and algorithms have been proposed for clustering. In this paper, we proposed a left-stochastic NNMF model for clustering, based on relaxing an ideal kernel model. We showed that the proposed LSD NNMF can be effectively approximately solved using a standard multiplicative update approach, but that the same or better objective values can be reached using a novel rotation-based algorithm. For $k = 2$, the proposed LSD algorithm provides a unique solution without iterations or the need for multiple starts. This fact motivated a fast hierarchical LSD clustering for problems where the number of clusters desired is large. For most data sets, the proposed LSD clustering and hierarchical LSD were top performers.

We showed that the set of possible LSD clusterings is related by rotations and gave conditions for when the LSD clustering is unique. This property makes it trivial to discover *multiple clusterings* (Niu et al., 2010) by simply rotating an LSD solution. In this paper, we only considered converting the LSD solution $P^*$ to a clustering by classifying each sample to the highest-probability cluster. However, an interesting advantage of LSD is that samples could instead be assigned in rank order

| | Amaz47 | AuralSon | FaceRec | Int Ads | Mirex07 | MSIFT | Patrol | Protein | Rhetoric | Voting | Yeast |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *# classes* | 47 | 2 | 139 | 2 | 10 | 9 | 8 | 4 | 8 | 2 | 13 |
| *# points* | 204 | 100 | 945 | 2359 | 3090 | 477 | 241 | 213 | 1924 | 435 | 2222 |
| LSD (M-Upd.) | 1.46 | 1.46 | 23.00 | 1.56 | 5.99 | 7.13 | 1.33 | 1.46 | 7.24 | 1.33 | 6.76 |
| LSD (Rotation) | **1.36** | **1.44** | 24.83 | 1.53 | **4.86** | 7.11 | **1.23** | 1.70 | **6.24** | 1.33 | **6.51** |
| Hier. LSD | 1.48 | **1.44** | 1.25 | 1.54 | 5.11 | 6.63 | 1.84 | 1.67 | **6.24** | 1.33 | 7.16 |
| Kernel Conv. NMF | 1.41 | 1.49 | 1.65 | 1.55 | 4.89 | 5.33 | 1.36 | 1.65 | 7.12 | 1.34 | **6.51** |
| Hier. Kernel Conv. NMF | 1.44 | 1.49 | 1.24 | 1.54 | 5.14 | **5.28** | 1.32 | 1.42 | 7.31 | 1.34 | 6.78 |
| Kernel k-Means | 7.78 | 1.56 | 1.49 | 1.53 | 8.56 | 7.73 | 3.41 | 2.18 | 6.96 | 1.33 | 8.46 |
| Hier. Kernel k-Means | 3.09 | 1.55 | 1.22 | 1.53 | 8.63 | 8.67 | 2.82 | 2.50 | 7.46 | 1.33 | 8.19 |
| Unnorm. Spec. | 1.37 | 1.99 | 17.61 | 1.56 | 9.81 | 8.67 | 1.24 | 3.60 | 7.43 | 1.33 | 8.76 |
| Hier. Unnorm. Spec. | 1.44 | 1.99 | 1.26 | 1.56 | 9.79 | 8.67 | 1.24 | 3.60 | 7.43 | 1.33 | 8.49 |
| Norm. Spec | 1.37 | **1.44** | 1.34 | 1.53 | 8.41 | 5.29 | 1.24 | 1.37 | 7.15 | 1.33 | 6.61 |
| Hier. Norm. Spec. | 1.45 | **1.44** | 1.20 | 1.54 | 7.38 | 5.55 | 1.43 | **1.35** | 7.24 | 1.33 | 6.95 |
| Sing. Link. | 1.61 | 1.99 | 1.06 | 1.56 | 9.87 | 8.67 | 1.24 | 3.68 | 7.43 | 1.94 | 8.78 |
| Comp. Link. | 5.19 | 1.96 | **1.04** | 1.54 | 9.87 | 8.51 | 7.06 | 2.72 | 7.39 | **1.18** | 7.51 |
| Avg. Link. | 1.37 | 1.92 | **1.04** | 1.55 | 9.87 | 8.67 | **1.23** | 3.39 | 7.43 | 1.22 | 7.30 |
| DIANA | 2.18 | 1.52 | 11.60 | **1.47** | 9.87 | 8.67 | **1.23** | 2.58 | 7.46 | 1.33 | 6.71 |

Table 4: Perplexity.

| | Amaz47 | AuralSon | FaceRec | Int Ads | Mirex07 | MSIFT | Patrol | Protein | Rhetoric | Voting | Yeast |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *# classes* | 47 | 2 | 139 | 2 | 10 | 9 | 8 | 4 | 8 | 2 | 13 |
| *# points* | 204 | 100 | 945 | 2359 | 3090 | 477 | 241 | 213 | 1924 | 435 | 2222 |
| LSD (M-Upd.) | 0.38 | 0.15 | 0.89 | 0.44 | **0.66** | 0.77 | 0.16 | 0.13 | 0.80 | 0.10 | 0.77 |
| LSD (Rotation) | **0.24** | **0.14** | 0.87 | 0.35 | **0.66** | 0.77 | 0.09 | 0.33 | **0.75** | 0.10 | 0.76 |
| Hier. LSD | 0.36 | **0.14** | 0.19 | 0.46 | **0.66** | 0.74 | 0.29 | 0.37 | **0.75** | 0.10 | 0.76 |
| Kernel Conv. NMF | 0.26 | **0.14** | 0.28 | 0.26 | **0.66** | 0.69 | 0.17 | 0.25 | 0.81 | 0.10 | 0.76 |
| Hier. Kernel Conv. NMF | 0.31 | **0.14** | 0.19 | 0.26 | **0.66** | 0.67 | 0.14 | 0.13 | 0.80 | 0.10 | 0.74 |
| Kernel k-Means | 0.69 | 0.20 | 0.31 | 0.29 | 0.83 | 0.80 | 0.55 | 0.47 | 0.77 | 0.10 | 0.78 |
| Hier. Kernel k-Means | 0.50 | 0.19 | 0.16 | 0.29 | 0.83 | 0.88 | 0.44 | 0.48 | 0.78 | 0.10 | 0.78 |
| Unnorm. Spec. | 0.25 | 0.49 | 0.92 | 0.17 | 0.88 | 0.88 | 0.09 | 0.65 | 0.78 | 0.09 | 0.78 |
| Hier. Unnorm. Spec. | 0.29 | 0.49 | 0.17 | 0.17 | 0.88 | 0.88 | 0.09 | 0.65 | 0.78 | 0.09 | 0.77 |
| Norm. Spec | 0.26 | **0.14** | 0.23 | 0.31 | 0.81 | **0.62** | 0.09 | 0.11 | 0.81 | 0.10 | 0.78 |
| Hier. Norm. Spec. | 0.31 | **0.14** | 0.19 | 0.31 | 0.78 | 0.66 | 0.18 | **0.10** | 0.80 | 0.10 | 0.75 |
| Sing. Link. | 0.30 | 0.49 | 0.04 | 0.16 | 0.88 | 0.88 | 0.06 | 0.66 | 0.78 | 0.38 | 0.78 |
| Comp. Link. | 0.69 | 0.47 | **0.03** | 0.16 | 0.88 | 0.87 | 0.78 | 0.59 | 0.79 | **0.04** | 0.82 |
| Avg. Link. | 0.25 | 0.41 | **0.03** | 0.16 | 0.88 | 0.88 | **0.05** | 0.62 | 0.78 | 0.05 | 0.72 |
| DIANA | 0.36 | 0.16 | 0.87 | **0.13** | 0.88 | 0.88 | 0.11 | 0.55 | 0.78 | 0.10 | **0.71** |

Table 5: Misclassification rates.

to different clusters to produce a clustering with a desired number of samples in each cluster. This approach can also be used to produce a set of multiple clusterings.

LSD produces cluster probabilities $P^*$, and this feature was not explored in this paper. Experimentally, tests are needed to assess and compare the performance of the actual cluster probabilities (rather than threshold them to form a hard clustering). LSD's probabilistic model does not explicitly assume a particular distribution, but there may be a relationship between the goodness of the LSD model and specific distributions. A related open question is why LSD is better suited to some data sets than others. For example, we know that $k$-means tends to perform well on well-separated

| | USPS | | MNIST | |
|---|---|---|---|---|
| | Perplexity | Misclassification Rate | Perplexity | Misclassification Rate |
| k-Means | 2.23 | 0.34 | 2.98 | 0.47 |
| LSD (M-Upd.) | 2.03 | 0.30 | 2.83 | 0.42 |
| LSD (Rotation) | **1.97** | **0.27** | **2.27** | **0.34** |
| Hier. LSD | 2.38 | 0.41 | 2.89 | 0.45 |
| Kernel Conv. NMF | 2.23 | 0.38 | 2.82 | 0.42 |
| Hier. Kernel Conv. NMF | 2.22 | 0.34 | 2.83 | 0.42 |
| Kernel k-Means | 2.20 | 0.33 | 2.90 | 0.46 |
| Hier. Kernel k-Means | 2.20 | 0.33 | 3.01 | 0.48 |
| Unnorm. Spec. | 9.07 | 0.81 | 8.75 | 0.81 |
| Hier. Unnorm. Spec. | 9.05 | 0.80 | 8.75 | 0.80 |
| Norm. Spec | 2.24 | 0.36 | 3.30 | 0.52 |
| Hier. Norm. Spec. | 2.23 | 0.32 | 3.18 | 0.50 |
| Sing. Link. | 9.20 | 0.83 | 8.80 | 0.81 |
| Comp. Link. | 3.38 | 0.53 | 5.06 | 0.66 |
| Avg. Link. | 3.89 | 0.61 | 7.24 | 0.73 |
| DIANA | 4.02 | 0.65 | 3.64 | 0.55 |

Table 6: Clustering results for two natively Euclidean benchmark data sets. K-means acts on the Euclidean features, all the other algorithms use the same RBF similarity (see text for details).

| | Amaz47 | AuralSon | FaceRec | Int Ads | Mirex07 | MSIFT | Patrol | Protein | Rhetoric | Voting | Yeast |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *# classes* | 47 | 2 | 139 | 2 | 10 | 9 | 8 | 4 | 8 | 2 | 13 |
| *# points* | 204 | 100 | 945 | 2359 | 3090 | 477 | 241 | 213 | 1924 | 435 | 2222 |
| LSD (M-Upd.) | 10 | 0 | 335 | 512 | 761 | 60 | 0 | 1 | 724 | 1 | 756 |
| LSD (Rotation) | 11 | 0 | 1672 | 0 | 40 | 2 | 3 | 12 | 60 | 0 | 883 |
| Hier. LSD | 1 | 0 | 2 | 1 | 4 | 0 | 0 | 0 | 1 | 0 | 1 |
| Kernel Conv. NMF | 14 | 1 | 552 | 1292 | 767 | 3 | 3 | 6 | 589 | 47 | 235 |
| Hier. Kernel Conv. NMF | 4 | 1 | 389 | 1297 | 385 | 50 | 1 | 10 | 1396 | 46 | 1705 |
| Kernel k-Means | 1154 | 9 | 5471 | 26452 | 84831 | 174 | 114 | 49 | 131131 | 44 | 330502 |
| Hier. Kernel k-Means | 267 | 9 | 783 | 262110 | 305100 | 361 | 109 | 69 | 314500 | 44 | 1318310 |
| Unnorm. Spec. | 5 | 0 | 529 | 111 | 13 | 14 | 1 | 1 | 52 | 0 | 258 |
| Hier. Unnorm. Spec. | 6 | 0 | 19 | 111 | 21 | 3 | 1 | 1 | 29 | 0 | 46 |
| Norm. Spec | 5 | 0 | 51 | 209 | 103 | 5 | 1 | 0 | 13 | 0 | 32 |
| Hier. Norm. Spec. | 5 | 0 | 16 | 209 | 19 | 2 | 1 | 0 | 5 | 0 | 6 |
| Sing. Link. | 8 | 1 | 796 | 12319 | 27544 | 103 | 13 | 9 | 6625 | 78 | 10363 |
| Comp. Link. | 9 | 1 | 849 | 13179 | 29279 | 109 | 14 | 10 | 7242 | 83 | 11155 |
| Avg. Link. | 21 | 3 | 2143 | 33892 | 75624 | 279 | 36 | 25 | 18367 | 212 | 27834 |
| DIANA | 1 | 0 | 58 | 21 | 5 | 0 | 2 | 0 | 8 | 5 | 301 |

Table 7: Runtimes in seconds.

and convex clusters, and poorly on data sets where the convex hulls of different clusters intersect. Currently, we lack a similar intuition for LSD, and this question may be intimately related to the question of what the LSD model implies in terms of the geometric distribution of samples among clusters.

Our model assumed a constant scale factor $c$ such that $K = P^T P/c$ for a given similarity matrix $K$. A different scaling could be better. For example, one could normalize $K$ by the inverse degree matrix $D^{-1}$, as done in Perron cluster analysis (Weber et al., 2004), but $D^{-\frac{1}{2}}KD^{-\frac{1}{2}}$ will always be right-stochastic, and $P^T P$ is not generally right-stochastic, so the resulting model $D^{-\frac{1}{2}}KD^{-\frac{1}{2}} = P^T P$ is not sensible. However, some other non-constant scaling could lead to a more sensible model.

We reported results in terms of standard clustering metrics. However, clustering is often a useful step in a bigger processing chain, and not an end to itself. For example, Nelson and Gupta (2007) clustered receivers to initialize algorithms that estimate the locations of multiple transmitters. Recently, Hanusa et al. (2011) clustered likelihood surfaces in order to fuse returns from multiple receivers in tracking. In such cases, the right metric for the clustering methods is the end-to-end application-specific metric; in these examples, the transmitter estimation and the multi-static tracking performance.

The rotation-based matrix factorization approach proposed in this paper can in fact be applied in a much more general setting. For example, it can be used as a heuristic for the completely positive matrix factorization problem, if we enforce only nonnegativity, and drop the $P^T\mathbf{1} = \mathbf{1}$ constraint. More broadly, the approach can be extended to any problem where the goal is to obtain vectors that lie in some *convex cone*, given the matrix of their pairwise inner products. That is, given the *Gram matrix* $K \in \mathbb{R}^{n\times n}$ where $K_{ij} = x_i^T x_j$, we want to find the underlying vectors $x_i \in \mathbb{R}^k$, $i = 1,\ldots,n$, knowing that all $x_i$ belong to the convex cone $\mathcal{C}$ (see problem (15), in Section 3.4, which describes the general algorithm idea). Similar to the LSD algorithm, this general iterative algorithm will start by factorizing the Gram matrix to obtain an initial set of vectors (we assume the correct dimension of the $x_i$ are known) and seek a rotation matrix that maps these vectors into the cone $\mathcal{C}$. At every iteration, the algorithm will project the vectors onto the cone, then update the rotation matrix accordingly as discussed in Section 3.4. Thus the LSD algorithm approach could be applied to other problems with a similar structure but with other constraint sets $\mathcal{C}$. For example, one might want to find an orthogonal transformation that maps a set of matrices into the positive semidefinite cone (this arises, for example, when dealing with estimates of a set of covariance matrices). This is a topic for future exploration.

Another future direction is to explore scalable algorithms for similarity-based clustering. All of the similarity-based algorithms studied in this paper are computationally expensive. For instance, the proposed rotation-based LSD algorithm has space complexity of $O(n^2)$ and computational complexity of $O(n^2k)$. For large data sets the similarity-based algorithms studied in this paper become computationally infeasible. One approach to address the scalability issue is to consider stochastic approximation algorithms that process small number of entries in the similarity matrix at each iteration. Note that the first step in the rotation-based LSD algorithm involves finding a rank-$k$ SVD of a given similarity matrix. For large similarity matrices, we can employ an incremental SVD algorithm that processes a single column of the matrix at each iteration (Arora et al., 2012). Such stochastic approximation approaches for large-scale matrix factorization problems in the kernel setting have recently been shown to be useful for machine learning tasks (Arora and Livescu, 2012, 2013).

## Acknowledgments

We thank James Chen, Sergey Feldman, Bela Frigyik and Kristi Tsukida for helpful discussions. This research was supported by a United States PECASE Award, the United States Office of Naval Research, NSF Award #1027812 and and the NSF CAREER Award ECCS-0847077.

## Appendix A. Proof of Proposition 1

**Proof** Note that the lower bound of (4) is zero; we will show that the LSD clustering solution $P_*$ achieves this lower bound. Let $J(F) = \|\Phi(X) - FP\|_F^2$, then a stationary point with respect to $F$ occurs at $\partial J(F)/\partial F = 0$, which is at $F_* = \Phi(X)P^T(PP^T)^{-1}$. Note that $(PP^T)^{-1}$ exists because $P$ is full rank. With this $F_*$, (4) can be written:

$$\begin{aligned}
&\arg \min_{P:P\geq 0, P^T\mathbf{1}=1} \|\Phi(X) - \phi(X)P^T(PP^T)^{-1}P\|_F^2 \\
\equiv &\arg \min_{P:P\geq 0, P^T\mathbf{1}=1} \operatorname{tr}(K) - 2\operatorname{tr}(KP^T(PP^T)^{-1}P) + \operatorname{tr}(KP^T(PP^T)^{-1}P) \\
\equiv &\arg \min_{P:P\geq 0, P^T\mathbf{1}=1} \operatorname{tr}(K) - \operatorname{tr}(KP^T(PP^T)^{-1}P) \qquad (18)
\end{aligned}$$

Consider any $P_*$ that solves the LSD problem such that $K = P_*^T P_*$, then the objective function in (18) becomes:

$$\begin{aligned}
J(P^*) &= \operatorname{tr}(K) - \operatorname{tr}(P_*^T P_* P_*^T (P_* P_*^T)^{-1} P_*) \\
&= \operatorname{tr}(K) - \operatorname{tr}(P_*^T P_*) \\
&= \operatorname{tr}(K) - \operatorname{tr}(K) \\
&= 0.
\end{aligned}$$

Since the LSD solution $P_*$ achieves the lower bound of (18), it must be a minimizer. ■

## Appendix B. Proof of Proposition 2

**Proof** By definition of LSDable, there exists a scalar $c \in \mathbb{R}_+$ and a left-stochastic matrix $P \in \mathbb{R}^{k\times n}$ such that $cZ^TZ = P^TP$. Then, Theorem 3(a) states that there exists an orthogonal matrix $R \in \mathbb{R}^{k\times k}$ such that

$$R(\sqrt{c}Z) = P. \qquad (19)$$

And by the left-stochasticity of $P$, we know that

$$P^T\mathbf{1}_k = \mathbf{1}_n. \qquad (20)$$

Substituting (19) into (20):

$$\begin{aligned}
&(R(\sqrt{c}Z))^T\mathbf{1}_k = \mathbf{1}_n, \\
\Rightarrow \quad &\sqrt{c}Z^TR^T\mathbf{1}_k = \mathbf{1}_n, \\
\Rightarrow \quad &\sqrt{c}ZZ^TR^T\mathbf{1}_k = Z\mathbf{1}_n, \\
\Rightarrow \quad &R^T\mathbf{1}_k = \frac{1}{\sqrt{c}}(ZZ^T)^{-1}Z\mathbf{1}_n, \qquad (21)
\end{aligned}$$

where $(ZZ^T)^{-1}$ exists because $Z$ is full-rank.

Take the $\ell_2$ norm of both sides of (21) to conclude:

$$\begin{aligned} \|R^T\mathbf{1}_k\|_2 &= \frac{\|(ZZ^T)^{-1}Z\mathbf{1}_n\|_2}{\sqrt{c}}, \\ \Rightarrow \qquad c &= \frac{\|(ZZ^T)^{-1}Z\mathbf{1}_n\|_2^2}{\|R^T\mathbf{1}_k\|_2^2}, \\ \Rightarrow \qquad c &= \frac{\|(ZZ^T)^{-1}Z\mathbf{1}_n\|_2^2}{k}, \end{aligned} \tag{22}$$

because the $\ell_2$ norm is unitary invariant and $\|\mathbf{1}_k\|_2 = \sqrt{k}$.

Next we show that $c^*$ does not depend on any particular factorization of the LSDable matrix $Z^TZ$. Let $Q \in \mathbb{R}^{k\times n}$ be any matrix such that $Q^TQ = Z^TZ$. Then Theorem 3(a) implies that there exists a rotation matrix $\tilde{R}$ such that $\tilde{R}Q = Z$. Substituting $Z = \tilde{R}Q$ in (22),

$$\begin{aligned} c &= \frac{\|(\tilde{R}QQ^T\tilde{R}^T)^{-1}\tilde{R}Q\mathbf{1}_n\|_2^2}{k}, \\ &= \frac{\|\tilde{R}^{-T}(\tilde{R}QQ^T)^{-1}\tilde{R}Q\mathbf{1}_n\|_2^2}{k}, \\ &= \frac{\|\tilde{R}(QQ^T)^{-1}\tilde{R}^{-1}\tilde{R}Q\mathbf{1}_n\|_2^2}{k}, \\ &= \frac{\|(QQ^T)^{-1}Q\mathbf{1}_n\|_2^2}{k}, \end{aligned}$$

where we used the fact that $(AB)^{-1} = B^{-1}A^{-1}$ if matrices $A$ and $B$ are invertible in the second and third steps, and unitary invariance of the $\ell_2$ norm in the last equality. ■

## Appendix C. Proof of Theorem 3

**Proof** (a) If $K$ has two decompositions $K = P^TP = Q^TQ$, for $P,Q \in \mathbb{R}^{m\times n}$ then $K_{ij} = P_i^TP_j = Q_i^TQ_j$, that is, the inner product between the $i$th and $j$th columns of $P$ and $Q$ are the same. The linear transformation that preserves inner-products for all $P_i \in \mathbb{R}^m$ is an orthogonal transformation $R \in O(m)$. The transformation is unique because if there were two different elements $R_1, R_2 \in O(m)$, $R_1 \neq R_2$, such that $P = R_1Q$ and $P = R_2Q$, then $Q = R_1^TR_2Q$, which would imply $R_1^TR_2 = I$. Multiplying both sides from the left by $R_1$, we get $R_2 = R_1$, which is a contradiction.

(b) Let $\mathcal{H} = \{(x_1,\ldots,x_k) \in \mathbb{R}^k | \sum_{j=1}^k x_k = 1\}$ denote the hyperplane that contains the simplex $\Delta^k$. Let $u$ denote the normal to the simplex $\Delta^k$. Then the proof follows from the fact that the subset of orthogonal transformations (from part(a)) that map the hyperplane $\mathcal{H}$ onto itself are the transformations that leave $u$ invariant, that is, the stabilizer subgroup of $u$. ■

## Appendix D. Proof of Theorem 4

**Proof** Let $\mathrm{SO}(k)$ denote the special orthogonal group in $\mathbb{R}^k$, that is, the set of all rotation matrices of size $k\times k$ and let $\mu$ denote the Haar measure associated with $\mathrm{SO}(k-1)$. Given a vector $x \in \mathbb{R}^k$, the

stabilizer subgroup of $x$ is defined to be the set of all rotations that leave $x$ invariant. From Theorem 1(b), we know that all left-stochastic decompositions are related by orthogonal transformations that leave $u = [1,\ldots,1]^T \in \mathbb{R}^k$ fixed. Since we are interested in clusterings that are unique up to a permutation of labels, and the orthogonal group modulo reflections is isomorphic to the rotation group, all left-stochastic decompositions are characterized by the stabilizer subgroup of $u$, given by

$$H_u = \{R \in \mathrm{SO}(k) | Ru = u\}.$$

Let $P$ be any LSD factor of a given similarity matrix $K$, and let $G(P) \subseteq H_u$ be the set of all rotations that gives feasible LSD factors for $K$, that is,

$$G(P) = \{R \in H_u | (RP)_j \in \Delta^k \text{ for } j = 1,\ldots,n\},$$

where $\Delta^k$ is the probability simplex in $\mathbb{R}^k$. Note that the set $G(P)$ is not empty as it always contains the identity matrix. We consider two partitions of $G(P)$: the first partition $G(P) = \cup_{i \in I^{(LSD)}} G_i^{(LSD)}(P)$ induced by the equivalence relationship that $R', R'' \in G_i^{(LSD)}(P)$ if and only if both $R'P$ and $R''P$ give the same LSD clustering; the second partition $G(P) = \cup_{i \in I^{(conn)}} G_i^{(conn)}(P)$ induced by the equivalence relationship that $R', R'' \in G_i^{(conn)}(P)$ if and only if there is a continuous connected path $\{R_t | t \in [0,1]\}$ such that $R_0 = R', R_1 = R''$ and $R_t \in G_i^{(conn)}$ for all $t \in [0,1]$. Note that neither partition is empty because $G(P)$ is not empty, and therefore the following volumes are well defined:

$$\begin{aligned} \alpha &= \sup_{i \in I^{(conn)}} \mu(G_i^{(conn)}(P)), \\ \beta &= \inf_{i \in I^{(LSD)}} \mu(G_i^{(LSD)}(P)). \end{aligned}$$

Next, note that $G(P) = H_u$ if and only if $\|P_j\|_2 \leq \frac{1}{\sqrt{k-1}}$ for all $j = 1,\ldots,n$ (i.e., columns of $P$, seen as points in $\mathbb{R}^k$, lie inside the largest sphere inscribed inside the simplex), in which case $I^{(conn)} = \{1\}$ with $G_1^{(LSD)}(P) = H_u$, giving $\alpha = \mu(H_u) = 1$. Clearly, we cannot have a unique clustering in this case as we can arbitrarily rotate the columns of $P$ about the centroid of the simplex without violating any LSD constraints and resulting in arbitrary LSD clusterings. Furthermore, an upper bound on $\beta$ is $1/k$ because of the symmetry about the centroid of the simplex and so $\alpha \not< \beta$; the uniqueness condition therefore takes care of this trivial scenario.

If the condition $\|P_j\|_2 \leq \frac{1}{\sqrt{k-1}}$ does not hold for all columns of $P$, then there is a non-trivial partition of $G(P)$ into connected components and therefore $\alpha < 1$. The result then follows by a simple observation that the clustering changes while still being a valid LSD if and only if we can rotate a point into a different clustering without rotating any points out of the simplex, that is, there exist sets $G_i^{(LSD)}(P)$ and $G_j^{(conn)}(P)$ in the two partitions such that $G_i^{(LSD)}(P) \subset G_j^{(conn)}(P)$ but then $\alpha \geq \beta$. Therefore, a sufficient condition for the LSD clustering to be unique (up to re-labeling) is if $\alpha < \beta$. ■

## Appendix E. Proof of Theorem 5

**Proof** The error bound in (11) involves three terms: the first term captures the perturbation of the eigenvalues and scales linearly with $\varepsilon$; the second term involves $\|K^{\frac{1}{2}}\|_F = \sqrt{\mathrm{tr}(K)}$ due to the

coupling between the eigenvectors of the original matrix $K$ and the perturbed eigenvectors as well as perturbed eigenvalues; and the third term proportional to $\varepsilon^2$ is due to the perturbed eigenvectors and perturbed eigenvalues. As expected, $||P - R\hat{P}||_F \to 0$ as $\varepsilon \to 0$, relating the LSD to the true factor with a rotation, which is consistent with Theorem 3. Our proof uses standard results from perturbation theory (Stewart, 1998).

Let $\hat{K} = \hat{P}^T\hat{P}$. By the triangle inequality,

$$\begin{aligned} ||K - \hat{K}||_F &= ||K - \tilde{K} + \tilde{K} - \hat{K}||_F \\ &\leq ||K - \tilde{K}||_F + ||\tilde{K} - \hat{K}||_F. \end{aligned}$$

By definition $\hat{K}$ minimizes $||K' - \tilde{K}||_F$ over all $K'$ such that $K' = P'^T P', P' \geq 0, P^T\mathbf{1}_k = \mathbf{1}_n$. Therefore $||\hat{K} - \tilde{K}||_F \leq ||K - \tilde{K}||_F$ which yields,

$$\begin{aligned} \|K - \hat{P}^T\hat{P}\|_F &= ||K - \hat{K}||_F \\ &\leq 2||K - \tilde{K}||_F \\ &= 2||W||_F \\ &\leq 2\varepsilon. \end{aligned}$$

Let $K = E\Lambda E^T$ and $\tilde{K} = \tilde{E}\tilde{\Lambda}\tilde{E}^T$ be the eigendecompositions of $K$ and $\tilde{K}$ respectively. To get a bound on $\hat{P}$, we use a result by Stewart (1998) that states that

$$\tilde{E}_k = R_E E_k + W_E, \tag{23}$$

where $E_k, \tilde{E}_k$ are the first $k$ columns of $E, \tilde{E}$ respectively, $R_E$ is an orthogonal matrix and $W_E$ is a $k \times n$ matrix that describes the perturbation of the eigenvectors. The $2-$norm of the perturbation matrix is bounded by the 2-norm of the additive noise as

$$||W_E||_2 \leq C_1 \frac{||W||_2}{|\delta_k|},$$

where $\delta_k = \tilde{\lambda}_k - \lambda_{k+1}$ is the difference between the $k^{th}$ eigenvalue of the perturbed matrix $\tilde{K}$ and $(k+1)^{th}$ eigenvalue of the original matrix $K$.

Now, from (23), pre-multiplying $\tilde{E}_k^T$ by $\tilde{\Lambda}^{\frac{1}{2}}$ and post-multiplying by $R_1 = R_E^T R_0$ gives

$$\tilde{\Lambda}^{\frac{1}{2}}\tilde{E}_k^T R_1 = \tilde{\Lambda}^{\frac{1}{2}}E_k^T R_0 + \tilde{\Lambda}^{\frac{1}{2}}W_E^T R_1, \tag{24}$$

where $R_0$ is the rotation matrix that gives $P = \Lambda^{\frac{1}{2}}E_k R_0$. The matrix $R_1$ can be written as $\tilde{R}R$ such that $\hat{P} = \tilde{\Lambda}^{\frac{1}{2}}\tilde{E}_k\tilde{R}$. Using this decomposition and perturbation expansion of $\tilde{\Lambda}^{\frac{1}{2}} = \Lambda^{\frac{1}{2}} + W_\Lambda$, where by Mirsky's theorem (Stewart, 1998), $||W_\Lambda||_F \leq ||W||_F$, write (24) as

$$\begin{aligned} \hat{P}R &= (\Lambda^{\frac{1}{2}} + W_\Lambda)E_k^T R_0 + \tilde{\Lambda}^{\frac{1}{2}}W_E^T R_1, \\ &= P + W_\Lambda E_k^T R_0 + \tilde{\Lambda}^{\frac{1}{2}}W_E^T R_1. \end{aligned} \tag{25}$$

Then from (25),

$$\begin{aligned}
||P-\hat{P}R||_F &= ||W_\Lambda E_k^T R_0 + \tilde{\Lambda}^{\frac{1}{2}} W_E^T R_1||_F \\
&\leq ||W_\Lambda E_k^T R_0||_F + ||\tilde{\Lambda}^{\frac{1}{2}} W_E^T R_1||_F \quad \text{(by triangle inequality)} \\
&= ||W_\Lambda||_F + ||\tilde{\Lambda}^{\frac{1}{2}} W_E||_F \quad \text{(by unitary-invariance of Frobenius norm)} \\
&\leq ||W||_F + ||\tilde{\Lambda}^{\frac{1}{2}} W_E||_F \quad \text{(by Mirsky's Theorem)} \\
&\leq ||W||_F + ||\tilde{\Lambda}^{\frac{1}{2}}||_F ||W_E||_F \quad \text{(by Cauchy Schwartz)} \\
&\leq ||W||_F + ||\Lambda^{\frac{1}{2}} + W_\Lambda||_F \, ||W_E||_F \quad \text{(perturbation expansion of eigenvalues)} \\
&\leq ||W||_F + \left( ||\Lambda^{\frac{1}{2}}||_F + ||W||_F \right) ||W_E||_F \quad \text{(by triangle inequaity)} \\
&\leq \varepsilon + \left( ||\Lambda^{\frac{1}{2}}||_F + \varepsilon \right) ||W_E||_F \quad \text{(from the bound on noise)} \\
&= \varepsilon + \left( ||K^{\frac{1}{2}}||_F + \varepsilon \right) ||W_E||_F \quad \text{(by definition)} \\
&\leq \varepsilon + \left( ||K^{\frac{1}{2}}||_F + \varepsilon \right) \sqrt{k} \, ||W_E||_2 \quad \text{(since rank of } W_E \text{ is at most } k) \\
&\leq \varepsilon + \left( ||K^{\frac{1}{2}}||_F + \varepsilon \right) \sqrt{k} \, C_1 \frac{||W||_2}{|\delta_k|} \quad \text{(from Stewart's perturbation theorem)} \\
&\leq \varepsilon + \left( ||K^{\frac{1}{2}}||_F + \varepsilon \right) \sqrt{k} \, C_1 \frac{||W||_F}{|\delta_k|} \quad \text{(since } \|W\|_2 \leq \|W\|_F) \\
&\leq \varepsilon + \left( ||K^{\frac{1}{2}}||_F + \varepsilon \right) \sqrt{k} \, C_1 \frac{\varepsilon}{|\delta_k|} \quad \text{(bound given on additive noise)} \\
&= \varepsilon \left( 1 + C_1 \frac{\sqrt{k}}{|\delta_k|} \left( ||K^{\frac{1}{2}}||_F + \varepsilon \right) \right),
\end{aligned}$$

which gives the desired bound. ■

## References


R. Arora. *Group Theoretical Methods in Signal Processing: Learning Similarities, Transformations and Invariants*. PhD thesis, Univ. of Wisconsin-Madison, 2009a.

R. Arora. On learning rotations. *Advances in Neural Information Processing Systems (NIPS)*, 2009b.

R. Arora and K. Livescu. Kernel CCA for multi-view learning of acoustic features using articulatory measurements. In *Symp. on Machine Learning in Speech and Language Processing*, 2012.

R. Arora and K. Livescu. Multi-view CCA-based acoustic features for phonetic recognition across speakers and domains. In *Int. Conf. on Acoustics, Speech, and Signal Processing*, 2013.

R. Arora and W. A. Sethares. An efficient and stable algorithm for learning rotations. In *Proc. Intl. Conf. Pattern Recognition*, 2010.

R. Arora, M. R. Gupta, A. Kapila, and M. Fazel. Clustering by left-stochastic matrix factorization. *Proc. Intl. Conf. Machine Learning (ICML)*, 2011.

R. Arora, A. Cotter, K. Livescu, and N. Srebro. Stochastic optimization for PCA and PLS. In *50th Annual Allerton Conference on Communication, Control, and Computing*, 2012.

A. Berman and N. Shaked-Monderer. *Completely Positive Matrices*. World Scientific, 2003.

M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics and Data Analysis*, 52(1):155 – 173, 2007.

M. Biggs, A. Ghodsi, and S. Vavasis. Nonnegative matrix factorization via rank-one downdate. *Proc. Intl. Conf. Machine Learning (ICML)*, 2008.

M. Brown and S. Susstrunk. Multispectral SIFT for scene category recognition. *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

J. Bruna and S. Mallat. Classification with scattering operators. *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

L. Cazzanti, M. R. Gupta, and S. Srivastava. Fusing similarities and Euclidean features with generative classifiers. In *Proc. IEEE Conf. on Information Fusion (FUSION)*, 2009.

Y. Chen, E. K. Garcia, M. R. Gupta, L. Cazzanti, and A. Rahimi. Similarity-based classification: Concepts and algorithms. *JMLR*, 2009a.

Y. Chen, M. R. Gupta, and B. Recht. Learning kernels from indefinite similarities. In *Proc. Intl. Conf. Machine Learning (ICML)*, 2009b.

N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel target alignment. *Advances in Neural Information Processing Systems (NIPS)*, 2001.

P. Deuflhard and M. Weber. Robust Perron cluster analysis in conformation dynamics. *Linear Algebra and Its Applications*, 398:161–184, 2005.

C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. *SIAM Conf. Data Mining*, 2005.

C. Ding, T. Li, and M. I. Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Trans. PAMI*, 32, 2010.

J. E. Driskell and T. McDonald. Identification of incomplete networks. Technical Report of Florida Maxima Corporation, 2008.

C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

I. Färber, S. Günnemann, H.-P. Kriegel, P. Kröger, E. Müller, E. Schubert, T. Seidl, and A. Zimek. On using class-labels in evaluation of clusterings. *Proc. ACM SIGKDD*, 2010.

S. Feng, H. Krim, and I. A. Kogan. 3D face recognition using Euclidean integral invariants signature. *Proc. IEEE Workshop Statistical Signal Processing*, 2007.

E. Hanusa, D. Krout, and M. R. Gupta. Clutter rejection by clustering likelihood-based similarities. In *Proc. IEEE Conf. on Information Fusion (FUSION)*, 2011.

T. Hastie, R. Tibshirani, and J.Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, 2nd edition, 2009.

N.-D. Ho. *Nonnegative Matrix Factorizations Algorithms and Applications*. PhD thesis, Universite Catholique de Louvain, 2008.

P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *JMLR*, 5:1457–1469, 2004.

L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data*. Wiley Series in Applied Probability and Statistics. Wiley, 1990.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97, 1955.

G. R. G. Lanckriet, M. Deng, N. Cristianini, M. I. Jordan, and W. S. Noble. Kernel-based data fusion and its application to protein function prediction in yeast. In *Proc. of the Pacific Symposium on Biocomputing*, 2004.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems (NIPS)*, 2000.

Y. Li and A. Ngom. NMF Toolbox Ver. 1.0. `cs.uwindsor.ca/~li11112c/nmf.html`, 2011.

P. MacNaughton-Smith, W.T. Williams, M. B. Dale, and L. G. Mockett. Dissimilarity analysis: A new technique of hierarchical sub-division. *Nature*, 202:1034–5, 1964.

C. Michelot. A finite algorithm for finding the projection of a point onto the canonical simplex of $R^n$. *J. Opt. Theory Appl.*, 50:195–200, 1986.

J. K. Nelson and M. R. Gupta. An EM technique for multiple transmitter localization. *41st Conference on Information Science and Systems*, pages 610–615, 2007.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems (NIPS)*, 2002.

D. Niu, J. G. Dy, and M. I. Jordan. Multiple non-redundant spectral clustering views. *Proc. Intl. Conf. Machine Learning (ICML)*, 2010.

P. Paatero. Least-squares formulation of robust non-negative factor analysis. *Chemometrics and Intell. Lab. Sys.*, 37:23 – 35, 1997.

P. Paatero. The multilinear engine: A table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model. *J. Comp. Graph. Stat.*, 8(4):854–888, 1999.

P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

S. Philips, J. Pitton, and L. Atlas. Perceptual feature identification for active sonar echoes. *Proc. IEEE OCEANS*, 2006.

S. Selim and M. A. Ismail. K-means type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Trans. PAMI*, 6(1):81–87, 1984.

G. W. Stewart. *Matrix Algorithms, Volume I: Basic Decompositions*. SIAM, 1998.

U. von Luxburg. A tutorial on spectral clustering. Technical Report TR-149, Max Planck Institute for Biological Cybernetics, August 2006.

M. Weber and S. Kube. Robust Perron cluster analysis for various applications in computational life science. *Lecture Notes in Computer Science*, 3695:55–66, 2005.

M. Weber, W. Rungsarityotin, and A. Schliep. Perron cluster analysis and its connection to graph partitioning for noisy data. *Zentrum Informationstechnik Berlin Report*, 04-39, 2004.

H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. *Advances in Neural Information Processing Systems (NIPS)*, 2001.

Journal of Machine Learning Research 14 (2013) 1747-1770 Submitted 8/12; Revised 1/13; Published 7/13

# On the Convergence of Maximum Variance Unfolding

**Ery Arias-Castro** EARIASCA@MATH.UCSD.EDU
*Department of Mathematics*
*University of California, San Diego*
*La Jolla, CA 92093, USA*

**Bruno Pelletier** BRUNO.PELLETIER@UNIV-RENNES2.FR
*Département de Mathématiques*
*IRMAR – UMR CNRS 6625*
*Université Rennes II, France*

**Editor:** Mikhail Belkin

## Abstract

Maximum Variance Unfolding is one of the main methods for (nonlinear) dimensionality reduction. We study its large sample limit, providing specific rates of convergence under standard assumptions. We find that it is consistent when the underlying submanifold is isometric to a convex subset, and we provide some simple examples where it fails to be consistent.

**Keywords:** maximum variance unfolding, isometric embedding, U-processes, empirical processes, proximity graphs.

## 1. Introduction

One of the basic tasks in unsupervised learning, aka multivariate statistics, is that of dimensionality reduction. While the celebrated Principal Components Analysis (PCA) and Multidimensional Scaling (MDS) assume that the data lie near an affine subspace, modern approaches postulate that the data are in the vicinity of a submanifold. Many such algorithms have been proposed in the past decade, for example, Isomap (Tenenbaum et al., 2000), Local Linear Embedding (LLE) (Roweis and Saul, 2000), Laplacian Eigenmaps (Belkin and Niyogi, 2003), Manifold Charting (Brand, 2003), Diffusion Maps (Coifman and Lafon, 2006), Hessian Eigenmaps (HLLE) (Donoho and Grimes, 2003), Local Tangent Space Alignment (LTSA) (Zhang and Zha, 2004), Maximum Variance Unfolding (Weinberger et al., 2004), and many others, some reviewed in Van der Maaten et al. (2008) and Saul et al. (2006).

Although some variants exist, the basic setting is that of a connected domain $D \subset \mathbb{R}^d$ isometrically embedded in Euclidean space as a submanifold $M \subset \mathbb{R}^p$, with $p > d$. We are provided with data points $x_1, \dots, x_n \in \mathbb{R}^p$ sampled from (or near) $M$ and our goal is to output $y_1, \dots, y_n \in \mathbb{R}^d$ that can be isometrically mapped to (or close to) $x_1, \dots, x_n$.

A number of consistency results exist in the literature. For example, Bernstein et al. (2000) show that, with proper tuning, geodesic distances may be approximated by neighborhood graph distances when the submanifold $M$ is geodesically convex, implying that Isomap asymptotically recovers the isometry when $D$ is convex. When $D$ is not convex, it fails in general (Zha and Zhang, 2003). Very close in spirit to what we do here, Zha and Zhang (2007) introduce and study a continuum version of Isomap. In accordance with the discrete version, they show that their Continuum Isomap is able to

©2013 Ery Arias-Castro and Bruno Pelletier.

recover an isometry when the manifold is isometric to a convex domain in some lower-dimensional Euclidean space. To justify HLLE, Donoho and Grimes (2003) show that the null space of the (continuous) Hessian operator yields an isometric embedding. See also Ye and Zhi (2012) for related results in a discrete setting. Smith et al. (2008) prove that LTSA is able to recover the isometry, but only up to an affine transformation. We also mention other results in the literature which show that, as the sample size increases, the output the algorithm converges to is an explicit continuous embedding. For instance, a number of papers analyze how well the discrete graph Laplacian based on a sample approximates the continuous Laplace-Beltrami operator on a submanifold (Belkin and Niyogi, 2005; von Luxburg et al., 2008; Singer, 2006; Hein et al., 2005; Giné and Koltchinskii, 2006; Coifman and Lafon, 2006), which is intimately related to the Laplacian Eigenmaps. However, such convergence results do not guaranty that the algorithm is successful at recovering the isometry when one exists. In fact, as discussed in detail by Goldberg et al. (2008) and Perrault-Joncas and Meila (2012), many of them fail in very simple settings.

In this paper, we analyze Maximum Variance Unfolding (MVU) in the large-sample limit. We are only aware of a very recent work of Paprotny and Garcke (2012) that establishes that, under the assumption that $D$ is convex, MVU recovers a distance matrix that approximates the geodesic distance matrix of the data. Our contribution is the following. In Section 2, we prove a convergence result, showing that the optimization problem that MVU solves converges (both in solution space and value) to a continuous version defined on the whole submanifold. The basic assumption here is that the submanifold $M$ is compact. In Section 3, we derive quantitative convergence rates, with mild additional regularity assumptions. In Section 4, we consider the solutions to the continuum limit. When $D$ is convex, we prove that MVU recovers an isometry. We also provide examples of non-convex $D$ where MVU provably fails at recovering an isometry. We also prove that MVU is robust to noise, which Goldberg et al. (2008) show to be problematic for algorithms like LLE, HLLE and LTSA. Some concluding remarks are in Section 5.

## 2. From Discrete MVU to Continuum MVU

In this section we state and prove a qualitative convergence result for MVU. This result applies with only minimal assumptions and its proof is relatively transparent. What we show is that the (discrete) MVU optimization problem converges to an explicit continuous optimization problem when the sample size increases. Although this convergence does not imply a dimensionality reduction per se, the continuous optimization problem is amenable to scrutiny with tools from analysis and geometry, and that will enable us to better understand (in Section 4) when MVU succeeds, and when it fails, at recovering an isometry to a Euclidean domain when it exists.

Let us start by recalling the MVU algorithm (Weinberger and Saul, 2006; Weinberger et al., 2004, 2005). We are provided with data points $x_1, \dots, x_n \in \mathbb{R}^p$. Let $\|\cdot\|$ denote the Euclidean norm. Let $\mathcal{Y}_{n,r}$ be the (random) set defined by

$$\mathcal{Y}_{n,r} = \left\{ y_1, \dots, y_n \in \mathbb{R}^p \,:\, \|y_i - y_j\| \le \|x_i - x_j\| \text{ when } \|x_i - x_j\| \le r \right\}.$$

Having chosen a neighborhood radius $r > 0$, MVU solves the following optimization problem:

DISCRETE MVU

$$\begin{aligned} \text{Maximize} \quad & \mathcal{E}(Y) := \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i} \|y_i - y_j\|^2, \quad \text{over } Y = (y_1, \ldots, y_n)^T \in \mathbb{R}^{n \times p}, \\ \text{subject to} \quad & Y \in \mathcal{Y}_{n,r}. \end{aligned} \tag{1}$$

When the data points are sampled from a distribution $\mu$ with support $M$, our main result in this section is to show that, when $M$ is sufficiently regular and $r = r_n \to 0$ sufficiently slowly, the discrete optimization problem converges to the following continuous optimization problem:

CONTINUUM MVU

$$\begin{aligned} \text{Maximize} \quad & \mathcal{E}(f) := \int_{M \times M} \|f(x) - f(x')\|^2 \mu(\mathrm{d}x) \mu(\mathrm{d}x'), \quad \text{over } f : M \to \mathbb{R}^p, \\ \text{subject to} \quad & f \text{ is Lipschitz with } \|f\|_{\mathrm{Lip}} \leq 1. \end{aligned}$$

$\|f\|_{\mathrm{Lip}}$ denotes the smallest Lipschitz constant of a function $f$. It is important to realize that the Lipschitz condition is with respect to the intrinsic metric on $M$ (i.e., the metric inherited from the ambient space $\mathbb{R}^p$), defined as follows: for $x, x' \in M$, let

$$\delta_M(x,x') = \inf\{T : \exists \gamma : [0,T] \to M, \text{ 1-Lipschitz, with } \gamma(0) = x \text{ and } \gamma(T) = x'\}.$$

When $M$ is compact, the infimum is attained. In that case, $\delta_M(x,x')$ is the length of the shortest continuous path on $M$ starting at $x$ and ending at $x'$, and $(M, \delta_M)$ is a complete metric space, also called a *length space* in the context of metric geometry (Burago et al., 2001). Then $f : M \to \mathbb{R}^p$ is Lipschitz with $\|f\|_{\mathrm{Lip}} \leq L$ if

$$\|f(x) - f(x')\| \leq L \, \delta_M(x,x'), \ \forall x, x' \in M. \tag{2}$$

For any $L > 0$, denote by $\mathcal{F}_L$ the class of Lipschitz functions $f : M \to \mathbb{R}^p$ satisfying (2).

One of the central condition is that $M$ is sufficiently regular that the intrinsic metric on $M$ is locally close to the ambient Euclidean metric.

*Regularity assumption.* There is a non-decreasing function $c : [0,\infty) \to [0,\infty)$ such that $c(r) \to 0$ when $r \to 0$, such that, for all $x, x' \in M$,

$$\delta_M(x,x') \leq \big(1 + c(\|x - x'\|)\big) \|x - x'\|. \tag{3}$$

This assumption is also central to Isomap. Bernstein et al. (2000) prove that it holds when $M$ is a compact, smooth and geodesically convex submanifold (e.g., without boundary). In Lemma 4, we extend this to compact, smooth submanifolds with smooth boundary, and to tubular neighborhoods of such sets. The latter allows us to study noisy settings.

Note that we always have

$$\|x - x'\| \leq \delta_M(x,x'). \tag{4}$$

Let $\mathcal{S}_1$ denote the set of functions that are solutions of Continuum MVU. We state the following qualitative result that makes minimal assumptions.

**Theorem 1** *Let $\mu$ be a (Borel) probability distribution with support $M \subset \mathbb{R}^p$, which is connected, compact and satisfying* (3)*, and assume that $x_1, \ldots, x_n$ are sampled independently from $\mu$. Then, for $r_n \to 0$ sufficiently slowly, we have*

$$\sup\{\mathcal{E}(Y) : Y \in \mathcal{Y}_{n,r_n}\} \to \sup\{\mathcal{E}(f) : f \in \mathcal{F}_1\}, \tag{5}$$

*and for any solution $\hat{Y}_n = (\hat{y}_1, \ldots, \hat{y}_n)$ of Discrete MVU,*

$$\inf_{f \in \mathcal{S}_1} \max_{1 \le i \le n} \|\hat{y}_i - f(x_i)\| \to 0, \tag{6}$$

*almost surely as $n \to \infty$.*

Thus Discrete MVU converges to Continuum MVU in the large sample limit, if $M$ satisfies the crucial regularity condition (3) and other mild assumptions. In Section 3, we provide explicit quantitative bounds for the convergence results (5) and (6) at the very end, under some additional (though natural) assumptions. In Section 4, we focus entirely on Continuum MVU, with the goal of better understanding the functions that are solutions to that optimization problem. Because of (6), we know that the output of Discrete MVU converges in a strong sense to one of these functions.

The rest of the section is dedicated to proving Theorem 1. We divide the proof into several parts which we discuss at length, and then assemble to prove the theorem.

### 2.1 Coverings and Graph Neighborhoods

For $r > 0$, let $G_r$ denote the undirected graph with nodes $x_1, \ldots, x_n$ and an edge between $x_i$ and $x_j$ if $\|x_i - x_j\| \le r$. This is the $r$-neighborhood graph based on the data. Remember that we consider Discrete MVU with $r = r_n \to 0$ as the sample size $n \to \infty$. For the result to hold, it is essential that $G_{r_n}$ be connected, for otherwise $\sup\{\mathcal{E}(Y) : Y \in \mathcal{Y}_{n,r_n}\} = \infty$, while $\sup\{\mathcal{E}(f) : f \in \mathcal{F}_1\}$ is finite. The latter comes from the fact that, for any $f \in \mathcal{F}_1$,

$$\mathcal{E}(f) \le \int_{M \times M} \delta_M(x,x')^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \le \mathrm{diam}(M)^2,$$

where we used (2) in the first inequality, and $\mathrm{diam}(M)$ is the intrinsic diameter of $M$, that is,

$$\mathrm{diam}(M) := \sup_{x,x' \in M} \delta_M(x,x').$$

Recall that the only assumptions on $M$ made in Theorem 1 are that $M$ is compact, connected, and satisfies (3), and this implies that $\mathrm{diam}(M) < \infty$. Indeed, as a compact subset of $\mathbb{R}^p$, $M$ is bounded, hence $\sup_{x,x' \in M} \|x - x'\| < \infty$. Reporting this in (3) immediately implies that $\mathrm{diam}(M) < \infty$.

That said, we ask more of $(r_n)$ than simply having $G_{r_n}$ connected. For $\eta > 0$, define

$$\Lambda(\eta) = \{\forall x \in M, \exists i = 1, \ldots, n : \|x - x_i\| \le \eta\},$$

which is the event that $x_1, \ldots, x_n$ forms an $\eta$-covering of $M$.

*Connectivity requirement.* $r_n \to 0$ in such a way that

$$\sum_{n=1}^{\infty} \mathbb{P}\left(\Lambda(\lambda_n r_n)^c\right) < \infty, \text{ for some sequence } \lambda_n \to 0. \tag{7}$$

Since $M$ is the support of $\mu$, there is always a sequence $(r_n)$ that satisfy the Connectivity requirement. For $\eta > 0$, we say that $z_1, \ldots, z_N \in M$ is $\eta$-packing (for the Euclidean metric) if it is a maximal collection of points such that $\|z_i - z_j\| > \eta$ for all $i \neq j$, meaning there is no other point $z \in M$ such that $\|z - z_i\| > \eta$ for all $i$. By definition then, an $\eta$-packing $z_1, \ldots, z_N \in M$ is also an $\eta$-covering, in the sense that, for all $z \in M$, there is $i \in \{1, \ldots, N\}$ such that $\|z - z_i\| \leq \eta$. (Note that not all $\eta$-coverings are $\eta$-packings.) Let $N_\eta$ denote the maximum cardinality of an $\eta$-packing of $M$. By compactness of $M$, $N_\eta < \infty$ for any $\eta > 0$. Let $p_\eta = \min_j \mu(B(z_j, \eta))$, where $B(z, \eta)$ denotes the Euclidean ball centered at $z$ and of radius $\eta > 0$. Since $M$ is the support of $\mu$, $\mu(B(z, \eta)) > 0$ for any $z \in M$ and any $\eta > 0$. Hence, $p_\eta > 0$ for any $\eta > 0$. We have

$$\begin{aligned}
\mathbb{P}(\Lambda(2\eta)^c) &= \mathbb{P}(\text{there exists } x \in M : \forall i = 1, \ldots, n, \|x - x_i\| > 2\eta\ ) \\
&\leq \mathbb{P}(\text{there is } j \text{ such that } B(z_j, \eta) \text{ is empty of data points}) \\
&\leq \sum_{j=1}^{N_\eta} \mathbb{P}(B(z_j, \eta) \text{ is empty of data points}) \\
&\leq N_\eta (1 - p_\eta)^n.
\end{aligned}$$

Let $\eta_n = \inf\{\eta > 0 : N_\eta(1 - p_\eta)^n \leq 1/n^2\}$ ; the sequence $1/n^2$ is chosen here for the simplicity of the exposition, but more general sequence can be considered, as will become apparent at the end of the paragraph.

Since $p_\eta > 0$ for all $\eta > 0$, $\eta_n \to 0$. To see this, let $\eta^\star = \mathrm{diam}(M)$. Clearly, for all $\eta \geq \eta^\star$, $p_\eta = 1$, which implies that the set of $\eta > 0$ such that $N_\eta(1 - p_\eta)^n \leq 1/n^2$ is non-empty. In particular, for all $n \geq 1$, we have $\eta_n \leq \eta^\star$. Now, let $\varepsilon > 0$ be fixed. Since $p_\varepsilon > 0$, there exists an integer $n_\varepsilon$ such that $N_\varepsilon(1 - p_\varepsilon)^n \leq 1/n^2$ for all $n \geq n_\varepsilon$, so that $\eta_n \leq \varepsilon$ for all $n \geq n_\varepsilon$. Since $\varepsilon$ is arbitrary, this proves that the sequence $(\eta_n)$ converges to 0 as $n$ tends to infinity.

With such a choice of $(\eta_n)$, we have $\sum_{n \geq 1} \mathbb{P}(\Lambda(2\eta_n)^c) \leq \sum_{n \geq 1} 1/n^2 < \infty$. Therefore, if we take $r_n = \sqrt{\eta_n}$, it satisfies the Connectivity requirement. In Section 3.2 we derive a quantitative bound on $r_n$ that guaranty (7) under additional assumptions. Note that the sequence $(1/n^2)$ in the definition of $\eta_n$ can be replaced by any summable decreasing sequence.

The rationale behind the requirement on $(r_n)$ is the same as in Bernstein et al. (2000): it allows to approximate each curve on $M$ with a path in $G_{r_n}$ of nearly the same length. We use this in the following subsection.

### 2.2 Interpolation

Assuming that the sampling is dense enough that $\Lambda(\eta)$ holds, we interpolate a set of vectors $Y \in \mathcal{Y}_{n,r}$ with a Lipschitz function $f \in \mathcal{F}_{1+O(\eta/r)}$. Formally, we have the following.

**Lemma 1** *Assume that $\Lambda(\eta)$ holds for some $\eta \leq r/4$. Then any vector $Y = (y_1, \ldots, y_n) \in \mathcal{Y}_{n,r}$ is of the form $Y = (f(x_1), \ldots, f(x_n))$ for some $f \in \mathcal{F}_{1+6\eta/r}$.*

We prove this result. The first step is to show that this is at all possible in the sense that

$$\|y_i - y_j\| \leq \big(1 + 6\eta/r\big)\delta_M(x_i, x_j),\ \forall i, j. \tag{8}$$

This shows that the map $g : \{x_1, \ldots, x_n\} \to \mathbb{R}^p$ defined by $g(x_i) = y_i$ for all $i$, is Lipschitz (for $\delta_M$ and the Euclidean metrics) with constant $L = 1 + 6\eta/r$. We apply a form of Kirszbraun's Extension—Lang and Schroeder (1997, Theorem B) or Brudnyi and Brudnyi (2012, Theorem 1.26)—to extend $g$ to the whole $M$ into $f \in \mathcal{F}_{1+6\eta/r}$.

Therefore, let's turn to proving (8). The arguments are very similar to those in Bernstein et al. (2000). If $\delta_M(x_i,x_j) \leq r$, then, by (4), $\|x_i - x_j\| \leq r$, which implies that

$$\|y_i - y_j\| \leq \|x_i - x_j\| \leq \delta_M(x_i,x_j).$$

Now suppose that $\delta_M(x_i,x_j) > r$. Let $\gamma$ be a path in $M$ connecting $x_i$ to $x_j$ of minimal length $l = \delta_M(x_i,x_j)$. Split $\gamma$ into $N$ arcs of lengths $l_1 = r/2$ plus one arc of length $l_{N+1} < l_1$, so that

$$\frac{l}{l_1} - 1 \leq N \leq \frac{l}{l_1}.$$

Denote by $x_i = x'_0, x'_1, \ldots, x'_N, x'_{N+1} = x_j$ the extremities of the arcs along $\gamma$.

For $k = 1, \ldots, N$, let $t_k \in \arg\min_t \|x'_k - x_t\|$. Assuming $\Lambda_n(\eta)$ holds, for all $k$ we have $\delta_M(x'_k, x_{t_k}) \leq \eta$, so that

$$\|x_{t_k} - x_{t_{k-1}}\| \leq \delta_M(x_{t_k}, x_{t_{k-1}}) \leq \delta_M(x'_k, x'_{k-1}) + 2\eta \leq l_1 + 2\eta \leq r/2 + 2(r/4) = r.$$

Hence, because $Y = (y_1 \ldots, y_n) \in \mathcal{Y}_{n,r}$,

$$\|y_{t_k} - y_{t_{k-1}}\| \leq l_1 + 2\eta.$$

Similarly, for the last arc, recalling that $x_{t_{N+1}} = x_j$, we have $\delta_M(x_j, x_{t_N}) \leq l_{N+1} + \eta < l_1 + \eta < r$, and therefore

$$\|y_{t_{N+1}} - y_{t_N}\| \leq l_{N+1} + \eta.$$

Consequently,

$$\begin{aligned} \|y_i - y_j\| &\leq N(l_1 + 2\eta) + (l_{N+1} + \eta) \\ &= N l_1 + l_{N+1} + (2N+1)\eta \\ &= l + (2N+1)\eta. \end{aligned}$$

We have

$$(2N+1)\eta \leq \left(2\frac{l}{l_1} + 1\right)\eta \leq l\frac{3\eta}{l_1} = l\frac{6\eta}{r},$$

and so (8) holds.

### 2.3 Bounds on The Energy

We call $\mathcal{E}$ the energy functional. For a function $f : \{x_1, \ldots, x_n\} \to \mathbb{R}^p$, let $Y_n(f) = (f(x_1), \ldots, f(x_n))^T \in \mathbb{R}^{n \times p}$. Assume that $\Lambda(\eta)$ holds $\eta \leq r/4$. Then Lemma 1 implies that any $Y \in \mathcal{Y}_{n,r}$ is equal to $Y_n(f)$ for some $f \in \mathcal{F}_{1+6\eta/r}$. Hence,

$$\sup_{Y \in \mathcal{Y}_{n,r}} \mathcal{E}(Y) \leq \sup_{f \in \mathcal{F}_{1+6\eta/r}} \mathcal{E}(Y_n(f)). \tag{9}$$

Recall the function $c(r)$ introduced in (3), and assume that $r > 0$ is small enough that $c(r) < 1$. For $f \in \mathcal{F}_{1-c(r)}$, and for any $i, j$ such that $\|x_i - x_j\| \leq r$, we have

$$\|f(x_i) - f(x_j)\| \leq (1 - c(r))\delta_M(x_i,x_j) \leq (1 - c(r))(1 + c(\|x_i - x_j\|))\|x_i - x_j\|.$$

Since the function $c$ is non-decreasing, $c(\|x_i - x_j\|) \leq c(r)$, and so

$$\|f(x_i) - f(x_j)\| \leq \left(1 - c(r)^2\right) \|x_i - x_j\| \leq \|x_i - x_j\|.$$

Consequently, $Y_n(f) \in \mathcal{Y}_{n,r}$, implying that

$$\sup_{Y \in \mathcal{Y}_{n,r}} \mathcal{E}(Y) \geq \sup_{f \in \mathcal{F}_{1-c(r)}} \mathcal{E}(Y_n(f)). \tag{10}$$

As a result of (9) and (10), we have

$$\Big| \sup_{Y \in \mathcal{Y}_{n,r}} \mathcal{E}(Y) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big| \leq \sup_{1-c(r) \leq L \leq 1+6\eta/r} \Big| \sup_{f \in \mathcal{F}_L} \mathcal{E}(Y_n(f)) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big|. \tag{11}$$

We have

$$\Big| \sup_{f \in \mathcal{F}_L} \mathcal{E}(Y_n(f)) - \sup_{f \in \mathcal{F}_L} \mathcal{E}(f) \Big| \leq \sup_{f \in \mathcal{F}_L} \Big| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \Big|,$$

and applying the triangle inequality, we arrive at

$$\Big| \sup_{f \in \mathcal{F}_L} \mathcal{E}(Y_n(f)) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big| \leq \sup_{f \in \mathcal{F}_L} \Big| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \Big| + \Big| \sup_{f \in \mathcal{F}_L} \mathcal{E}(f) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big|.$$

Since $\mathcal{F}_L = L\mathcal{F}_1$ and $\mathcal{E}(Lf) = L^2 \mathcal{E}(f)$, we have

$$\Big| \sup_{f \in \mathcal{F}_L} \mathcal{E}(f) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big| \leq |L^2 - 1| \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \leq |L^2 - 1| \operatorname{diam}(M)^2,$$

and

$$\sup_{f \in \mathcal{F}_L} \Big| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \Big| = L^2 \sup_{f \in \mathcal{F}_1} \Big| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \Big|. \tag{12}$$

Consequently,

$$\Big| \sup_{f \in \mathcal{F}_L} \mathcal{E}(Y_n(f)) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big| \leq L^2 \sup_{f \in \mathcal{F}_1} \Big| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \Big| + |L^2 - 1| \operatorname{diam}(M)^2.$$

Reporting this inequality in (11) on the event $\Lambda(\eta)$ with $\eta \leq r/4$, we have

$$\Big| \sup_{Y \in \mathcal{Y}_{n,r}} \mathcal{E}(Y) - \sup_{f \in \mathcal{F}_1} \mathcal{E}(f) \Big| \leq (1 + 6\eta/r)^2 \sup_{f \in \mathcal{F}_1} \Big| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \Big| + \beta(r,\eta)\big(2 + \beta(r,\eta)\big) \operatorname{diam}(M)^2, \tag{13}$$

where $\beta(r,\eta) := \max(c(r), 6\eta/r)$.

Finally, we show that $\mathcal{E}$ is continuous (in fact Lipschitz) on $\mathcal{F}_1$ for the supnorm. For any $f$ and $g$ in $\mathcal{F}_1$, and any $x$ and $x'$ in $M$, we have:

$$\begin{aligned}
& \Big| \|f(x) - f(x')\|^2 - \|g(x) - g(x')\|^2 \Big| \\
\leq\ & \|f(x) - f(x') - g(x) + g(x')\| \, \|f(x) - f(x') + g(x) - g(x')\| \\
\leq\ & \big[\|f(x) - g(x)\| + \|f(x') - g(x')\|\big] \big[\|f(x) - f(x')\| + \|g(x) - g(x')\|\big] \\
\leq\ & 4\|f - g\|_\infty \operatorname{diam}(M).
\end{aligned}$$

The first inequality is that of Cauchy-Schwarz. Hence,

$$\big| \mathcal{E}(f) - \mathcal{E}(g) \big| \leq 4\|f - g\|_\infty \operatorname{diam}(M), \tag{14}$$

and

$$\big| \mathcal{E}(Y_n(f)) - \mathcal{E}(Y_n(g)) \big| \leq 4\|f - g\|_\infty \operatorname{diam}(M). \tag{15}$$

### 2.4 More Coverings and the Law of Large Numbers

The last step is to show that the supremum of the empirical process (12) converges to zero. For this, we use a packing (covering) to reduce the supremum over $\mathcal{F}_1$ to a maximum over a finite set of functions. We then apply the Law of Large Numbers to each difference in the maximization.

Fix $x_0 \in M$ and define

$$\mathcal{F}_1^0 = \{f \in \mathcal{F}_1 : f(x_0) = 0\}.$$

Note that $f \in \mathcal{F}_1$ if, and only if, $f - f(x_0) \in \mathcal{F}_1^0$, and by the fact that $\mathcal{E}(f+a) = \mathcal{E}(f)$ for any function or vector $f$ and any constant $a \in \mathbb{R}^p$, we have

$$\sup_{f \in \mathcal{F}_1} \left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right| = \sup_{f \in \mathcal{F}_1^0} \left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right|.$$

The reason to use $\mathcal{F}_1^0$ is that it is bounded in supnorm. Indeed, for $f \in \mathcal{F}_1^0$, we have

$$\|f(x)\| = \|f(x) - f(x_0)\| \leq \delta_M(x, x_0) \leq \operatorname{diam}(M), \ \forall x \in M.$$

Let $\mathcal{N}_\infty(\mathcal{F}_1^0, \varepsilon)$ denote the covering number of $\mathcal{F}_1^0$ for the supremum norm, that is, the minimal number of balls that are necessary to cover $\mathcal{F}_1^0$, and let $f_1, \ldots, f_N \in \mathcal{F}_1$ be an $\varepsilon$-covering of $\mathcal{F}_1^0$ of minimal size $N := \mathcal{N}_\infty(\mathcal{F}_1^0, \varepsilon)$. Since $\mathcal{F}_1^0$ is equicontinuous and bounded, it is compact for the topology of the supremum norm by the Arzelà-Ascoli Theorem, so that $\mathcal{N}_\infty(\mathcal{F}_1^0, \varepsilon) < \infty$ for any $\varepsilon > 0$.

Fix $f \in \mathcal{F}_1^0$ and let $k$ be such that $\|f - f_k\| \leq \varepsilon$. By (14) and (15), we have

$$\begin{aligned} |\mathcal{E}(Y_n(f)) - \mathcal{E}(f)| &\leq |\mathcal{E}(Y_n(f)) - \mathcal{E}(Y_n(f_k))| + |\mathcal{E}(Y_n(f_k)) - \mathcal{E}(f_k)| + |\mathcal{E}(f_k) - \mathcal{E}(f)| \\ &\leq 8 \operatorname{diam}(M) \|f - f_k\|_\infty + |\mathcal{E}(Y_n(f_k)) - \mathcal{E}(f_k)| \\ &= 8 \operatorname{diam}(M) \varepsilon + |\mathcal{E}(Y_n(f_k)) - \mathcal{E}(f_k)|. \end{aligned}$$

Thus,

$$\sup_{f \in \mathcal{F}_1} \left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right| \leq 8 \operatorname{diam}(M) \varepsilon + \max\{|\mathcal{E}(Y_n(f_k)) - \mathcal{E}(f_k)| : k = 1, \ldots, \mathcal{N}_\infty(\mathcal{F}_1^0, \varepsilon)\}. \quad (16)$$

The Law of Large Numbers (LLN) imply that, for any bounded $f$, $\mathcal{E}(Y_n(f)) \to \mathcal{E}(f)$, almost surely as $n \to \infty$. Indeed,

$$\begin{aligned} \mathcal{E}(Y_n(f)) &= \frac{n^2}{n(n-1)} \frac{1}{n^2} \sum_{i,j} \|f(x_i) - f(x_j)\|^2 \\ &= \frac{2n}{n-1} \left[ \frac{1}{n} \sum_i \|f(x_i)\|^2 - \left\| \frac{1}{n} \sum_i f(x_i) \right\|^2 \right] \\ &\to 2 \mathbb{E} \|f(x)\|^2 - 2 \|\mathbb{E} f(x)\|^2 = \mathcal{E}(f), \quad \text{almost surely as } n \to \infty, \end{aligned}$$

by the LLN applied to each term. Therefore, when $\varepsilon > 0$ is fixed, the second term in (16) tends to zero almost surely, and since $\varepsilon > 0$ is arbitrary, we conclude that

$$\sup_{f \in \mathcal{F}_1} \left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right| \to 0, \text{in probability, as } n \to \infty. \quad (17)$$

### 2.5 Large Deviations of the Sample Energy

To show an almost sure convergence in (17), we need to refine the bound on the supremum of the empirical process (12). For this, we apply Hoeffding's Inequality for U-statistics (Hoeffding, 1963), which is a special case of (de la Peña and Giné, 1999, Theorem 4.1.8).

**Lemma 2 (Hoeffding's Inequality for U-statistics)** *Let $\phi : M \times M \to \mathbb{R}$ be a bounded measurable map, and let $\{x_i : i \geq 1\}$ be a sequence of i.i.d. random variables with values in $M$. Assume that $\mathbb{E}[\phi(x_1,x_2)] = 0$ and that $b := \|\phi\|_\infty < \infty$, and let $\sigma^2 = \mathrm{Var}(\phi(x_1,x_2))$. Then, for all $t > 0$,*

$$\mathbb{P}\left[\frac{1}{n(n-1)}\sum_{1\leq i\neq j\leq n}\phi(x_i,x_j) > t\right] \leq \exp\left(-\frac{nt^2}{5\sigma^2+3bt}\right).$$

Let $f \in \mathcal{F}_1$. To bound the deviations of $\mathcal{E}(Y_n(f))$, we apply this result with $\phi(x,x') = \|f(x) - f(x')\|^2 - \mathcal{E}(f)$. Then,

$$\mathcal{E}(Y_n(f)) - \mathcal{E}(f) = \frac{1}{n(n-1)}\sum_{i\neq j}\phi(x_i,x_j).$$

By construction, $\mathbb{E}[\phi(x_1,x_2)] = 0$. Since $f$ is Lipschitz with constant 1, for any $x$ and $x'$ in $M$, $\|f(x) - f(x')\|^2 \leq \mathrm{diam}(M)^2$ and $\mathcal{E}(f) \leq \mathrm{diam}(M)^2$. Hence $\|\phi\|_\infty \leq \mathrm{diam}(M)^2$, and $\mathrm{Var}(\phi(x_1,x_2)) \leq \|\phi\|_\infty^2 \leq \mathrm{diam}(M)^4$. Applying Lemma 2 (twice), we deduce that, for any $\varepsilon > 0$,

$$\mathbb{P}\left(|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)| > \varepsilon\right) \leq 2\exp\left(-\frac{n\varepsilon^2}{5\,\mathrm{diam}(M)^4 + 3\,\mathrm{diam}(M)^2\varepsilon}\right). \tag{18}$$

Using (18) in (16), coupled with the union bound, we get that

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}_1}\left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right| > 9\varepsilon\,\mathrm{diam}(M)\right) \leq \mathcal{N}_\infty(\mathcal{F}_1^0,\varepsilon)\cdot 2\exp\left(-\frac{n\varepsilon^2}{5\,\mathrm{diam}(M)^2 + 3\varepsilon}\right). \tag{19}$$

Clearly, the RHS is summable for every $\varepsilon > 0$ fixed, so the convergence in (17) happens in fact with probability one, that is,

$$\sup_{f\in\mathcal{F}_1}\left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right| \to 0, \text{ almost surely, as } n \to \infty.$$

### 2.6 Convergence in Value: Proof of (5)

Assume $r_n$ satisfies the Connectivity requirement, and that $n$ is large enough that we have $\max(c(r_n), 6\lambda_n) < 1$. When $\Lambda(\lambda_n r_n)$ holds, by (13), we have

$$\left|\sup_{Y\in\mathcal{Y}_{n,r}}\mathcal{E}(Y) - \sup_{f\in\mathcal{F}_1}\mathcal{E}(f)\right| \leq (1+6\lambda_n)^2\sup_{f\in\mathcal{F}_1}\left|\mathcal{E}(Y_n(f)) - \mathcal{E}(f)\right| + 3\max\left(c(r_n), 6\lambda_n\right)\mathrm{diam}(M)^2,$$

while when $\Lambda(\lambda_n r_n)$ does not hold, since the energies are bounded by $\mathrm{diam}(M)^2$, we have

$$\left|\sup_{Y\in\mathcal{Y}_{n,r}}\mathcal{E}(Y) - \sup_{f\in\mathcal{F}_1}\mathcal{E}(f)\right| \leq 2\,\mathrm{diam}(M)^2.$$

Combining these inequalities, we deduce that

$$\begin{aligned}\Big|\sup_{Y\in\mathcal{Y}_{n,r}}\mathcal{E}(Y)-\sup_{f\in\mathcal{F}_1}\mathcal{E}(f)\Big| \;\leq\; & 3\max\big(c(r_n),6\lambda_n\big)\operatorname{diam}(M)^2\,\mathbb{I}_{\Lambda(\lambda_n r_n)}\\ & +2\operatorname{diam}(M)^2\,\mathbb{I}_{\Lambda(\lambda_n r_n)^{\mathsf{c}}}\\ & +(1+6\lambda_n)^2\sup_{f\in\mathcal{F}_1}\big|\mathcal{E}(Y_n(f))-\mathcal{E}(f)\big|. \qquad (20)\end{aligned}$$

Almost surely, the sum of the first two terms on the RHS tends to 0 by the fact that $c(r)\to 0$ when $r\to 0$, and (7) since $r_n$ satisfies the Connectivity requirement. The third term tends to 0 by (17). Hence, (5) is established.

### 2.7 Convergence in Solution: Proof of (6)

Assume $r_n$ satisfies the Connectivity requirement, and that $n$ is large enough that $\lambda_n\leq 1/2$. Let $\hat{Y}_n$ denote any solution of Discrete MVU. When $\Lambda(\lambda_n r_n)$ holds, there is $\hat{f}_n\in\mathcal{F}_{1+6\lambda_n}$ such that $\hat{Y}_n = Y_n(\hat{f}_n)$. Note that the existence of the interpolating function $\hat{f}_n$ holds on $\Lambda(\lambda_n r_n)$ for each fixed $n$, and that this does not imply the existence of an interpolating sequence $(\hat{f}_n)_{n\geq 1}$. That said, for each $\omega$ in the event $\liminf_n\Lambda(\lambda_n r_n)$, there exists a sequence $\hat{f}_n(.;\omega)$ and an integer $n_0(\omega)$ such that $\hat{Y}_n = Y_n(\hat{f}_n)$ for all $n\geq n_0(\omega)$, that is, the sequence is interpolating a solution of Discrete MVU for all $n$ large enough. In addition, when $r_n$ satisfies the Connectivity requirement, then with probability one, $\Lambda(\lambda_n r_n)^{\mathsf{c}}$ holds for only finitely many $n$'s by the Borel-Cantelli lemma, implying that, with probability one, $\Lambda(\lambda_n r_n)$ holds infinitely often.

In fact, without loss of generality, we may assume that $\hat{f}_n\in\mathcal{F}^0_{1+6\lambda_n}\subset\mathcal{F}^0_4$. Since $\mathcal{F}^0_4$ is equicontinuous and bounded, it is compact for the topology of the supnorm by the Arzelà-Ascoli Theorem. Hence, any subsequence of $\hat{f}_n$ admits a subsequence that converges in supnorm. And since $\mathcal{F}^0_L$ increases with $L$ and $\mathcal{F}^0_1=\cap_{L>1}\mathcal{F}^0_L$, any accumulation point of $(\hat{f}_n)$ is in $\mathcal{F}^0_1$.

In fact, if we define $\mathcal{S}^0_1=\mathcal{S}_1\cap\mathcal{F}^0_1$, then all the accumulation points of $(\hat{f}_n)$ are in $\mathcal{S}^0_1$. Indeed, we have

$$\mathcal{E}(\hat{f}_n)=\mathcal{E}(\hat{f}_n)-\mathcal{E}(Y_n(\hat{f}_n))+\mathcal{E}(Y_n(\hat{f}_n)),$$

with

$$\big|\mathcal{E}(\hat{f}_n)-\mathcal{E}(Y_n(\hat{f}_n))\big|\leq\sup_{f\in\mathcal{F}_1}\big|\mathcal{E}(Y_n(f))-\mathcal{E}(f)\big|\to 0,$$

by (17), and

$$\mathcal{E}(Y_n(\hat{f}_n))=\sup_{Y\in\mathcal{Y}_{n,r_n}}\mathcal{E}(Y)\to\sup_{f\in\mathcal{F}_1}\mathcal{E}(f),$$

by (5), almost surely as $n\to\infty$. Hence, if $f_\infty=\lim_k\hat{f}_{n_k}$, by continuity of $\mathcal{E}$ on $\mathcal{F}^0_4$, we have

$$\mathcal{E}(f_\infty)=\lim_k\mathcal{E}(\hat{f}_{n_k})=\sup_{f\in\mathcal{F}_1}\mathcal{E}(f),$$

and given that $f_\infty\in\mathcal{F}^0_1$, we have $f_\infty\in\mathcal{S}^0_1$ by definition.

The fact that $(\hat{f}_n)$ is compact with all accumulation points in $\mathcal{S}^0_1$ implies that

$$\inf_{f\in\mathcal{S}^0_1}\|\hat{f}_n-f\|_\infty\to 0, \qquad (21)$$

and since we have $\max_{1\leq i\leq n}\|\hat{y}_i-f(x_i)\|=\|\hat{f}_n(x_i)-f(x_i)\|\leq\|\hat{f}_n-f\|_\infty$, this immediately implies (6). The convergence in (21) is a consequence of the following simple result.

**Lemma 3** *Let $(a_n)$ be a sequence in a compact metric space with metric $\delta$, that has all its accumulation points in a set $A$. Then*

$$\inf_{a\in A} \delta(a_n, a) \to 0.$$

**Proof** If this is not the case, then there is $\varepsilon > 0$ such that, $\inf_{a\in A} \delta(a_n, a) \geq \varepsilon$ for infinitely many $n$'s, denoted $n_1 < n_2 < \cdots$. The space being compact, $(a_{n_k})$ has at least one accumulation point, which is in $A$ by assumption. However, by construction, $(a_{n_k})$ cannot have an accumulation point in $A$. This is a contradiction. ■

## 3. Quantitative Convergence Bounds

We obtained a general, qualitative convergence result for MVU in the preceding section and now specify some of the supporting arguments to obtain quantitative convergence speeds. This will require some (natural) additional assumptions on $\mu$ and $M$. While the proof of a result like Theorem 1 is necessarily complex, we endeavored in making it as transparent and simple as we could. The present section is more technical, and the reader might choose to first read Section 4 to learn about the solutions to Continuum MVU, which imply consistency (and inconsistencies) for MVU as a dimensionality-reduction algorithm.

We consider two specific types of sets $M$:

- *Thin sets.* $M$ is a $d$-dimensional compact, connected, $C^2$ submanifold with $C^2$ boundary (if nonempty). In addition, $M \subset M_\star$, where $M_\star$ is a $d$-dimensional, geodesically convex $C^2$ submanifold.

- *Thick sets.* $M$ is a compact, connected subset that is the closure of its interior and has a $C^2$ boundary.

The ambient space is $\mathbb{R}^p$. Note that our results are equally valid for piecewise smooth sets. Thin sets are a model for noiseless data, where that the data points are sampled from a submanifold. Note that they may have holes and boundaries. And thick sets are a model for noisy data, where that the data points are sampled from the vicinity of a submanifold.

An important example of thick sets are tubular neighborhoods of thin sets. For a set $A \subset \mathbb{R}^p$ and $\eta > 0$, the $\eta$-neighborhood of $A$ is the set of points in $\mathbb{R}^p$ within Euclidean distance $\eta$ of $A$, and is denoted $B(A,\eta)$. The reach of a set $A \subset \mathbb{R}^p$ is defined in Federer (1959) as the largest $\eta$ such that, for any $x \in B(A,\eta)$ there is a unique point $a \in A$ closest to $x$. We denote by $\rho(A)$ the reach of $A$. Note that any thin set $A$ has positive reach, which bounds its radius of curvature from below. While for any thick set $A$, $\partial A$ is a thin set without boundary, for any $\eta < \rho(A)$, $\bar{B}(A,\eta)$ is a thick set, with boundary having reach $\geq \rho(A) - \eta$.

In what follows, $C$ and $C_k$ denote constants that depend only on $p$ and $d$, which may change with each appearance.

### 3.1 The Regularity Condition

The first thing we do is specify the function $c$ in (3). When $M$ is a thin set, we define $r_M = \min\big(\rho(M_\star), \rho(\partial M)\big)$, where by convention $\rho(\emptyset) = \infty$. And when $M$ is a thick set, we let $r_M = \rho(\partial M)$.

The following result seems valid when $r_M = \rho(M)$ in both cases, but the proof seems much more involved.

**Lemma 4** *Whether $M$ is a thin or a thick set,* (3) *is valid with*

$$c(r) = \frac{4r}{r_M} \mathbb{I}_{\{r<r_M/2\}} + \mathbb{I}_{\{r \geq r_M/2\}}.$$

**Proof** We borrow results from Niyogi et al. (2008). Let $x, x' \in M$ such that $\|x - x'\| \leq r_M/2$.

First, suppose that $M$ is thick. Consider the (straight) line segment joining these two points. If this segment is included in $M$, then $\delta_M(x,x') = \|x - x'\|$. Otherwise, it intersects $\partial M$ in at least two points; among these points, let $z$ be the closest to $x$ and $z'$ the closest to $x'$. Since $\partial M$ has no boundary, it is geodesically convex, so that there is a geodesic on $\partial M$, denoted $\xi$, joining $z$ and $z'$. Niyogi et al. (2008, Prp. 6.3) applies since $\|z - z'\| \leq \|x - x'\| \leq r_M/2 \leq \rho(\partial M)/2$, and $\rho(\partial M)$ coincides with the condition number of $\partial M$ as defined in Niyogi et al. (2008)—and denoted by $\tau$ there. Hence, if $\ell$ is the length of $\xi$, we have

$$\ell \leq \rho(\partial M) - \rho(\partial M)\sqrt{1 - \frac{2\|z - z'\|}{\rho(\partial M)}} \leq \|z - z'\| + 4\|z - z'\|^2/r_M, \tag{22}$$

using the fact that $\sqrt{1-t} \geq 1 - t/2 - t^2$ for all $t \in [0,1]$ and $r_M \leq \rho(\partial M)$. Let $\gamma$ be the path made of $\xi$ concatenated with the segments $[xz]$ and $[z'x']$. If $L$ is the length of $\gamma$, we have

$$\begin{aligned} L &= \|x - z\| + \|z' - x'\| + \ell \\ &\leq \|x - z\| + \|z' - x'\| + \|z - z'\| + 4\|z - z'\|^2/r_M \\ &\leq \|x - x'\| + 4\|x - x'\|^2/r_M, \end{aligned}$$

using the fact that $x, z, z', x'$ are in that order on the line segment joining $x$ and $x'$. This concludes the proof when $M$ is thick.

When $M$ is thin, we distinguish two cases. Either there is a geodesic joining $x$ and $x'$, and Niyogi et al. (2008, Prp. 6.3) is directly applicable. Otherwise, $M$ is not geodesically convex. Let $\gamma_\star$ be a geodesic on $M_\star$ joining $x$ and $x'$. Necessarily, it hits the boundary $\partial M$ in at least two points. Let $z$, $z'$, $\xi$ and $\ell$ be defined as before. We again have (22). Let $(xz)_\star$ and $(z'x')_\star$ denote the arcs along $\gamma_\star$ joining $x$ and $z$, and $z'$ and $x'$, respectively. Applying Niyogi et al. (2008, Prp. 6.3) to each arc, which is possible since $r_M \leq \rho(M_\star)$, we also have

$$\mathrm{length}((xz)_\star) \leq \|x - z\| + 4\|x - z\|^2/r_M, \qquad \mathrm{length}((z'x')_\star) \leq \|z' - x'\| + 4\|z' - x'\|^2/r_M.$$

Let $\gamma$ be the curve made of concatenating these two arcs and $\xi$, and let $L$ denote its length. We have

$$\begin{aligned} L &= \mathrm{length}((xz)_\star) + \mathrm{length}((z'x')_\star) + \ell \\ &\leq \|x - z\| + \frac{4\|x - z\|^2}{r_M} + \|z' - x'\| + \frac{4\|z' - x'\|^2}{r_M} + \|z - z'\| + \frac{4\|z - z'\|^2}{r_M} \\ &\leq \|x - x'\| + \frac{4\|x - x'\|^2}{r_M}. \end{aligned}$$

This concludes the proof when $M$ is thin. ■

### 3.2 Covering Numbers and a Bound on the Neighborhood Radius

At what speed can we have $r_n \to 0$ and still have (7) hold? This question is of practical importance, since the neighborhood radius may affect the output of MVU in a substantial way. Computationally, it is preferable to have $r_n$ small, so there are fewer constraints in (1). However, we already explained that $r_n$ needs to be large enough that, at the very minimum, the resulting neighborhood graph is connected. In fact, we required the stronger condition (7).

To keep the exposition simple, we assume that $\mu$ is comparable to the uniform distribution on $M$, that is, we assume that there is a constant $\alpha > 0$ such that

$$\mu(B(x,\eta)) \geq \alpha \operatorname{vol}_d(B(x,\eta) \cap M), \quad \forall x \in M, \forall \eta > 0, \tag{23}$$

where $\operatorname{vol}_d$ denotes the $d$-dimensional Hausdorff measure and $d$ denotes the Hausdorff dimension of $M$. We need the following result. Let $\omega_d$ be the volume of the $d$-dimensional unit ball.

**Lemma 5** *Whether $M$ is thin or thick, there is $C > 0$ such that, for any $\eta \leq r_M$ and any $x \in M$,*

$$\operatorname{vol}_d(B(x,\eta) \cap M) \geq C\eta^d.$$

**Proof** It suffices to prove the result for $x \in M \setminus \partial M$ and for $\eta$ small enough.

*Thick set.* We first assume that $M$ is thick. Take $x \in M$ and $\eta < r_M$. If $\operatorname{dist}(x, \partial M) \geq \eta$, then $B(x,\eta) \subset M$ and the result follows immediately. Otherwise, let $u$ be the metric projection of $x$ onto $\partial M$, and define $z = x + (\eta/4)(x-u)/\|x-u\|$. By the triangle inequality, $B(z,\eta/4) \subset B(x,\eta)$. Also, by Federer (1959, Theorem 4.8), $u$ is also the metric projection of $z \in M$ onto $\partial M$, so that $\operatorname{dist}(z,\partial M) = \|z-u\| = \|x-u\| + \eta/4 > \eta/4$. And, necessarily, $z \in M$, for otherwise the line segment joining $z$ to $x$ would intersect $\partial M$, and any point on that intersection would be closer to $z$ than $u$ is, which cannot be. Therefore, $B(z,\eta/4) \subset B(x,\eta) \cap M$ and the result follows immediately.

*Thin set.* We now assume that $M$ is thin. For $y \in M$, let $T_y$ be the tangent subspace of $M$ at $y$ and let $\pi_y$ denote the orthogonal projection onto $T_y$. Because $M$ is a $C^2$ submanifold, for every $y \in M$, there is $\varepsilon_y > 0$ such that $\pi_y$ is a $C^2$ diffeomorphism on $K_y := B(y,\varepsilon_y) \cap M$, with $\pi_y^{-1}$ being 2-Lipschitz on $\pi_y(K_y)$—the latter comes from the fact that $D_y\pi_y$ is the identity map and $z \to D_z\pi_y$ is continuous. Since $M$ is compact, there is $y_1, \ldots, y_m \in M$, with $m < \infty$, such that $M \subset \cup_j B(y_j, \varepsilon_j/2)$. Let $\varepsilon = \min_j \varepsilon_{y_j}$, which is strictly positive. Let $y$ be among the $y_j$'s such that $x \in B(y,\varepsilon_j/2)$. Assuming that $\eta < \varepsilon/2$, we have that $B(x,\eta) \subset B(y,\varepsilon_j)$. Let $U := B(y,\varepsilon_j)$, $K = K_y$, $T = T_y$ and $\pi = \pi_y$ for short.

We first show that, if $\partial M \cap K \neq \emptyset$ and $W := \pi(\partial M \cap K)$, then $\rho(W) \geq \rho(\partial M)$. Indeed, for any $z, z' \in K$, we have

$$\operatorname{dist}(\pi(z') - \pi(z), \operatorname{Tan}(W, \pi(z))) \leq \operatorname{dist}(z'-z, \operatorname{Tan}(\partial M, z)) \leq \frac{1}{2\rho(\partial M)} \|z'-z\|^2,$$

where the first inequality follows from the facts that $\operatorname{Tan}(W, \pi(z)) = \pi(\operatorname{Tan}(\partial M, z))$ and that $\pi$ is 1-Lipschitz, and the second inequality from Federer (1959, Theorem 4.18) applied to $\partial M$. In turn, Federer (1959, Theorem 4.17) applied to $W$ implies that $\rho(W) \geq \rho(\partial M)$.

We can now reason as we did for thick sets, but with a twist. To be sure, let $a = \pi(x)$ and notice that $B(a,\eta) \cap T = \pi(B(x,\eta)) \subset \pi(U)$ since $B(x,\eta) \subset U$. If $\operatorname{dist}(a,W) \geq \eta/2$, $B(a,\eta/2) \cap T \subset \pi(K)$. If $\operatorname{dist}(a,W) < \eta/2$, let $b$ be the metric projection of $a$ onto $W$ and define $c = a + (\eta/8)(a-b)/\|a-b\|$. Arguing exactly as we did for thick sets, we have that $B(c,\eta/8) \cap T \subset B(a,\eta/2) \cap \pi(K)$. Let

$L = \pi^{-1}(B(c,\eta/8) \cap T)$. Note that $L \subset \pi^{-1}(B(a,\eta/2) \cap T) \cap K \subset B(x,\eta) \cap K \subset B(x,\eta) \cap M$, since $\pi$ is injective on $K$ and $\pi^{-1}$ is 2-Lipschitz on $\pi(K)$. In addition, since $\pi$ is 1-Lipschitz on $K$, we have $\mathrm{vol}_d(L) \geq \mathrm{vol}_d(\pi(L)) = \mathrm{vol}_d(B(c,\eta/8) \cap T)$. This immediately implies the result. ■

When (23) is satisfied, and $M$ is either thin or thick, we can provide sharp rates for $r_n$. Just as we did in Section 2.1, we work with coverings of $M$. Let $\mathcal{N}(M,\eta)$ denote the cardinality of a minimal $\eta$-covering of $M$ for the Euclidean norm.

**Lemma 6** *Suppose $\eta \leq r_M$. When $M$ is thick,*

$$\mathcal{N}(M,\eta) \leq C \,\mathrm{vol}_p(M) \eta^{-p};$$

*and when $M$ is thin and $0 \leq \sigma < \rho(M)$,*

$$\mathcal{N}(B(M,\sigma),\eta) \leq C \,\mathrm{vol}_d(M) \max(\sigma,\eta)^{p-d} \eta^{-p}.$$

*The constant $C$ depends only on $p$ and $d$.*

**Proof** Suppose $M$ is thick and let $z_1,\ldots,z_{N_\eta}$ an $\eta$-packing of $M$ of size $N_\eta := \mathcal{N}(M,\eta)$. Since $B(z_i,\eta/2) \cap B(z_j,\eta/2) = \emptyset$ when $i \neq j$, we have

$$\mathrm{vol}_p(M) \geq \sum_j \mathrm{vol}_p(B(z_j,\eta/2) \cap M) \geq N_\eta C_p \eta^p,$$

where $C_p$ is the constant in Lemma 5. The bound on $N_\eta$ follows.

Suppose $M$ is thin. When $\sigma \leq \eta/4$, let $z_1,\ldots,z_{N_{\eta/4}}$ an $(\eta/4)$-packing of $M$. Then by the triangle inequality, $B(M,\sigma) \subset \cup_j B(z_j,\eta/2)$, and therefore $\mathcal{N}(B(M,\sigma),\eta) \leq N_{\eta/4}$. When $\sigma \geq \eta/4$, let $z_1,\ldots,z_N$ be an $(\eta/4)$-packing of $B(M,\sigma-\eta/4)$. Since $B(z_i,\eta/8) \cap B(z_j,\eta/8) = \emptyset$ when $i \neq j$, and $B(z_i,\eta/8) \subset B(M,\sigma)$, we have

$$\mathrm{vol}_p(B(M,\sigma)) \geq \sum_j \mathrm{vol}_p(B(z_j,\eta/8)) = N \omega_p (\eta/8)^p.$$

Hence, $N \leq \omega_p^{-1}(\eta/8)^{-p} \,\mathrm{vol}_p(B(M,\sigma))$. By Weyl's volume formula for tubes (Weyl, 1939), we have $\mathrm{vol}_p(B(M,\sigma)) \leq C_1 \,\mathrm{vol}_d(M) \sigma^{p-d}$ for a constant $C_1$ depending on $p$ and $d$. Then the result follows from the fact that, by the triangle inequality, $B(M,\sigma) \subset \cup_j B(z_j,\eta/2)$, so that $\mathcal{N}(B(M,\sigma),\eta) \leq N$. ■

We are now ready to take a closer look at (7). Let $\eta_n$ be defined as in Section 2.1. By (23) and Lemma 5, we have $p_\eta \geq C_1 \alpha \eta^d$, and we have $\mathcal{N}(M,\eta) \leq C_2 \eta^{-d}$ by Lemma 6, where $C_1$ and $C_2$ depend only on $M$. Hence,

$$\mathcal{N}(M,\eta)(1-p_\eta)^n \leq C_2 \eta^{-d} \big(1 - C_1 \alpha \eta^d\big)^n \leq C_2 \eta^{-d} e^{-n C_1 \alpha \eta^d} \leq \frac{1}{n^2},$$

when

$$\eta^d \geq (C_1 \alpha n)^{-1} \log\big(C_2 \eta^{-d} n^2\big).$$

We deduce that any $r_n \gg r_n^\dagger := (\log(n)/n)^{1/d}$ satisfies (7) with any $\lambda_n \to 0$ such that $\lambda_n \gg r_n^\dagger / r_n$.

### 3.3 Packing Numbers of Lipschitz Functions on $M$

It appears necessary to provide a bound for $\mathcal{N}_\infty(\mathcal{F}_1^0,\eta)$. For this, we follow the seminal work of Kolmogorov and Tikhomirov (1961) on entropy bounds for classical functions classes (including Lipschitz classes). We provide details for completeness.

**Lemma 7** *For any $M$ compact, connected subset of $\mathbb{R}^p$ satisfying* (3)*, there is a constant $C$ such that*

$$\log \mathcal{N}_\infty(\mathcal{F}_1^0,\eta) \leq C\,(\log(1/\eta)+\mathcal{N}(M,\eta/C)),$$

*for all $0<\eta\leq 1$.*

**Proof** Take $0<\varepsilon\leq 1/\sqrt{p}$ and let $C_0=2\sqrt{p}(2+c(2))$. For $j=(j_1,\ldots,j_p)\in\mathbb{Z}^p$, let $Q_j=\prod_{s=1}^p[j_s\varepsilon,(j_s+1)\varepsilon)$. Let $J=\{j: Q_j\cap M\neq\emptyset\}$, which we see as a subgraph of the lattice for the $2^p$-nearest neighbor topology.

Note that $|J|\leq C_1\mathcal{N}(M,\varepsilon)$. Indeed, let $e_1,\ldots,e_{2^p}$ be the vertices of the unit hypercube of $\mathbb{R}^p$ and let $Z_s=e_s+(2\mathbb{Z})^p$. Also, let $Z_0=(2\mathbb{Z})^p$. By construction, $Z_1,\ldots,Z_{2^p}$ is a partition of $\mathbb{Z}^p$. Therefore, there is $s$ (say $s=1$) such that $|J\cap Z_s|\geq|J|/2^p$. For each $j\in J\cap Z_1$, pick $x_j\in Q_j\cap M$. By construction, for any $j\neq j'$ both in $J\cap Z_1$, $\|x_j-x_{j'}\|>2\varepsilon$, so $|J\cap Z_1|$ is smaller than the $2\varepsilon$-packing number of $M$, which is smaller than the $\varepsilon$-covering number of $M$.

Note also that $\cup_j Q_j$ is connected because $M$ is. Let $\pi_1,\ldots,\pi_\ell$ be a sequence covering $J$ and such that $Q_{\pi_s}$ and $Q_{\pi_{s-1}}$ are adjacent. A depth-first construction gives a sequence $\pi$ of length at most $\ell\leq C_2|J|$, since each $Q_j$ has a constant number ($=2^p$) of adjacent hypercubes.

Let $y_1,\ldots,y_m$ be an enumeration of the $\varepsilon$-grid $(\varepsilon\mathbb{Z}\cap[-\mathrm{diam}(M),\mathrm{diam}(M)])^p$. Note that $m\leq C_3\varepsilon^{-p}$ and that, for each $s$ there are at most $C_4$ indices $t$ such that $\|y_s-y_t\|\leq C_0\varepsilon$.

Consider the class $\mathcal{G}$ of piecewise-constant functions $g:M\to\mathbb{R}^p$ of the form $g(x)=y_{t_j}$ for all $x\in Q_j\cap M$ and such that $\|y_{t_j}-y_{t_k}\|\leq C_0\varepsilon$ when $Q_j$ and $Q_k$ are adjacent. This is a subclass of the class of functions of the form $g(x)=y_{t_{\pi(j)}}$ for all $x\in Q_{\pi(j)}$ and such that $\|y_{t_{\pi(j)}}-y_{t_{\pi(j-1)}}\|\leq C_0\varepsilon$. The cardinality of the larger class is at most $mC_4^{\ell-1}$, since there are $m$ possible values for $y_{t_{\pi(1)}}$ and then, at each step along $\pi$, there at most $C_4$ choices. Therefore,

$$\begin{aligned}\log|\mathcal{G}| &\leq \log m+\ell\log C_4\\ &\leq \log(C_3)+p\log(1/\varepsilon)+C_2C_1\mathcal{N}(M,\varepsilon)\log(C_4)\\ &\leq C_5(\log(1/\varepsilon)+\mathcal{N}(M,\varepsilon)).\end{aligned}$$

For each $j$, choose $z_j\in Q_j\cap M$. Take any $f\in\mathcal{F}_1^0$. For each $j$, let $t_j$ be such that $\|f(z_j)-y_{t_j}\|\leq\sqrt{p}\varepsilon$ and let $g$ be defined by $g(x)=y_{t_j}$ for all $x\in Q_j$. Suppose $Q_j$ and $Q_k$ are adjacent, so that $\|z_j-z_k\|\leq 2\sqrt{p}\varepsilon\leq 2$. By the triangle inequality, (2) and (3), we have

$$\begin{aligned}\|y_{t_j}-y_{t_k}\| &\leq \|f(z_j)-f(z_k)\|+\|y_{t_j}-f(z_j)\|+\|y_{t_k}-f(z_k)\|\\ &\leq (1+c(\|z_j-z_k\|))\|z_j-z_k\|+\sqrt{p}\varepsilon+\sqrt{p}\varepsilon\\ &\leq (1+c(2))2\sqrt{p}\varepsilon+2\sqrt{p}\varepsilon\\ &= C_0\varepsilon.\end{aligned}$$

so that $g\in\mathcal{G}$. Moreover, for $x\in Q_j\cap M$,

$$\|g(x)-f(x)\|=\|y_{t_j}-f(z_j)\|+\|f(z_j)-f(x)\|\leq\sqrt{p}\varepsilon+(1+c(\sqrt{p}\varepsilon))\sqrt{p}\varepsilon\leq(2+c(1))\sqrt{p}\varepsilon.$$

The result follows from choosing $\varepsilon = \eta/((2+c(1))\sqrt{p})$. ■

In particular, if $M$ is thin or thick, we have

$$\log \mathcal{N}_\infty(\mathcal{F}_1^0, \eta) \leq C\eta^{-d},$$

by Lemma 6 (applied with $\sigma = 0$ with $M$ is thin) and Lemma 7. Recall that $d$ is the intrinsic dimension of $M$.

### 3.4 Quantitative Convergence Bound

From (19) and Lemma 7, there is a constant $C > 0$ such that

$$\mathbb{P}\left( \sup_{f \in \mathcal{F}_1} \left| \mathcal{E}(Y_n(f)) - \mathcal{E}(f) \right| > Cn^{-1/(d+2)} \right) \leq \exp(-n^{-d/(d+2)}).$$

Using this fact in (20), together with Lemma 4 and the order of magnitude for $r_n$ derived in Section 3.2, leads to a bound on the rate of convergence in (5) via the Borel-Cantelli Lemma.

**Theorem 2** *Suppose that $M$ is either thin or thick, of dimension $d$, and that* (23) *holds. Assume that $r_n \to 0$ such that $r_n \gg r_n^\dagger := (\log(n)/(\alpha n))^{1/d}$ and take any $a_n \to \infty$. Then, with probability one,*

$$\left| \sup\{\mathcal{E}(Y) : Y \in \mathcal{Y}_{n,r_n}\} - \sup\{\mathcal{E}(f) : f \in \mathcal{F}_1\} \right| \leq a_n\big(r_n + \frac{r_n^\dagger}{r_n} + n^{-1/(d+2)}\big),$$

*for n large enough.*

We speculate that this convergence rate is not sharp and that the first term in brackets can be replaced by $r_n^2$. Indeed, we believe the result (Niyogi et al., 2008, Prop. 6.3) for approximating geodesics distances is not rate-optimal, leading to a loose Lemma 4, while we anticipate that, in fact, $c(r) = O(r^2)$.

Unfortunately, we do not have a quantitative bound on the rate of convergence of the solutions in (6).

## 4. Continuum MVU

Now that we established the convergence of Discrete MVU to Continuum MVU, we study the latter, and in particular its solutions. We mostly focus on the case where $M$ is isometric to a Euclidean domain.

*Isometry assumption.* We assume that $M$ is isometric to a compact, connected domain $D \subset \mathbb{R}^d$. Specifically, there is a bijection $\psi : M \to D$ satisfying $\delta_D(\psi(x), \psi(x')) = \delta_M(x, x')$ for all $x, x' \in M$.

As a glimpse of the complexity of the notion of isometry, and also for further reference, consider a domain $D$ as above. Then the canonical inclusion $\iota$ of $D$ in $\mathbb{R}^d$ is not necessarily an isometry between the metric spaces $(D, \delta_D)$ and $(\mathbb{R}^d, \|\cdot\|)$. To see this, let $x$ and $x'$ be two points of $D$. Let $\gamma$ be a shortest path connecting $x$ to $x'$ in $D$. Suppose that $\iota : (D, \delta_D) \to (\mathbb{R}^d, \|\cdot\|)$ is an isometry. Then, $L(\iota \circ \gamma) = L(\gamma) = \delta_D(x, x') = \|\iota(x) - \iota(x')\|$. So the image path $\iota \circ \gamma$ is a shortest path connecting $\iota(x)$ to $\iota(x')$, hence a segment. Since this segment lies in $\iota(D) = D$, and since this holds for any pair of

points $x, x'$ in $D$, this implies that $D$ is convex. Conversely, if $D$ is convex, the canonical inclusion $\iota$ is an isometry.

We start by showing that, in the case where $M$ is isometric to a convex domain, then MVU recovers this convex domain modulo a rigid transformation, so that MVU is consistent is that case. The last part of the section is dedicated to a perturbation analysis that shows two things. First, that Continuum MVU changes slowly with the amount of noise, up to a point. And second, that when $M$ is isometric to a domain that is not convex, MVU may not recover this domain. We provide some illustrative examples of that.

In the following, we identify $\mathbb{R}^d$ with $\mathbb{R}^d \times \{0\}^{p-d} \subset \mathbb{R}^p$.

### 4.1 Consistency under the Convex Assumption

If we assume that $D$ is convex, then MVU recovers $D$ up to a rigid transformation, in the following sense. Recall that $\mathcal{S}_1$ is the solution space of Continuum MVU.

**Theorem 3** *Suppose that $M$ is isometric to a convex subset $D \subset \mathbb{R}^d$ with isometry mapping $\psi : M \to D$, and that* (23) *holds. Then*

$$\mathcal{S}_1 = \{\zeta \circ \psi : \zeta \in \mathrm{Isom}(\mathbb{R}^p)\}.$$

**Proof** Note first that, since $D$ is convex, its intrinsic distance coincides with the Euclidean distance of $\mathbb{R}^d$, that is, $\delta_D = \|\cdot\|$. For all $f$ in $\mathcal{F}_1$, we have

$$\begin{aligned}
\mathcal{E}(f) &= \int_{M\times M} \|f(x) - f(x')\|^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \\
&\leq \int_{M\times M} \delta_M(x,x')^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \\
&= \int_{M\times M} \delta_D(\psi(x),\psi(x'))^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \\
&= \int_{M\times M} \|\psi(x) - \psi(x')\|^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \\
&= \int_{D\times D} \|z - z'\|^2 (\mu\circ\psi^{-1})(\mathrm{d}z)(\mu\circ\psi^{-1})(\mathrm{d}z'),
\end{aligned}$$

while

$$\mathcal{E}(\psi) = \int_{D\times D} \|z - z'\|^2 (\mu\circ\psi^{-1})(\mathrm{d}z)(\mu\circ\psi^{-1})(\mathrm{d}z').$$

So

$$\sup_{f\in\mathcal{F}_1} \mathcal{E}(f) = \mathcal{E}(\psi) = \int_{D\times D} \|z - z'\|^2 (\mu\circ\psi^{-1})(\mathrm{d}z)(\mu\circ\psi^{-1})(\mathrm{d}z').$$

Hence $\psi \in \mathcal{S}_1$, and since $\mathcal{E}(\zeta\circ\psi) = \mathcal{E}(\psi)$ for any isometry $\zeta : \mathbb{R}^p \to \mathbb{R}^p$,

$$\{\zeta \circ \psi : \zeta \in \mathrm{Isom}(\mathbb{R}^p)\} \subset \mathcal{S}_1.$$

Now let $f : M \to \mathbb{R}^p$ be a function in $\mathcal{F}_1$ so that $\|f(x) - f(x')\| \leq \delta_M(x,x')$ for any points $x$ and $x'$ in $M$. Suppose that $f$ is not an isometry. Then there exists two points $x$ and $x'$ in $M$ such that

$$\|f(x) - f(x')\| < \delta_M(x,x').$$

By continuity of $f$, there exists a nonempty open subset $U$ of $M \times M$ containing $(x,x')$ such that $\|f(z)-f(z')\| < \delta_M(z,z')$ for all $(z,z')$ in $U$. In addition, $\mu(U) > 0$ by (23). Consequently

$$\begin{aligned}\mathcal{E}(f) &= \int_{M\times M\setminus U} \|f(x)-f(x')\|^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') + \int_U \|f(x)-f(x')\|^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \\ &< \int_{M\times M} \delta_M(x,x')^2 \mu(\mathrm{d}x)\mu(\mathrm{d}x') \\ &= \sup_{f\in\mathcal{F}_1} \mathcal{E}(f).\end{aligned}$$

So any function $f$ in $\mathcal{F}_1$ which is not an isometry onto its image does not belong to $\mathcal{S}_1$.

At last, since for any isometry $f$ in $\mathcal{S}_1$, the map $f\circ\psi^{-1} : \mathbb{R}^p \to \mathbb{R}^p$ is an isometry, there exists some isometry $\zeta \in \mathrm{Isom}(\mathbb{R}^p)$ such that $f = \zeta\circ\psi$, and we conclude that

$$\{\zeta\circ\psi : \zeta \in \mathrm{Isom}(\mathbb{R}^p)\} = \mathcal{S}_1.$$

■

In conclusion, MVU recovers the isometry when the domain $D$ is convex. Note that this is also the case of Isomap.

### 4.2 Noisy Setting

When the setting is noisy, with noise level $\sigma \geq 0$, $x_1,\ldots,x_n$ are sampled from $\mu_\sigma$, a (Borel) probability distribution on $\mathbb{R}^p$ with support $M_\sigma := \bar{B}(M,\sigma)$, that is, $M_\sigma$ is composed of all the points of $\mathbb{R}^p$ that are at a distance at most $\sigma$ from $M$. To speak of noise stability, we assume that $\mu_\sigma$ converges weakly when $\sigma \to 0$. Let $\mathcal{F}_{1,\sigma}$ denote the class of 1-Lipschitz functions on $M_\sigma$, and so on. Our simple perturbation analysis is plainly based on the fact that $\mathcal{E}$ is continuous with respect to the noise level, in the following sense. This immediately implies that MVU is tolerant to noise.

**Lemma 8** *Let $M \subset \mathbb{R}^p$ be of positive reach $\rho(M) > 0$ and assume that $\mu_\sigma \to \mu_0$ weakly when $\sigma \to 0$. Then as $\sigma \to 0$, we have*

$$\sup_{f\in\mathcal{F}_{1,\sigma}} \mathcal{E}_\sigma(f) \to \sup_{f\in\mathcal{F}_1} \mathcal{E}(f), \tag{24}$$

*and*

$$\sup_{f\in\mathcal{S}_{1,\sigma}} \inf_{g\in\mathcal{S}_1} \sup_{x\in M_\sigma} \inf_{z\in M} \|f(x)-g(z)\| \to 0. \tag{25}$$

**Proof** The metric projection $\pi : B(M,\rho(M)) \to M$ with $\pi(x) = \arg\min\{\|x-x'\| : x' \in M\}$, is well-defined and 1-Lipschitz (Federer, 1959, Theorem 4.8).

Consider any sequence $\sigma_m \to 0$ with $\sigma_m < \rho(M)$ for all $m \geq 1$, and let $f_m \in \mathcal{S}^0_{1,\sigma_m}$. Let $g_m$ denote the restriction of $f_m$ to $M$. Since $(g_m) \subset \mathcal{F}^0_1$ and $\mathcal{F}^0_1$ is compact for the supnorm, it admits a convergent subsequence. Assume $(g_m)$ itself is convergent, without loss of generality. Then $g_m \to g_\star$, with $g_\star \in \mathcal{F}^0_1$. For $x \in B(M,\rho(M))$, define $f_\star(x) = g_\star(\pi(x))$. Then for $x \in M_{\sigma_m}$, we have

$$\begin{aligned}\|f_\star(x)-f_m(x)\| &\leq \|g_\star(\pi(x))-g_m(\pi(x))\| + \|f_m(\pi(x))-f_m(x)\| \\ &\leq \|g_\star-g_m\|_\infty + \|\pi(x)-x\| \\ &\leq \|g_\star-g_m\|_\infty + \sigma_m,\end{aligned}$$

since $f_m \in \mathcal{F}_{1,\sigma_m}$ and the segment $[\pi(x),x] \subset M_{\sigma_m}$. The latter is due to $\|\pi(x)-x\| \leq \sigma_m$ and $B(\pi(x),\sigma_m) \subset M_{\sigma_m}$, both by definition. Hence, as functions on $M_{\sigma_m}$, we have $\|f_\star(x)-f_m(x)\|_\infty \to 0$, that is,

$$\sup_{x\in M_{\sigma_m}} \|f_\star(x)-f_m(x)\| \to 0.$$

By (14), again applied to functions on $M_{\sigma_m}$ for a fixed $m$, we have

$$\begin{aligned}\left|\mathcal{E}_{\sigma_m}(f_m)-\mathcal{E}_{\sigma_m}(f_\star)\right| &\leq 4\|f_\star(x)-f_m(x)\|_\infty \mathrm{diam}(M_{\sigma_m}) \\ &\leq 4\|f_\star(x)-f_m(x)\|_\infty \mathrm{diam}(B(M,\rho(M))) \\ &\to 0,\end{aligned}$$

and since $f_\star$ does not depend on $m$ and is bounded, we also have

$$\mathcal{E}_{\sigma_m}(f_\star) \to \mathcal{E}(f_\star) = \mathcal{E}(g_\star) \leq \sup_{\mathcal{F}_1} \mathcal{E}. \tag{26}$$

Hence

$$\begin{aligned}\sup_{\mathcal{F}_{1,\sigma_m}} \mathcal{E}_{\sigma_m} &= \mathcal{E}_{\sigma_m}(f_m) \\ &= \mathcal{E}(f_\star)+\mathcal{E}_{\sigma_m}(f_\star)-\mathcal{E}(f_\star)+\mathcal{E}_{\sigma_m}(f_m)-\mathcal{E}_{\sigma_m}(f_\star) \\ &\leq \sup_{\mathcal{F}_1}\mathcal{E}+\mathcal{E}_{\sigma_m}(f_\star)-\mathcal{E}(f_\star)+\mathcal{E}_{\sigma_m}(f_m)-\mathcal{E}_{\sigma_m}(f_\star),\end{aligned}$$

and we deduce that

$$\overline{\lim_{m\to\infty}} \sup_{\mathcal{F}_{1,\sigma_m}} \mathcal{E}_{\sigma_m} \leq \sup_{\mathcal{F}_1}\mathcal{E},$$

and since this is true for all sequences $\sigma_m \to 0$ (and $m$ large enough), we have

$$\overline{\lim_{\sigma\to 0}} \sup_{\mathcal{F}_{1,\sigma}} \mathcal{E}_\sigma \leq \sup_{\mathcal{F}_1}\mathcal{E}.$$

For the reverse relation, choose $g \in \mathcal{S}_1$ and for $x \in B(M,\rho(M))$ define $f(x) = g(\pi(x))$. As above, let $\sigma_m \to 0$ with $\sigma_m \leq \rho(M)$. Then $f \in \mathcal{F}_{1,\sigma_m}$ by composition, so that

$$\mathcal{E}_{\sigma_m}(f) \leq \sup_{\mathcal{F}_{1,\sigma_m}} \mathcal{E}_{\sigma_m}.$$

On the other hand,

$$\mathcal{E}_{\sigma_m}(f) \to \mathcal{E}(f) = \mathcal{E}(g) = \sup_{\mathcal{F}_1}\mathcal{E}.$$

Hence,

$$\sup_{\mathcal{F}_1}\mathcal{E} \leq \overline{\lim_{\sigma\to 0}} \sup_{\mathcal{F}_{1,\sigma}} \mathcal{E}_\sigma.$$

This concludes the proof of (24).

Equation (25) is now proved based on (24) in the same way (6) is proved based on (5), by contradiction. To be sure, assume (25) is not true. Then it is also not true for $\mathcal{S}^0_{1,\sigma}$ and $\mathcal{S}^0_1$. Hence, there is $\varepsilon > 0$, a sequence $\sigma_m \to 0$ and $f_m \in \mathcal{S}^0_{1,\sigma_m}$ such that

$$\inf_{g\in\mathcal{S}^0_1} \sup_{x\in M_{\sigma_m}} \inf_{z\in M} \|f_m(x)-g(z)\| \geq \varepsilon,$$

for infinitely many $m$'s. Without loss of generality, we assume this is true for all $m$. For each $m$, let $g_m$ be the restriction of $f_m$ to $M$. Then, taking a subsequence if needed, $g_m \to g_\star \in \mathcal{F}_1^0$ in supnorm. As before, define $f_\star(x) = g_\star(\pi(x))$ for $x \in B(M, \rho(M))$. Following the same arguments, we have

$$\sup_{x \in M_{\sigma_m}} \|f_\star(x) - f_m(x)\| \to 0.$$

We also see that, necessarily, $g_\star \in S_1^0$, for otherwise the inequality in (26) would be strict and this would imply that (24) does not hold. Hence

$$\sup_{x \in M_{\sigma_m}} \|f_\star(x) - f_m(x)\| \geq \sup_{x \in M_{\sigma_m}} \inf_{z \in M} \|f_m(x) - g_\star(z)\| \geq \inf_{g \in S_1^0} \sup_{x \in M_{\sigma_m}} \inf_{z \in M} \|f_m(x) - g(z)\|.$$

This leads to a contradiction. Hence the proof of (25) is complete. ■

### 4.3 Inconsistencies

We provide two emblematic situations where MVU fails to recover $D$. They are both consequences of MVU's robustness to noise. In both cases, we consider the simplest situation where $M = D \subset \mathbb{R}^2$ and $\mu$ is the uniform distribution. Note that $\psi$ is the identity function in this case, that is, $\psi(x) = x$, and the Isometry Assumption is clearly satisfied. We use the same notation as in Section 4.2 and let $\mu_\sigma$ denote the uniform distribution on $M_\sigma$. *Nonconvex without holes.* Suppose $M_0 \subset \mathbb{R}^2$ is a curve homeomorphic to a line segment, but different from a line segment, and for $\sigma > 0$, let $M_\sigma$ be the (closed) $\sigma$-neighborhood of $M_0$. We show that there is a numeric constant $\sigma_0 > 0$ such that, when $\sigma < \sigma_0$, $\psi$ does not maximize the energy $\mathcal{E}_\sigma$. To see this, we use Lemma 8 to assert that $\mathcal{S}_{1,\sigma} \to \mathcal{S}_{1,0}$ in the sense of (25), and that $\psi \notin \mathcal{S}_{1,0}$, because $\mathcal{S}_{1,0}$ is made of all the functions that map $M$ to a line segment isometrically. So there is $\sigma_0 > 0$ such that $\psi \notin \mathcal{S}_{1,\sigma}$ for all $\sigma < \sigma_0$. This also implies that no rigid transformation of $\mathbb{R}^2$ is part of $\mathcal{S}_{1,\sigma}$. If we now let $D = M = M_\sigma$ for some $0 < \sigma < \sigma_0$, we see that we do not recover $D$ up to a rigid transformation.

*Convex boundary and convex hole.* Let $K_a$ denote the axis-aligned ellipse of $\mathbb{R}^2$ with semi-major axis length equal to $a$ and perimeter equal to $2\pi$. Note that, necessarily, $1 \leq a < \pi/2$, with the extreme cases being the unit circle ($a = 1$) and the interval $[-\pi/2, \pi/2]$ swept twice ($a = \pi/2$). Denote by $b = b(a)$ the semi-minor axis length of $K_a$, implicitly defined by

$$\int_0^{2\pi} \sqrt{a^2 \sin^2 t + b^2 \cos^2 t}\, \mathrm{d}t = 2\pi.$$

We have

$$F(a) := \int_{K_a} \|x\|^2 \mathrm{d}x = \int_0^{2\pi} \left(a^2 \cos^2 t + b^2 \sin^2 t\right) \sqrt{a^2 \sin^2 t + b^2 \cos^2 t}\, \mathrm{d}t.$$

This daunting expression is much simplified when $a = 1$, in which case it is equal to $2\pi$, and when $a = \pi/2$, in which case it is equal to $\pi^2/12$. Since the former is larger than the latter, and $F$ is continuous in $a$, there is $a_\star$ such that, for $a > a_\star$, $F(a) < F(1)$. (We actually believe that $a_\star = 1$.)

Fix $a \in (a_\star, \pi/2)$ and let $M_0 = K_a = \phi^{-1}(K_1)$, where $\phi : \mathbb{R}^2 \to \mathbb{R}^2$ sends $x = (x_1, x_2)$ to $\phi(x) = (x_1/a, x_2/b)$. Note that $K_1$ is the unit circle. By the previous calculations and our choice for $a$, the identity function $\psi$ is not part of $\mathcal{S}_{1,0}$, since

$$\mathcal{E}_0(\psi) = \frac{1}{\pi} \int_{M_0} \|x\|^2 \mathrm{d}x = \frac{1}{\pi} F(a) < \frac{1}{\pi} F(1) = 2 = \frac{1}{\pi} \int_{M_0} \|\phi(x)\|^2 \mathrm{d}x = \mathcal{E}_0(\phi).$$

As before, let $M_\sigma$ be the (closed) $\sigma$-neighborhood of $M_0$. Again, there is a numeric constant $\sigma_0 > 0$ such that, when $\sigma < \sigma_0$, $\psi$ does not maximize the energy $\mathcal{E}_\sigma$, and we conclude again that if $D = M = M_\sigma$, MVU does not recover $D$ up to a rigid transformation.

## 5. Discussion

We leave behind a few interesting problems.

- *Convergence rate for the solution(s).* We obtained a convergence rate for the energy in Theorem 2, but no corresponding result for the solution(s). Such a result necessitates a fine examination of the speed at which the energy decreases near the space of maximizing functions.

- *Flattening property of MVU.* Assume that $M$ satisfies the Isometry Assumption. Though we showed that MVU is not always consistent in the sense that it may not recover the domain $D$ up to a rigid transformation, we believe that MVU always flattens the manifold $M$ in this case, meaning that it returns a set $S$ which is a subset of some $d$-dimensional affine subspace. If this were true, it would make MVU consistent in terms of dimensionality reduction!

- *Solution space in general.* As pointed out by Paprotny and Garcke (2012), and as we showed in Theorem 1, characterizing the solutions to Continuum MVU is crucial to understanding the behavior of Discrete MVU. In Theorem 3, we worked out the case where $M$ is isometric to a convex set. What can we say when $M$ is isometric to a sphere? Is MVU able to recover this isometry? This question is non-trivial even when $M$ is isometric to a circle. In fact, showing that the energy over ellipses (of same perimeter) is maximized for a circle is not straightforward, as seen in Section 4.3.

We speculate that a similar analysis would show that (Discrete) Isomap (Tenenbaum et al., 2000) converges to Continuum Isomap (Zha and Zhang, 2007). We are curious about the correspondence between Continuum Isomap and Continuum MVU.

## Acknowledgments

This work was partially supported by a grant from the National Science Foundation (NSF DMS 09-15160) and by a grant from the French National Research Agency (ANR 09-BLAN-0051-01). We are grateful to two anonymous referees for their helpful comments and for pointing out some typos.

## References

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(16):1373–1396, 2003.

M. Belkin and P. Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. In Peter Auer and Ron Meir, editors, *Learning Theory*, volume 3559 of *Lecture Notes in Computer Science*, pages 835–851. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-26556-6.

M. Bernstein, V. De Silva, J.C. Langford, and J.B. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Technical report, Department of Psychology, Stanford University, 2000.

M. Brand. Charting a manifold. *Advances In Neural Information Processing Systems*, pages 985–992, 2003.

A. Brudnyi and Y. Brudnyi. *Methods Of Geometric Analysis In Extension And Trace Problems. Volume 1*, volume 102 of *Monographs in Mathematics*. Birkhäuser/Springer Basel AG, Basel, 2012. ISBN 978-3-0348-0208-6.

D. Burago, Y. Burago, and S. Ivanov. *A Course In Metric Geometry*, volume 33 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2001. ISBN 0-8218-2129-6.

R.R. Coifman and S. Lafon. Diffusion maps. *Applied And Computational Harmonic Analysis*, 21 (1):5–30, 2006.

V. H. de la Peña and E. Giné. *Decoupling*. Probability and its Applications (New York). Springer-Verlag, New York, 1999. ISBN 0-387-98616-2. From dependence to independence, Randomly stopped processes. $U$-statistics and processes. Martingales and beyond.

D.L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *P. Natl. Acad. Sci. USA*, 100(10):5591–5596, 2003.

H. Federer. Curvature measures. *Trans. Amer. Math. Soc.*, 93:418–491, 1959. ISSN 0002-9947.

E. Giné and V. Koltchinskii. Empirical graph Laplacian approximation of Laplace-Beltrami operators: large sample results. In *High Dimensional Probability*, volume 51 of *IMS Lecture Notes Monogr. Ser.*, pages 238–259. Inst. Math. Statist., Beachwood, OH, 2006. doi: 10.1214/074921706000000888. URL http://dx.doi.org/10.1214/074921706000000888.

Y. Goldberg, A. Zakai, D. Kushnir, and Y. Ritov. Manifold learning: the price of normalization. *J. Mach. Learn. Res.*, 9:1909–1939, 2008. ISSN 1532-4435.

M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds – weak and strong pointwise consistency of graph laplacians. In Peter Auer and Ron Meir, editors, *Learning Theory*, volume 3559 of *Lecture Notes in Computer Science*, pages 470–485. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-26556-6.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.*, 58:13–30, 1963. ISSN 0162-1459.

A. N. Kolmogorov and V. M. Tikhomirov. $\varepsilon$-entropy and $\varepsilon$-capacity of sets in functional space. *Amer. Math. Soc. Transl. (2)*, 17:277–364, 1961. ISSN 0065-9290.

U. Lang and V. Schroeder. Kirszbraun's theorem and metric spaces of bounded curvature. *Geom. Funct. Anal.*, 7(3):535–560, 1997. ISSN 1016-443X. doi: 10.1007/s000390050018. URL http://dx.doi.org/10.1007/s000390050018.

P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete Comput. Geom.*, 39(1):419–441, 2008. ISSN 0179-5376. doi: http://dx.doi.org/10.1007/s00454-008-9053-2.

A. Paprotny and J. Garcke. On a connection between maximum variance unfolding, shortest path problems and isomap. In *Fifteenth International Conference on Artificial Intelligence and Statistics*, 2012.

D. Perrault-Joncas and M. Meila. Metric learning and manifolds: Preserving the intrinsic geometry. Technical report, Department of Statistics, University of Washington, 2012.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

L.K. Saul, K.Q. Weinberger, J.H. Ham, F. Sha, and D.D. Lee. Semisupervised learning. In B. Schoelkopf, O. Chapelle, and A. Zien, editors, *Spectral Methods For Dimensionality Reduction*, pages 293–308. MIT Press, 2006.

A. Singer. From graph to manifold laplacian: The convergence rate. *Applied and Computational Harmonic Analysis*, 21(1):128–134, 2006.

A.K. Smith, X. Huo, and H. Zha. Convergence and rate of convergence of a manifold-based dimension reduction algorithm. In *Proceedings of Neural Information Processing Systems*, pages 1529–1536. Citeseer, 2008.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

L.J.P. Van der Maaten, E.O. Postma, and H.J. Van den Herik. Dimensionality reduction: A comparative review. Technical report, Tilburg University, 2008.

U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, 36(2):555–586, 2008.

K. Q. Weinberger and Lawrence K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *National Conference on Artificial Intelligence (AAAI)*, volume 2, pages 1683–1686, 2006.

K.Q. Weinberger, F. Sha, and L.K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *International Confernence on Machine Learning (ICML)*, page 106, 2004.

K.Q. Weinberger, B.D. Packer, and L.K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *International Workshop on Artificial Intelligence and Statistics*, pages 381–388, 2005.

H. Weyl. On the volume of tubes. *Amer. J. Math.*, 61(2):461–472, 1939. ISSN 0002-9327.

Q. Ye and W. Zhi. Discrete hessian eigenmaps method for dimensionality reduction. Technical report, Department of Mathematics, University of Kentucky, 2012.

H. Zha and Z. Zhang. Continuum isomap for manifold learnings. *Computational Statistics & Data Analysis*, 52(1):184–200, 2007.

H.Z. Zha and Z. Zhang. Isometric embedding and continuum isomap. In *In Proceedings of the Twentieth International Conference on Machine Learning*. Citeseer, 2003.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via tangent space alignment. *SIAM J. Sci. Comput.*, 26(1):313–338, 2004.

Journal of Machine Learning Research 14 (2013) 1771-1800 Submitted 6/12; Revised 4/13; Published 7/13

# Variable Selection in High-Dimension with Random Designs and Orthogonal Matching Pursuit

**Antony Joseph** ANTONY.JOSEPH@STAT.BERKELEY.EDU
*Department of Statistics*
*University of California, Berkeley*
*Berkeley, CA-94720, USA*

**Editor:** Xiaotong Shen

## Abstract

The performance of orthogonal matching pursuit (OMP) for variable selection is analyzed for random designs. When contrasted with the deterministic case, since the performance is here measured after averaging over the distribution of the design matrix, one can have far less stringent sparsity constraints on the coefficient vector. We demonstrate that for exact sparse vectors, the performance of the OMP is similar to known results on the Lasso algorithm (Wainwright, 2009). Moreover, variable selection under a more relaxed sparsity assumption on the coefficient vector, whereby one has only control on the $\ell_1$ norm of the smaller coefficients, is also analyzed. As consequence of these results, we also show that the coefficient estimate satisfies strong oracle type inequalities.

**Keywords:** high dimensional regression, greedy algorithms, Lasso, compressed sensing

## 1. Introduction

Consider linear regression model,

$$Y = X\beta + \varepsilon \tag{1}$$

where $X \in \mathbb{R}^{n\times p}$, the coefficient vector $\beta \in \mathbb{R}^p$ and noise $\varepsilon \in \mathbb{R}^n$. The high dimensional case, where $p$ is of the same order, or possibly much larger than $n$, has been of immense interest nowadays. In many applications, interest is not primarily on prediction of the response $Y$, but on the accuracy of estimation of the coefficient $\beta$. Examples of such applications include, micro-array data analysis, graphical model selection (Meinshausen and Buhlmann, 2006), compressed sensing (Donoho, 2006a; Candès and Tao, 2006), and in communications (Barron and Joseph, 2012, 2010; Tropp, 2006). As is well known, in the high dimensional setting, $\beta$ is unidentifiable unless the design matrix $X$ is well-structured and there is some sparsity constraint on the coefficient vector $\beta$. This sparsity assumption corresponds to restricting $\beta$ to few non-zero entries ($\ell_0$-sparsity), or more generally, assuming that $\beta$ has only few terms that are large in magnitude.

The orthogonal matching pursuit (Pati et al., 1993) is a variant of the matching pursuit algorithm (Mallat and Zhang, 1993), where, successive fits are computed through the least squares projection of $Y$ on the current set of selected terms. For deterministic $X$ matrices, variable selection properties of this algorithm, for $\ell_0$-sparse vectors, have been analyzed for the noisy case in Zhang (2009a) and Cai and Wang (2011). However, as we shall review Section 1.2, although they give strong performance guarantees under certain conditions on the X matrix, they impose severe constraints on the sparsity of $\beta$. Similar results have been shown for the Lasso, for example in Zhao and Yu (2006).With random designs one can have reliable detection of the support with far less stringent

©2013 Antony Joseph.

sparsity constraints; the performance is here measured after averaging over the distribution of $X$. For example, Wainwright (2009) proved such results for the Lasso algorithm. The main results of this paper, apart from showing that similar properties hold for the OMP, demonstrate two important additional properties. Firstly, we give results on partial support recovery, which is important since exact recovery of support places strong requirements on $n$ if some of the non-zero elements are small in magnitude. Secondly, and more importantly, we relax the assumption that $\beta$ is $\ell_0$-sparse and address variable selection under a more general notion of sparsity, whereby one has only control on the $\ell_1$ norm of the smaller elements of $\beta$. We demonstrate that even under this more relaxed assumption, one can reliably estimate the position of the larger entries using the OMP. This has certain parallels with recent work on the Lasso by Zhang and Huang (2008). As a consequence of these results, we show that our coefficient estimate, after running the algorithm, satisfies strong oracle inequalities, similar to that demonstrated for the Lasso (Zhang, 2009b) and Dantzig selector (Candès and Tao, 2007).

The paper is organized as follows. Below, we describe the OMP algorithm. The stopping criterion we use is slightly different from what is traditionally used in literature. Section 1.2 motivates in greater detail our interest in random designs. In Section 2.1 we give results for design matrices that have i.i.d sub-Gaussian entries and $\ell_0$-sparse vectors. This extends the results in Tropp and Gilbert (2007) for the noisy case. In Section 2.2 we describe more general results with correlated Gaussian designs, where we only have control over the $\ell_1$ norm of the smaller coefficients. Sections 3, 4 and 5 gives proofs of our main results. The appendices contains auxiliary results.

### 1.1 The Orthogonal Matching Pursuit Algorithm

Denote as $J = J_1 = \{1, 2, \ldots, p\}$ to be the set of indices corresponding to columns in the $X$ matrix. For each step $i$, with $i \geq 1$, a single index $a(i)$ is detected to be non-zero in that step. Accordingly, denoting $d(i) = a(1) \cup a(2) \ldots \cup a(i)$ as the set of detected columns after $i$ steps, step $i+1$ of the algorithm only operates on the columns in $J_{i+1} = J - d(i)$, that is, the columns not detected in the previous steps. In other words, indices detected in previous steps remain detected.

The decision on whether a particular index $j$ is detected during a particular step $i$ is based on the absolute value of a statistic $\mathcal{Z}_{ij}$. Here, $\mathcal{Z}_{ij}$ is simply the inner product between $X_j$ and the normalized residual $R_{i-1}$ computed for the previous step.

Apart from the response vector $Y$ and design matrix $X$, the other input to the algorithm is a positive threshold value $\tau$. Our theoretical analysis assumes that each entry of $X$ has a sub-Gaussian distribution with mean 0 and scale 1. In practice, however, the algorithm should be performed after standardizing the columns of $X$ to have average 0 and norm $\sqrt{n}$. Denote $\|.\|$ as the euclidean norm. We now describe the OMP algorithm.

- Initialize $R_0 = Y$, $d(0) = \emptyset$. Start with step $i = 1$.
- Update
$$\mathcal{Z}_{ij} = X_j^{\mathrm{T}} \frac{R_{i-1}}{\|R_{i-1}\|}, \quad \text{for} \quad j \in J_i.$$
- If $\max_{j \in J_i} |\mathcal{Z}_{ij}| > \tau$, do the following:
  - Assign $a(i) = \arg\max\{|\mathcal{Z}_{i,j}| : j \in J_i\}$.

- Set $d(i) = d(i-1) \cup a(i)$. Update $R_i = (I - \mathcal{P}_i)Y$, where $\mathcal{P}_i$ is the projection matrix for the column space of $X_{d(i)}$, and set $J_{i+1} = J_i - a(i)$.
- Increase $i$ by one and go to step 2.

- Stop if $\max_{j \in J_i} |\mathcal{Z}_{ij}| \leq \tau$.

We remark that for any step $i$, the inner product $X_j^{\mathrm{T}} R_{i-1}$, for $j \in d(i-1)$, is 0. Correspondingly, since $\mathcal{Z}_{ij} = 0$, for $j \in d(i-1)$, the maximum of $\mathcal{Z}_{ij}$ over $j \in J_i$, is the same as the maximum over all $j \in J$. Also, the newly selected term $a(i)$ may be equivalently expressed as,

$$a(i) = \arg\min_{j \in J} \inf_{w \in \mathbb{R}} \|Y - Fit_{i-1} - wX_j\|^2,$$

where $Fit_{i-1}$ is the least squares fit of $Y$ on the columns in $d(i-1)$. In this respect, the OMP is similar to other greedy algorithms such as relaxed greedy and forward-stepwise algorithms (Barron et al., 2008; Huang et al., 2008; Jones, 1992; Lee et al., 1996), that operate through successive reduction in the approximation error.

As mentioned earlier, the stopping criterion considered here is slightly different from that considered in literature. Traditionally, for the no noise setting, the algorithm is run until there is a perfect fit between $Y$ and the selected terms, that is $R_i = 0$ (Tropp, 2004; Tropp and Gilbert, 2007). In the noisy case, as analyzed over here, there are two standard approaches. The first method (Cai and Wang, 2011; Zhang, 2009a) is to stop when $\max_{j \in J} |X_j^{\mathrm{T}} R_{i-1}|$ is less than some fixed threshold. The second approach (Donoho et al., 2006; Cai and Wang, 2011), is to stop when $\|R_i\|$ is less than some pre-specified value.

Our stopping criterion, which is more similar to the first approach, is equivalent to continuing the algorithm until $\max_{j \in J} |X_j^{\mathrm{T}} R_{i-1}| \leq \tau \|R_{i-1}\|$. The motivation for the use of such a statistic comes from the analysis of a similar iterative algorithm in Barron and Joseph (2010) for a communications setting. However, there the values of the non-zero $\beta_j$'s were known in advance; this added information played an important role in the analysis of the algorithm. A similar statistic was used by Fletcher and Rangan (2011) for an asymptotic analysis of the OMP for exact support recovery using i.i.d designs.

*Notation*: Let $a = a(n, p, k)$, $b = b(n, p, k)$ be two positive functions of $n$, $p$ and $k$. We denote as $a = O(b)$, if $a \leq c_1 b$ for some constant positive constant $c_1$ that is independent of $n$, $p$ or $k$. Similarly, $a = \Omega(b)$ means $a \geq c_2 b$ for positive $c_2$ independent of $n$, $p$ or $k$.

### 1.2 Related Work

As mentioned earlier, we are interested in variable selection in the high dimensional setting. Apart from iterative schemes, another popular approach is the convex relaxation scheme Lasso (Tibshirani, 1996). In order to motivate our interest in random design matrices, we describe existing results on variable selection, using both methods, with deterministic as well as random design matrices. For convenience, we concentrate on implications of these results assuming the simplest sparsity constraint on $\beta$, namely that $\beta$ has only a few non-zero entries.

In particular, we assume that,

$$|S_0(\beta)| = k, \quad \text{where } S_0(\beta) = \{j : \beta_j \neq 0\}. \tag{2}$$

In other words, attention is restricted to all $k$-sparse vectors, that is, those that have exactly $k$ non-zero entries. For convenience, we drop the dependence on $\beta$ and denote $S_0(\beta)$ as $S_0$ whenever there is no ambiguity. The simplest goal then is to recover $S_0$ exactly, under the additional assumption that all $\beta_j$, for $j \in S_0$, have magnitude at least $\beta_{min}$, where $\beta_{min} > 0$. Denote as $\mathcal{C} \equiv \mathcal{C}(\beta_{min}, k)$, as the set of coefficient vectors satisfying this assumption.

Further, denote $\hat{S}$ as the estimate of $S_0$ obtained using either method, and $\mathcal{E} = \{\hat{S} \neq S_0\}$ the error event that one is not able to recover the support exactly. For deterministic $X$, interest is mainly on conditions on $X$ so that

$$P_{err,X} = \sup_{\beta \in \mathcal{C}} \mathbb{P}_\beta(\mathcal{E}|X) \tag{3}$$

can be made arbitrarily small when $n$, $p$, or $k$ become large. Here $\mathbb{P}_\beta(.|X)$ denotes the distribution of $Y$ for the given $X$ and $\beta$.

A common sufficient condition on $X$ for this type of recovery is the *mutual incoherence condition*, which requires that the the inner product between distinct columns be small. In particular, letting $\|X_j\|^2/n = 1$, for all $j \in J$, it is assumed that

$$\gamma(X) = \frac{1}{n} \max_{j \neq j'} \left| X_j^{\mathrm{T}} X_{j'} \right| \tag{4}$$

is $O(1/k)$. Another related criterion is the *irrepresentable criterion* (Tropp, 2004; Zhao and Yu, 2006), which assumes, for all subset $T$ of size $k$, that

$$\|(X_T^{\mathrm{T}} X_T)^{-1} X_T^{\mathrm{T}} X_j\|_1 < 1, \quad \text{for all} \quad j \in J - T. \tag{5}$$

Here $\|.\|_1$ denotes the $\ell_1$ norm.

Observe that if $P_{err,X}$ (3) is small, it gives strong guarantees on support recovery, since it ensures that any $\beta$, with $|S_0(\beta)| = k$, can be recovered with high probability. However, it imposes severe constraints on the $X$ matrix. As as example, when the entries of $X$ are i.i.d Gaussian, the coherence $\gamma(X)$ is around $\sqrt{2\log p/n}$. Correspondingly, for (4) to hold, $n$ needs to be $\Omega(k^2 \log p)$. In other words, the sparsity $k$ should be $O(\sqrt{n/\log p})$, which is rather strong since ideally one would like $k$ to be of the same order as $n$. Similar requirements are needed for the irrepresentable condition to hold. Recovery using the irrepresentable condition has been shown for Lasso (Zhao and Yu, 2006; Wainwright, 2009), and for the OMP (Zhang, 2009a; Cai and Wang, 2011).

A natural question is to ask about requirements on $X$ to ensure recovery in an average sense, as opposed to the strong sense described above. One way to proceed, as done over here, is to consider random $X$ matrices and ask about the requirements on $n$, $p$, $k$, as well as $\beta_{min}$, so that

$$P_{err} = \sup_{\beta \in \mathcal{C}} \mathbb{P}_\beta(\mathcal{E}) \tag{6}$$

is small. Here $\mathbb{P}_\beta(\mathcal{E}) = E_X \mathbb{P}_\beta(\mathcal{E}|X)$, where the expectation on the right is over the distribution of $X$. For the Lasso, Wainwright (2009) considers random $X$ matrices, with rows drawn i.i.d $N_p(0,\Sigma)$. It is shown that under certain conditions on $\Sigma$, which can be described as population counterparts of the conditions for deterministic $X$'s, one can recover $S_0$ with high probability with $n = \Omega(k \log p)$ observations, with the constant depending inversely on $\beta_{min}^2$. The form of $n$ is in a sense ideal since now $k = O(n/\log p)$ is nearly the same $n$, if we ignore the $\log p$ factor. As mentioned earlier, apart from establishing similar properties to hold for the OMP with $k$-sparse vectors, we also demonstrate

strong support recovery results under a more general notion of sparsity. These results are described in the next section.

We also note that instead of averaging over $X$, one could assume a distribution on $\beta$ and analyze the average probability of $\mathcal{E}$ over this distribution. This is done in Candès and Plan (2009) for the Lasso. Here, for fixed magnitudes of the $k$ non-zero $\beta$, the support of $\beta$ is uniformly assigned over all possible subsets of size $k$. Once the support is chosen, the signs for the non-zero $\beta_j$'s are assigned $\pm 1$ with equal probability. If Avg[.] denotes the expectation with this distribution of $\beta$, it is shown that one could keep Avg $\left[\mathbb{P}_\beta(\mathcal{E}|X)\right]$ low for $\gamma(X)$ as high as $O(1/\log p)$. This condition on $\gamma(X)$ is less stringent than before and leads to a demonstration that $n = \Omega(k \log p)$ is sufficient for support recovery, provided $|||X||| \approx \sqrt{p/n}$, where $|||.|||$ denotes the spectral norm. We provide comparisons with this work in Section 6.

*Notation:* For a set $\mathcal{A} \subseteq J$, we denote as $X_\mathcal{A}$ the sub-matrix of $X$ comprising of columns with indices in $\mathcal{A}$. Similarly, for any $p \times 1$ vector $\beta$, we denote as $\beta_\mathcal{A}$ the $|\mathcal{A}| \times 1$ sub-vector with indices in $\mathcal{A}$. Also let $\mathcal{A}^c = J - \mathcal{A}$.

## 2. Results

Before discussing our main results with Gaussian matrices, in Section 2.1 we state results when the entries of $X$ are i.i.d sub-Gaussian and when the vector $\beta$ has $k$ non-zero entries. The noise vector is also assumed to come from a sub-Gaussian distribution with scale $\sigma$. This generalizes the results of Tropp and Gilbert (2007) for the noisy case. While preparing this manuscript we discovered that Fletcher and Rangan (2011) have analyzed the OMP for i.i.d designs and for $k$-sparse vectors, similar to that in Section 2.1. However, there the analysis was for exact support recovery and was asymptotic in nature. Further, they focused on a specific regime, where $k\beta_{min}^2/\sigma^2$ tends to infinity. We provide more comparisons with this work later on in the paper.

One consequence of our results is that $n = \Omega(k \log p)$ samples are sufficient for the recovery of any coefficient vector with $\beta_{min}$ that is at least the same order as the *noise level*. More specifically, define

$$\mu_n = \sqrt{(2 \log p)/n}. \tag{7}$$

The quantity $\sigma\mu_n$ can thought of as the noise level. To see why this is so, consider the orthogonal design where $X^T X/n = I$ and noise $\varepsilon \sim N(0, \sigma^2 I)$. Assume that, as usual, we are interested in recovering any $\beta$ with $|S_0(\beta)| = k$. A natural estimate of the support would be,

$$\hat{S} = \{j : |z_j| > t\} \quad \text{with} \quad z_j = X_j^T Y/n, \tag{8}$$

where $t$ is positive. Notice that $z_j \sim N(\beta_j, \sigma^2/n)$ for each $j \in J$. Correspondingly, since $z_j \sim N(0, \sigma^2/n)$, for $j \in J - S_0$, one sees that $t$ has to be of the form $\sigma\mu_n$ in order to prevent false discoveries with high probability. Similarly $\beta_j$, for all $j \in S_0$, has to have magnitude at least $\sigma\mu_n$ if one wanted to avoid false negatives.

The analysis of iid designs, as done in Section 2.1, forms an important ingredient to compressed sensing (Candès and Tao, 2006; Donoho, 2006a). However, it may not be useful for statistical applications, where typically the choice of the $X$ matrix is not under ones control. Accordingly, in Section 2.2, we assume that the rows of $X$ are drawn i.i.d from $N_p(0, \Sigma)$, with certain assumptions on $\Sigma$. This model was also employed to detect the neighborhood of a node in high dimensional graphs by Meinshausen and Buhlmann (2006). Moreover, we relax the assumption that $\beta$ is $k$-sparse and

only assume that there is a set $S = S(\beta)$, of size $k$, such that $\beta_{S^c}$ is sparse in a more general sense. Here $\beta_{S^c}$ denotes the vector of coefficients outside of $S$. More specifically, for a constant $\nu \geq 0$, if

$$S = \left\{ j : |\beta_j| > \sigma\nu\mu_n \right\}, \quad \text{with} \quad |S| = k, \tag{9}$$

we assume

$$\|\beta_{S^c}\|_1 \leq \sigma\eta\mu_n, \tag{10}$$

for an appropriately chosen $\eta$. A natural choice would be to take $\nu = 1$. Then, $S$ would correspond to the indices above the noise level. We show that for $\eta$ not too large, the OMP can detect the large indices in $S$ with high probability, provided $\Sigma$ satisfies certain conditions. As a consequence of these results, we show that the coefficient estimate satisfies strong oracle inequalities.

### 2.1 Recovery With Sub-Gaussian Designs

In this section we address the requirements on $n$, $p$, $k$ as well as $\beta_{min}$, to recover the support of $\beta$, either exactly or nearly so, where we assume that $|S_0(\beta)| = k$. Here $S_0(\beta)$ is as in (2). We allow the case that $k$ may be zero. Further, since it may not be a realistic assumption that $k$ is known, we assume that we only know an upper bound $\bar{k}$ on $k$, with $\bar{k} \geq \max\{k, 1\}$.

Let $X_{\ell j}$, for $\ell = 1, \ldots, n$ and $j = 1, \ldots, p$, denote the entries of the $X$ matrix. Throughout this section we assume that the $X_{\ell j}$'s are independent sub-Gaussian with mean 0 and scale 1, that is $\mathrm{E}e^{tX_{\ell j}} \leq e^{t^2/2}$, for $t \in \mathbb{R}$. Further, we assume that the noise vector $\varepsilon$ is independent of $X$ and has independent sub-Gaussian entries with mean 0 and scale $\sigma$, that is $\mathrm{E}e^{t\varepsilon_\ell} \leq e^{\sigma^2 t^2/2}$, for $t \in \mathbb{R}$, $\ell = 1, \ldots, n$. Additionally, if $k \geq 1$, we assume that the following two conditions are satisfied with high probability.

**Condition 1.** There exists $\lambda_{max} \geq \lambda_{min} > 0$, so that the eigenvalues of $X_{S_0}^{\mathrm{T}} X_{S_0}/n$ are between $\lambda_{min}$ and $\lambda_{max}$, that is

$$\lambda_{max}\|v\|^2 \geq \|X_{S_0} v\|^2/n \geq \lambda_{min}\|v\|^2 \quad \text{for all} \quad v \in \mathbb{R}^k.$$

**Condition 2.** The $\ell_2$ norm of the noise vector is bounded, that is $\|\varepsilon\|^2/n \leq \sigma^2\lambda$, for some $\lambda > 0$.

Let $\mathcal{E}_{cond}$ be the event that Conditions 1 or 2 fail. The first assumption is related to the restricted isometry property (Candès and Tao, 2005) and the sparse eigenvalues conditions (Zhang and Huang, 2008). Condition 1 is satisfied for a wide variety of random ensembles. For example (Baraniuk et al., 2007), it is satisfied with high probability for the Gaussian ensemble, where the $X_{\ell j}$ are i.i.d $N(0,1)$ and the binary ensemble, where the $X_{\ell j}$ are i.i.d uniform on $\{-1, +1\}$. Notice that since we are interested in controlling the probability $P_{err}$ in (6), because of the averaging over $X$, we do require that the Condition 1 hold uniformly over all $S_0$, with $|S_0| = k$. Condition 2, which bounds the $\ell_2$ norm of the noise vector, is required for controlling the norm of the residuals $R_i$. It is satisfied with high probability, for example, when the noise $\varepsilon \sim N(0, \sigma^2)$.

Below, we state the theorem giving sufficient conditions on $n$ for reliable recovery of the support of $\beta$. The threshold $\tau$ is taken to be

$$\tau = \sqrt{2(1+a)\log p}, \tag{11}$$

for some $a > 0$. Here $n$ will be a function $\bar{k}$ and $p$, as well as the various quantities defined above. The results of course hold with $\bar{k}$ replaced by $k$, provided $k$ is non-zero. In particular, for $\alpha, \delta > 0$, define

$$\xi \equiv \xi(\alpha, \delta) = \max\left\{(1+\delta)r_1, \sigma^2 r_2^2 f(\delta)/(\bar{k}\alpha)\right\}. \tag{12}$$

where,

$$r_1 = \frac{\max\{\lambda_{max}, \lambda\}}{\lambda_{min}^3} \qquad r_2 = \left[\frac{1}{\sqrt{\lambda_{min}}} + \sqrt{r_1}\right] \tag{13}$$

and

$$f(\delta) = \frac{1}{\left(1 - 1/\sqrt{1+\delta}\right)^2}.$$

Denote as $\hat{S} = \hat{S}(Y, X, \tau)$, the estimate of the support obtained after running the algorithm with the given $Y, X$ and threshold $\tau$. Further, denote the undetected elements of the support as $\hat{F} = S_0 - \hat{S}$. The theorem below, provides bounds on $\sum_{j \in \hat{F}} \beta_j^2$, the signal strength of the undetected components; here we assume that $\sum_{j \in \hat{F}} \beta_j^2 = 0$ if $\hat{F} = \emptyset$.

The following function of $k$ characterizes the probability of failure of the algorithm.

$$p_{err,k} = \mathbb{P}(\mathcal{E}_{cond}) + 2(k+1)/p^a + 2k/p^{1+a}, \quad \text{for } k \geq 1, \tag{14}$$

and $p_{err,0} = 2/p^a$. Here, recall that $\mathcal{E}_{cond}$ is event that Conditions 1 or 2 fail. Notice that $p_{err,k} \leq p_{err,\bar{k}}$, since $k \leq \bar{k}$.

Regarding the choice of $a$, if $k$ is $O(\log p)$, then $a$ can be taken to be slightly larger than 0 for $p_{err,k}$ to be small, assuming $p$ is large; however, if $k$ scales, for example, linearly with $p$, then $a$ needs to be taken to be larger than 1. We now state our theorem.

**Theorem 1** *Let the threshold $\tau$ be as in (11). Further, let n be of the form*

$$n = \xi \bar{k} \tau^2, \tag{15}$$

*with $\xi$ as in (12).*

*Then, if $k \geq 1$, the following condition holds, except on a set with probability $p_{err,k}$:*

$$\hat{S} \subseteq S_0 \quad \textit{and} \quad \sum_{j \in \hat{F}} \beta_j^2 \leq \alpha |\hat{F}|. \tag{16}$$

*In particular, if $\beta_{min}^2 > \alpha$ then $\hat{S} = S_0$, that is the support is recovered exactly, with probability at least $1 - p_{err,k}$.*

*If $k = 0$, $\hat{S} = \emptyset$ with probability at least $1 - p_{err,0}$.*

We remark that the proof of the theorem shows that the algorithm stops in at most $k$ steps, with probability at least $1 - p_{err,k}$. Also, notice that $\alpha$ controls accuracy to which the support is estimated. Assuming $\hat{F}$ is non-empty, another way of stating the theorem is that the average signal strength of the undetected components, that is $\|\beta_{\hat{F}}\|^2/|\hat{F}|$, is at most $\alpha$. It may seem desirable to make $\alpha$ as small as possible, however, doing so increases the value of $n$ in (15), since $n$ is inversely related to $\alpha$

through $\xi(\alpha, \delta)$. Further, if $\alpha$ is taken to be less than $\beta_{min}^2$, then the above theorem guarantees exact recovery of the support. Correspondingly, from (15) and (12), one sees that if

$$n = \max\left\{b_1\bar{k}, \frac{b_2}{\beta_{min}^2}\right\}\log p,$$

for some $b_1, b_2 > 0$, then the support can recovered exactly with high probability.

The following corollary, which is a consequence of Theorem 4, shows that if $n = \Omega(\bar{k}\log p)$, one can reliably detect the indices with large coefficient values, while ensuring that there are no false discoveries. Further, if all the non zero components are above the noise level (up to a constant factor), one can estimate the support exactly with the same number of observations.

**Corollary 2** *Define* $\bar{\xi} = 32r_2^2(1+a)$ *and* $r = 2r_2\sqrt{1+a}$. *Let*

$$n \geq \bar{\xi}\bar{k}\log p.$$

*Then, if* $k \geq 1$, *with probability at least* $1 - p_{err,k}$, *the estimate* $\hat{S}$ *is contained in* $S_0$ *and further,*

$$\left\{j : |\beta_j| > r\sigma\sqrt{k}\mu_n\right\} \subseteq \hat{S}.$$

*Further, if* $\beta_{min} > r\sigma\mu_n$, *then algorithm can recover the entire support of* $\beta$, *that is* $\hat{S} = S_0$, *with probability at least* $1 - p_{err,k}$.

*If* $k = 0$, *then* $\hat{S} = \emptyset$ *with probability at least* $1 - p_{err,0}$. *Here* $p_{err,\cdot}$ *is as in (14).*

### 2.2 More General Results With Gaussian Designs

For Gaussian ensembles, the methods used in the proof of Theorem 1 can be extended to give more general results on support recovery. In particular, we relax the assumption that $X$ has i.i.d entries and assume that rows of the $X$ matrix are i.i.d $N_p(0,\Sigma)$. We remark that knowledge of $\Sigma$ is not required for the implementation of the algorithm. The noise vector is assumed to be independent of $X$, with entries i.i.d. $N(0,\sigma^2)$. As mentioned earlier, here we also address a more general type of variable selection question, where we are not interested in recovering all non-zero entries but only the ones that are large compared to the noise level. In particular, for a constant $\nu \geq 0$, let $S$ be a set of size $k$ as in (9), consisting of the indices corresponding to the larger elements (in magnitude) of $\beta$. Once again, we do not assume that $k$ is known, but only assume that we have an upper bound $\bar{k}$ on $k$, with $\bar{k} \geq 1$. Unlike before, we do not require that the coefficients outside of $S$ are zero, but only assume that that $\|\beta_{S^c}\|_1 \leq \sigma\eta\mu_n$, where $\eta$ is allowed to scale at most linearly with $\bar{k}$, that is we assume that $\bar{\eta} = \eta/\bar{k}$ is $O(1)$.

Through a permutation of the columns one can, without loss of generality, write $\Sigma$ as

$$\Sigma = \begin{bmatrix} \Sigma_{SS} & \Sigma_{SS^c} \\ \Sigma_{S^cS} & \Sigma_{S^cS^c} \end{bmatrix},$$

where for $\mathcal{A}$, $\mathcal{A}' \subseteq J$, $\Sigma_{\mathcal{A},\mathcal{A}'} = \mathrm{Cov}(X_{1,\mathcal{A}}, X_{1,\mathcal{A}'})$ is the covariance matrix between terms in $\mathcal{A}$ and $\mathcal{A}'$. We denote the elements of the matrix as $\sigma_{ij}$, or $\Sigma_{ij}$, and use both notations interchangeably.

Without loss, we may assume that $\sigma_{jj} = 1$ for all $j$. To see this, notice that if the covariance matrix $\Sigma$ were known, we could divide each $X_j$ by $\sigma_{jj}$. If $\Sigma$ were unknown, we suggest dividing the

statistic $Z_{ij}$ by $\|X_j\|/\sqrt{n}$. This is equivalent to scaling the columns to have norm $\sqrt{n}$. In this case, the statistic becomes,

$$\tilde{Z}_{ij} := \frac{\sqrt{n}X_j^{\mathrm{T}}R_{i-1}}{\|X_j\|\|R_{i-1}\|} = \left(\frac{\sqrt{n}\sigma_j}{\|X_j\|}\right)\frac{X_j^{\mathrm{T}}R_{i-1}}{\sigma_j\|R_{i-1}\|}.$$

From Lemma 15, for any $a > 1$, with probability $1 - 1/p^{a-1}$,

$$\|X_j\|/\sqrt{n} \leq 1 + \sqrt{2a(\log p)/n}, \quad \text{for all } j = 1, \ldots, p.$$

Correspondingly, if $n$ were large compared to $\log p$, the statistic $\tilde{Z}_{ij}$ is close to $Z_{ij}$ and the analysis can be shown to carry through.

In a more general case, where the rows of $X$ are i.i.d. $N(\mu, \Sigma)$, with an unknown mean vector $\mu$, we recommend that one estimates the parameter $\mu$ by taking the average of rows. As mentioned in Section 1.1, the algorithm should be implemented after standardizing the $X$ matrix by subtracting out this estimated mean vector, followed by scaling the columns to ensure that they have norm $\sqrt{n}$. This case is more difficult to analyze theoretically since subtracting the estimated mean vector removes the independence of the rows, which is required for the analysis. However, we mention that one can obtain the same order of magnitude results on the sample size $n$ by doing the following: Divide the rows of $X$ into two sets, each of size $n/2$. Estimate $\mu$ by taking the average of the rows in the first set. Implement the algorithm on the second set after standardizing the columns in this set using the above estimated mean vector from the first set.

Notice that doing the above transforms the rows of the second set to have mean zero, while preserving its independence. The covariance matrix of the second set, after subtracting the mean vector, becomes $(1 + 2/n)\Sigma$, which is near $\Sigma$ for large $n$.

Correspondingly, from hereon, we assume that $\mu = 0$ and $\sigma_{jj} = 1$ for each $j$. We make the following assumptions on the correlation matrix $\Sigma$, when $k \geq 1$. These are essentially population analogs of the sparse eigenvalue and the irrepresentable conditions respectively.

1. There exists $s_{min}, s_{max} > 0$ so that,

$$\lambda_{min}(\Sigma_{TT}) \geq s_{min} \quad \text{and} \quad \lambda_{\max}(\Sigma_{TT}) \leq s_{min}, \tag{17}$$

   uniformly for all subsets $T$, with $|T| = k$. Here $\lambda_{min}(A), \lambda_{max}(A)$ denotes the minimum and maximum eigenvalues respectively of a square matrix $A$.

2. For some $\omega \in [0, 1)$, the following holds,

$$\max_{j \in J - T} \|\Sigma_{TT}^{-1}\Sigma_{Tj}\|_1 \leq \omega, \tag{18}$$

   uniformly for all subsets $T$ of size $k$. This is essentially the population analog of the irrepresentable condition (5).

Additionally, for $k \geq 1$, we make the following assumption that imposes bounds on certain interactions between $\beta_{S^c}$ and the correlation matrix $\Sigma$. As stated below, they are not very intuitive. Lemma 3, however, shows that under a simple condition, which controls the magnitude of correlations of the off diagonal elements of $\Sigma$, and along with (10), one can show (17) - (19) to hold.

Let $\Sigma_{S^c|S} = \Sigma_{S^cS^c} - \Sigma_{S^cS}\Sigma_{SS}^{-1}\Sigma_{SS^c}$, denote the variance of the conditional distribution of $X_{1,S^c}$ given $X_{1,S}$, where we recall that $S$ is the subset of indices comprising of the $k$ largest elements (in magnitude) of $\beta$. Let $\mu_n$ be as in (7). We make the following additional assumption.

3. For constants $\nu_1, \tilde{\nu_1} \geq 0$, the following holds,

$$\|\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}\|_\infty \leq \sigma\tilde{\nu_1}\mu_n \quad \text{and} \quad \|\Sigma_{S^c|S}\beta_{S^c}\|_\infty \leq \sigma\nu_1\mu_n. \tag{19}$$

Notice that condition (19) is not required when $\beta$ is exactly sparse, that is when it has $k$ non-zero entries, since in this case $\beta_{S^c}$ is identically equal to zero. In this case, assumptions (17, 18) for exactly sparse vectors are identical to the sufficient conditions for support recovery for the Lasso by Wainwright (2009).

As an example, for the standard Gaussian design, condition (17) is satisfied with $s_{min} = s_{max} = 1$. Condition (18) is satisfied with $\omega = 0$. Condition (19) reduces to requiring that $\max_{j\in S^c}|\beta_j| \leq \sigma\nu_1\mu_n$, which is satisfied with $\nu_1 = \nu$.

For the case $k = 0$, instead of (17) - (19), we only make the assumption,

$$\|\Sigma\beta\|_\infty \leq \sigma\nu_1\mu_n. \tag{20}$$

Notice that since in this case $S = \emptyset$ and $J = S^c$, alternatively, one may express the left side of the above as $\|\Sigma_{S^c|S}\beta_{S^c}\|_\infty$.

It is well known, see, for example, Cai and Wang (2011), Tropp (2004), that if the correlations between any two distinct columns are small, as given by the incoherence condition, it implies both the sparse eigenvalue condition (17) as well as the irrepresentable condition (18). We use these results to give simple sufficient conditions for (17) - (19), as well as (20) when $k = 0$, in the following lemma. For this, define the coherence parameter,

$$\gamma \equiv \gamma(\Sigma) = \max_{1\leq j\neq j'\leq p}|\Sigma_{jj'}|.$$

Further, recall that $\bar{\eta} = \eta/\bar{k}$. Then we have the following.

**Lemma 3** *Let $S$, with $|S| = k$, be as in (9). Assume that the correlation matrix $\Sigma$ satisfies,*

$$\gamma(\Sigma) \leq \omega_0/(2\bar{k}), \quad \textit{where} \quad 0 \leq \omega_0 < 1. \tag{21}$$

*Further, assume that the coefficient vector $\beta$ satisfies, for some $\eta \geq 0$,*

$$\|\beta_{S^c}\|_1 \leq \sigma\eta\mu_n. \tag{22}$$

*Define:*

$$s_{min} = 1 - \omega_0/2, \qquad s_{max} = 1 + \omega_0/2, \qquad \omega = \omega_0, \tag{23}$$

$$\tilde{\nu_1} = \omega_0\bar{\eta}, \qquad \nu_1 = \nu + \omega_0\bar{\eta}, \tag{24}$$

*Then, conditions (17) - (19) holds, for $k = 1, \ldots, \bar{k}$, with the above values of $s_{min}, s_{max}, \omega, \nu_1$ and $\tilde{\nu_1}$.*

*If $k = 0$, condition (20) holds with $\nu_1$ in (24).*

The above lemma is proved in Appendix C. Equation (21) controls the maximum correlation between distinct columns and can be regarded as the population analog of the incoherence condition (4). Condition (22) imposes that $\beta_{S^c}$ has $\ell_1$ norm that is $O(\eta\mu_n)$, where as mentioned before, $\eta$ is allowed to scale at most linearly with $\bar{k}$.

Henceforth, for convenience sake, assume that we have control over the incoherence parameter as in (21) and that $\beta$ satisfies (22). Further, the quantities $s_{min}$, $s_{max}$, $\omega$, $\nu_1$ and $\tilde{\nu_1}$ will be as in (23) and (24).

Condition (22) is more appropriate than an $\ell_1$ constraint on the whole vector $\beta$ since it does not impose any constraint on the larger coefficient values. Since the $\beta_j$, for $j \in S^c$, has magnitude at most $\sigma\nu\mu_n$, which is of the same order as the noise level, it makes sense for any algorithm to only estimate $S$ accurately. In Theorem 4 below, we give sufficient conditions on $n$ so that one can reliably estimate $S$. We note that this goal is different from that required in Zhang and Huang (2008) for support recovery with approximately sparse $\beta$. There, the only constraint on $\beta$ was that $\|\beta_{A_0}\|_1 = O(\eta\mu_n)$, for some set $A_0$, with $|A_0^c| = k$, and where $\eta$ is also allowed to grow at most linearly $k$. Since there was no constraint on the magnitude of $\beta_j$, for $j \in A_0$, some these $\beta_j$'s may have magnitude as high as $O(k\mu_n)$. For this reason, it made no longer sense to estimate $A_0^c$ accurately. Their criterion for an estimate $\hat{S}$ to be good was that $|\hat{S}| = O(k)$ and that the least squares fit of $Y$ on the columns in $\hat{S}$ produced a good approximation to $X\beta$.

The quantities $\lambda_{min}$, $\lambda_{max}$ and $\lambda$ are redefined here. These will now be expressed as functions $\nu$, $\omega_0$ and $\eta$ using the various quantities $s_{min}$, $s_{max}$, $\omega$, $\tilde{\nu_1}$ and $\nu_1$ defined in (23) and (24).

We will need that the quantity $h = \sqrt{k/n} + \mu_n$ to be strictly less than one. Below, we arrange $n > 2\bar{k}\log p$. Correspondingly, one sees that $h < 1$ if, for example, $\bar{k} \geq 5$ and $p \geq 8$. Let $h_\ell = (1-h)^2$ and $h_u = (1+h)^2$. We define the values of $\lambda_{min}$, $\lambda_{max}$ and $\lambda$ in the following manner:

$$\lambda_{min} = s_{min} h_\ell \quad \text{and} \quad \lambda_{max} = s_{max} h_u. \tag{25}$$

Further,

$$\lambda = (1 + s_{max}^2 \tilde{\nu_1}^2 + \nu_1 \bar{\eta}) \left(1 + \bar{k}^{-1/2}\right)^2. \tag{26}$$

Let $r_1$ be as in (13), now replaced with the above values of $\lambda_{min}, \lambda_{max}, \lambda$. The quantity $r_2$ is now given by,

$$r_2 = \left[ (1-\omega) \left( \tilde{\nu_1} + \sqrt{\frac{1+\nu_1\bar{\eta}}{\lambda_{min}}} \right) + \sqrt{r_1} \right]. \tag{27}$$

Notice that for the i.i.d Gaussian ensemble and when $\beta$ is $k$-sparse, the quantities $\omega, \tilde{\nu_1}, \nu_1$ and $\bar{\eta}$ can be taken as zero. Correspondingly, $r_2$ has the same form as that in (13).

Further, let $\xi = \xi(\alpha, \delta)$ be as in (12), with $r_1$ and $r_2$ appearing in its definition replaced with the values of these quantities defined above. The quantity $\tilde{p}_{err,k}$, for $k \geq 1$, which controls the probability of failure of the algorithm, is defined as,

$$\tilde{p}_{err,k} = 4/p + \frac{\sqrt{2/\pi}}{\tau} \left[ (k+1)/p^a + k/p^{1+a} \right]. \tag{28}$$

We define $\tilde{p}_{err,0} = 1/p + \sqrt{(2/\pi)}/(\tau p^a)$. The threshold will now be denoted as $\tau_1$. It will be greater than $\tau$ by a factor $\rho \geq 1$. This factor is strictly greater than one if $\beta$ is not $\ell_0$-sparse or if $\gamma(\Sigma)$ is non-zero. We are now in a position to state our main theorem.

**Theorem 4** *Let the assumptions of Lemma 3 hold. Set the threshold as $\tau_1 = \rho\tau$, where $\tau$ as in (11), and*

$$\rho = \frac{\nu_1 \left(1 + \bar{k}^{-1/2}\right) + 1}{1 - \omega}.$$

*Further, let*

$$n = \xi \bar{k} \tau_1^2. \tag{29}$$

*Then, if* $k \geq 1$ *the following holds with probability at least* $1 - \tilde{p}_{err,k}$*:*

$$\hat{S} \subseteq S \quad \textit{and} \quad \sum_{j \in \hat{F}} \beta_j^2 \leq \alpha |\hat{F}|, \tag{30}$$

*where* $\hat{F} = S - \hat{S}$*. In particular, if* $\beta_j^2 > \alpha$*, for all* $j \in S$*, then* $\hat{S} = S$ *with probability at least* $1 - \tilde{p}_{err,k}$*.*

*If* $k = 0$*, one has that* $\hat{S} = \emptyset$ *with probability at least* $1 - \tilde{p}_{err,0}$*.*

Before stating the analog of Corollary 2, as an aside, we give implications of the above theorem for exact recovery of support for $k$-sparse vectors and i.i.d designs for large $n$, $p$ and $k$. This will help in understanding the results of Theorem 4 better.

Wainwright (2009) show that for $k$-sparse vectors and i.i.d Gaussian designs, that there is a sharp threshold, namely $n \asymp 2k \log p$, for exact recovery of the support as $n$, $p$, $k$, as well as $k\beta_{min}^2/\sigma^2$, tends to infinity. This was also proved for the OMP by Fletcher and Rangan (2011), under an additional condition on rate of increase of the signal-to-noise ratio ($\|\beta\|^2/\sigma^2$). We can get similar results using our method by recalling that for i.i.d Gaussian designs and exact sparse vectors, $s_{min} = s_{max} = 1$ and $\omega, \nu_1, \tilde{\nu}_1$ and $\eta$ are all zero. Further, take $\bar{k} = k$. Correspondingly, since $h$ goes to 0, the quantities $\lambda_{min}, \lambda_{max}$ and $\lambda$ in (25, 26) tend to 1 as $n$, $p$ and $k$ become large. This implies that $r_1$ tends to one and $r_2$ (27) tends to 2. Further, as $k\beta_{min}^2/\sigma^2$ tends to infinity, one may also allow $k\alpha/\sigma^2$ tend to infinity, while keeping $\alpha < \beta_{min}$. From Theorem 4, this will ensure that the support will be recovered exactly. Next, let's evaluate the quantity $\xi$ (12) appearing in the expression for $n$. As $k\alpha/\sigma^2$ tends to infinity, one sees that the first term in the maximum in (12) is the active one and hence $\xi$ tends to $(1+\delta)$ (using $r_1$ tends to 1). One may also appropriately choose $\delta$ to tend to zero, making $\xi$ tend to 1. Accordingly, from (29), one sees that if $n \approx 2(1+a)k \log p$, for large $k$, $p$, one can recover the support exactly, with probability at least $1 - \tilde{p}_{err,k}$. When $\beta$ is extremely sparse, for example, when $k = O(\log p)$, then it is possible to arrange for $a$ to decrease to 0, while making $\tilde{p}_{err,k}$ also to 0. In this case, one gets the threshold $n \approx 2k \log p$ for exact recovery. However, in the regime where $k$ is not negligible compared to $p$ (for example, when $k/p$ is constant), then our results only allow for $a$ to tend to 1 (from above), so as to ensure $\tilde{p}_{err,k}$ goes to zero. In this case our results are slightly inferior, requiring $n \approx 4k \log p$ for exact recovery. We remark in Section 6 on how the results in Fletcher and Rangan (2011) may be carried over to the general case analyzed here.

We now state the analog of Corollary 2. The goal now is not to recover the non-zero entries, but only those that are large compared to the noise level, which is a subset of $S$. We have the following.

**Corollary 5** *Let the assumptions of Lemma 3 hold and set the threshold to be* $\tau_1$ *as in Theorem 4. Define* $\bar{\xi} = 32(r_2\rho)^2(1+a)$ *and* $r = 2r_2\rho\sqrt{1+a}$*, where* $r_2$ *as in (27). Let*

$$n \geq \bar{\xi} \bar{k} \log p.$$

*Then, if* $k \geq 1$*, with probability at least* $1 - \tilde{p}_{err,k}$*, the estimate* $\hat{S}$ *is contained in* $S$ *and,*

$$\left\{ j : |\beta_j| > r\sigma\sqrt{k}\mu_n \right\} \subseteq \hat{S}. \tag{31}$$

*Further, if* $|\beta_j| > r\sigma\mu_n$*, for all* $j \in S$*, one has* $\hat{S} = S$ *with probability at least* $1 - \tilde{p}_{err,k}$*.*

*If* $k = 0$*, then* $\hat{S}$ *is* $\emptyset$ *with probability at least* $1 - \tilde{p}_{err,0}$*.*

We remark that it was assumed that the conditions of Lemma 3 holds for convenience. More generally, the above holds under (17)-(19). In particular, for exact sparse vectors, the conditions are exactly similar to those in Wainwright (2009). Also, as with the case with sub-Gaussian designs, the proof also demonstrates that the algorithm stops within $k$ steps, with probability at least $1-\tilde{p}_{err,k}$.

Corollary 5 gives strong performance guarantees for the OMP under certain conditions on the correlation matrix and an $\ell_1$ constraint on the smaller coefficients. From (31), one sees that the larger coefficients, that is, those with magnitude $\Omega(\sqrt{k}\mu_n)$, are contained in $\hat{S}$ with high probability. Better performance can be demonstrated when all $\beta_j$'s, for $j \in S$, have magnitude $\Omega(\mu_n)$. In this case, it is possible to recover $S$, while ensuring that there are no false positives. This is in a sense ideal, since it is nearly what one would expect in the orthogonal design case discussed in the beginning of Section 2. In this case, assuming $\hat{S}$ is as in (8), one sees that in order to prevent false positives, $t$ needs to be $\Omega(\mu_n)$. Thus $|\beta_j|$, for $j \in S$, also needs to be $\Omega(\mu_n)$, with a slightly larger constant, to ensure $\hat{S} = S$. For example, if the $|\beta_j|$'s, for $j \in S$, is at least $\tilde{t} = (\nu + 2\sqrt{1+a})\sigma\mu_n$, then it is not hard to see that the probability $\hat{S} = S$ is at least $1 - 2/p^a$. Of course, the factor of $r\sigma$ obtained here, is larger than the corresponding factor for the orthogonal case, since the $X$ matrix is in general quite far from being orthogonal; indeed, it is singular when $p > n$.

As a consequence of the above, we state results demonstrating strong oracle inequalities for parameter estimation under the $\ell_2$-loss.

### 2.2.1 ORACLE INEQUALITIES UNDER $\ell_2$-LOSS

Let $\hat{\beta}$ be the coefficient estimate obtained after running the algorithm. More explicitly, $(\hat{\beta}_j : j \in \hat{S})$ is simply the least squares estimate when $Y$ is regressed on $X_{\hat{S}}$ and $\hat{\beta}_j = 0$ for $j \in \hat{S}^c$.

We assume that the correlation matrix $\Sigma$ satisfies (21), that is,

$$\gamma(\Sigma) \leq \omega_0/(2\bar{k}), \tag{32}$$

where $0 \leq \omega_0 < 1$.

For simplicity, we consider the case that $\beta$ satisfies (9) with $\nu = 1$, that is,

$$S = \{j : |\beta_j| > \sigma\mu_n\} \quad \text{and} \quad \|\beta_{S^c}\|_1 \leq \sigma\eta\mu_n, \tag{33}$$

where $|S| = k$ and $\eta$ is allowed to grow at most linearly with $\bar{k}$, that is $\bar{\eta} = \eta/\bar{k}$ is $O(1)$. With $\nu = 1$, $S$ denotes the set of indices greater than the noise level.

For the above values of $\eta$, $\omega_0$ and with $\nu = 1$, evaluate the quantities $s_{min}$, $s_{max}$ as well as $\tilde{\nu_1}$, $\nu_1$ and $\omega$ using expressions (23) and (24). Evaluate $r_2$ as in (27), where the quantities $\lambda$, $\lambda_{min}$, $\lambda_{max}$ are calculated using Equations (25, 26). Further, let $\bar{\xi}$ and $r$ be as in Corollary 5. Then we have the following.

**Theorem 6** *Let (32) and (33) hold. For fixed such* $\beta$*, if*

$$n \geq \bar{\xi}\bar{k}\log p,$$

*then the following holds with probability at least* $1-\tilde{p}_{err,k}$*:*

$$\|\hat{\beta} - \beta\|^2 \leq C\sum_{j=1}^{p} \min\left(\beta_j^2, \sigma^2\mu_n^2\right),$$

*where* $C = (4/9)r^2$.

The above theorem is essentially the analog of similar results for the Lasso (Zhang, 2009b, Corollary 6.1) and Dantzig selector (Candès and Tao, 2007, Theorem 1.2). Note, the latter assumes that $\beta$ is $k$-sparse. Our results are more general since we only assume that the $\ell_1$ norm of the smaller coefficients satisfies a certain bound. We proceed to state the corollary of the result assuming $\beta$ is $k$-sparse.

For $k$-sparse $\beta$, we only assume that (32) holds. Take $\eta = \bar{k}$, so that $\bar{\eta} = 1$. Evaluate $r_2$ using this values of $\eta$, and with $\nu = 1$, and call it $r_2^*$, that is,

$$r_2^* = \left[ (1-\omega_0) \left( \omega_0 + \sqrt{\frac{2+\omega_0}{\lambda_{min}}} \right) + \sqrt{r_1} \right],$$

where once again, the quantities $r_1$ and $\lambda_{min}$ as calculated using (13, 25) and Equations (23) and (24). Further, let $\xi^*$ have the same expression as $\bar{\xi}$, except it is evaluated using $r_2^*$ instead of $r_2$. Similarly, let $r^* = 2r_2^*\rho\sqrt{1+a}$. Then we have the following.

**Corollary 7** *Let (32) hold and let $\beta$ be a fixed $k$-sparse vector, for some $k \geq 0$. If*

$$n \geq \xi^* \bar{k} \log p,$$

*then for $C_1 = (4/9)(r^*)^2$, the following holds except on a set with probability $\tilde{p}_{err,k}$:*

$$\|\hat{\beta} - \beta\|^2 \leq C_1 \sum_{j=1}^{p} \min\left(\beta_j^2, \sigma^2 \mu_n^2\right).$$

We now proceed to give proofs of our main results. The proofs employs techniques developed in Zhang (2009a) and Tropp and Gilbert (2007).

## 3. Proof Of Results In Section 2.1

**Proof** [Proof of Theorem 1] The following statistics will be useful in our analysis. Denote,

$$Z_i = \max_{j \in S_0} |Z_{ij}| \quad \text{and} \quad \tilde{Z}_i = \max_{j \in S_0^c} |Z_{ij}|$$

Notice if $Z_i > \tau$ and $Z_i > \tilde{Z}_i$, then the index detected in step $i$, that is $a(i)$, belongs to $S$.

We first prove for the case $k \geq 1$. Let $\mathcal{E}$ be the event that statement (16) in Theorem 1 does not hold. We want to show that the probability of $\mathcal{E}$ is small. There are two types of errors that we wish to control. Let $\mathcal{E}_1$ be the event that $\hat{S}$ in not contained in $S_0$. Further, let $\mathcal{E}_2$ be the event that $\hat{S}$ is contained is $S_0$, however $\sum_{j \in \hat{F}} \beta_j^2 > \alpha |\hat{F}|$. Clearly, $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$.

We use an argument similar to that used by Tropp and Gilbert (2007). We initially pretend that $X = X_{S_0}$ and that the coefficient vector $\beta$ is shortened to a $k \times 1$ vector $\beta_{S_0}$ with all non-zero entries. Notice that $Y = X_{S_0}\beta_{S_0} + \varepsilon$. For a given threshold $\tau$, we run the algorithm on this truncated problem. Notice that $m$ has to be less than $k$ since the number of columns of $X_{S_0}$ is at most $k$. Let $\tilde{R}_1, \tilde{R}_2, \ldots, \tilde{R}_m$ be the associated residuals after each step. Also, denote as $\tilde{R}_0$ the vector $Y$. Notice that $m, \tilde{R}_0, \tilde{R}_1, \ldots, \tilde{R}_m$ are functions of $A = [X_{S_0} : \varepsilon]$.

Let $\mathcal{E}_u$ be the event that statement (16) does not hold for the truncated problem. More explicitly, taking $\hat{S}_1 = \hat{S}(Y, X_{S_0}, \tau)$ and $\hat{F}_1 = S_0 - \hat{S}_1$, it is the event that $\|\beta_{\hat{F}_1}\|^2 > \alpha |\hat{F}_1|$.

Denote $T_i = \max_{j\in S_0} \left|X_j^{\mathrm{T}}\tilde{R}_{i-1}/\|\tilde{R}_{i-1}\|\right|$ and $\tilde{T}_i = \max_{j\in S_0^c} \left|X_j^{\mathrm{T}}\tilde{R}_{i-1}/\|\tilde{R}_{i-1}\|\right|$, for $i = 1,\ldots,m+1$. Notice that the statistics $T_i, \tilde{T}_i$ are similar to $\mathcal{Z}_i, \tilde{\mathcal{Z}}_i$, the only difference being that the residuals involved in the former arise from running the algorithm on the truncated problem, whereas in the latter they arise from consideration of the original problem. Further, let $\mathcal{E}_f$ be the event

$$\mathcal{E}_f = \left\{\tilde{T}_i > \tau, \tilde{T}_i \geq T_i \text{ for some } i \leq m+1\right\}.$$

We now show that $\mathcal{E} \subseteq \mathcal{E}_u \cup \mathcal{E}_f$. To see this, write $\mathcal{E}$ as a disjoint union $\mathcal{E}_1 \cup \tilde{\mathcal{E}}_2$, where $\tilde{\mathcal{E}}_2 = \mathcal{E}_2 \cap \mathcal{E}_1^c$. Let's first consider the case that $\tilde{\mathcal{E}}_2$ occurs. Clearly this means that $\mathcal{E}_u$ has occurred if the algorithm were run on the truncated problem for the given $A$.

Next, consider the case that $\mathcal{E}_1$ occurs. Let $R_0, R_1 \ldots$etc. be the residuals for the original problem (1), for the given realization of $[X : \varepsilon]$. Let $i^*$ be the step for which the false alarm occurs for the first time. Clearly, $i^* \leq m+1$, since otherwise it would mean that the truncated problem (with $X = X_{S_0}$) ran for more than $m$ steps. Also, we must have $\{\mathcal{Z}_i > \tau, \mathcal{Z}_i > \tilde{\mathcal{Z}}_i\}$ occur for $1 \leq i \leq i^*-1$ and $\{\tilde{\mathcal{Z}}_{i^*} > \tau, \tilde{\mathcal{Z}}_{i^*} \geq \mathcal{Z}_{i^*}\}$ occur. Correspondingly, one sees that $R_0 = \tilde{R}_0, \ldots, R_{i^*-1} = \tilde{R}_{i^*-1}$, which implies that $T_{i^*} = \mathcal{Z}_{i^*}$ and $\tilde{T}_{i^*} = \tilde{\mathcal{Z}}_{i^*}$. Consequently, as $\{\tilde{T}_{i^*} > \tau, \tilde{T}_{i^*} \geq T_{i^*}\}$ occurs, $\mathcal{E}_f$ occurs. Hence, $\mathcal{E} \subseteq \mathcal{E}_u \cup \mathcal{E}_f$ which gives,

$$\mathbb{P}(\mathcal{E}) \leq \mathbb{P}(\mathcal{E}_u) + \mathbb{P}(\mathcal{E}_f).$$

Consequently, all we are left with is to bound the probabilities of $\mathcal{E}_f$ and $\mathcal{E}_u$.

We first bound the probability of $\mathcal{E}_f$. For this, notice that $\mathcal{E}_f \subseteq \mathcal{E}_f'$, where $\mathcal{E}_f' = \{\max_{1\leq i\leq m+1} \tilde{T}_i > \tau\}$. Since $X_{S_0^c}$ is independent of $A = [X_{S_0} : \varepsilon]$, one has that $X_{S_0^c}$ is independent of $\tilde{R}_1, \ldots, \tilde{R}_m$. Correspondingly, from Lemma 13 (a), conditional on $A$, we have that $X_j^{\mathrm{T}}\tilde{R}_i/\|\tilde{R}_i\|$ is sub-Gaussian with mean 0 and scale 1, for $j \in S_0^c$ and $1 \leq i \leq m+1$. Consequently, using standard results on the maximum of sub-Gaussian random variables (Lemma 13 (b)), if $\tau$ be as in (11), one gets that $\mathbb{P}(\mathcal{E}_f|A) \leq 2(m+1)/p^a$, using $|S_0^c| \leq p$. Since $m \leq k$, this probability is bounded by $2(k+1)/p^a$, which implies $\mathbb{P}(\mathcal{E}_f) \leq 2(k+1)/p^a$.

Next, we bound the probability of $\mathcal{E}_u$. For this, consider a linear model of the form,

$$U = H\varphi + w, \tag{34}$$

where $H$ is an $n \times k$ matrix satisfying, $w$ an $n \times 1$ vector and $\varphi$ a $k \times 1$ dimensional coefficient vector. After running the OMP on this model (with $Y = U, X = H$ and threshold $\tau_0$), let $\hat{S}_2 = \hat{S}(U, H, \tau_0)$ be the estimate of the support. Further, let $\hat{\varphi}$ be the coefficient estimate obtained, that is, $(\hat{\varphi}_j : j \in \hat{S}_2)$ is the least squares estimate when $U$ is regressed on $H_{\hat{S}_2}$ and $\hat{\varphi}_j = 0$ for $j$ not in $\hat{S}_2$. We use the following Lemma, the proof of which is similar to the analysis by Zhang (2009a).

**Lemma 8** *For the model (34), let the following hold.*

*(i) Condition 1 holds for $H$, that is the eigenvalues of $H^{\mathrm{T}}H/n$ are between $\lambda_{min}$ and $\lambda_{max}$.*

*(ii) Condition 2 holds for $w$, that is $\|w\|^2 \leq n\sigma^2\lambda$, for some $\lambda > 0$.*

*(iii) $\|\hat{\varphi}_{ls} - \varphi\|_\infty \leq \sigma c_0 \tau_0/\sqrt{n}$, for some constant $c_0 > 0$, where $\hat{\varphi}_{ls}$ is the coefficient vector of the least square fit of $U$ on $H$.*

*Under the above, if the OMP is run with $Y = U$, $X = H$ and threshold $\tau_0$, when the algorithm stops we must have the following,*

*(a)*

$$\left(1-\tau_0\sqrt{r_1 k/n}\right)\|\varphi_{\hat{F}_2}\| \leq \tilde{r}_2\sigma\tau_0\sqrt{\frac{|\hat{F}_2|}{n}}, \tag{35}$$

*where $\hat{F}_2 = \{1,\ldots,k\} - \hat{S}_2$, denotes the indices not detected after running the algorithm. Further, $r_1$ has the same form as (13), replaced with the above values of $\lambda_{min}$, $\lambda_{max}$ and $\lambda$. Also, $\tilde{r}_2 = c_0 + \sqrt{r_1}$.*

*(b)*

$$\|\hat{\varphi} - \varphi\| \leq \frac{\tilde{r}_2\sigma\tau_0\sqrt{k/n}}{1-\tau_0\sqrt{r_1 k/n}}.$$

The above lemma is proved in Appendix B. We only require the conclusions in part (a) of the lemma for the time being. Part (b) will be required of Section 2.2.1 to get bounds on $\ell_2$-error of the coefficient estimate.

Now apply Lemma 8 to the truncated problem, that is, with $H = X_{S_0}$, $\varphi = \beta_{S_0}$, $U = Y$ and $\tau_0 = \tau$. Notice that in this case $\hat{F}_2 = \hat{F}_1$ and $\hat{S}_2 = \hat{S}_1$. We know that requirements (i) and (ii) of the Lemma 8 hold, except on a set $\mathcal{E}_{cond}$. The following lemma shows that (iii) holds with high probability.

**Lemma 9** *Let $\hat{\beta}_{ls}$ be the least squares fit when $Y$ is regressed on $X_{S_0}$. Further, let*

$$\mathcal{E}_{ls} = \{\|\hat{\beta}_{ls} - \beta_{S_0}\|_\infty > \sigma c_0 \tau/\sqrt{n}\},$$

*where $c_0 = 1/\sqrt{\lambda_{min}}$. Then $\mathbb{P}\left(\mathcal{E}_{ls} \cap \mathcal{E}_{cond}^c\right) \leq 2k/p^{1+a}$.*

The above lemma is proved after this proof. Using the above lemma, all requirements of Lemma 8 hold, except on a set $\tilde{\mathcal{E}}_u = \mathcal{E}_{cond} \cup \mathcal{E}_{ls}$, the probability of which is bounded by $\mathbb{P}(\mathcal{E}_{cond}) + 2k/p^{1+a}$. We now show that $\mathcal{E}_u \subseteq \tilde{\mathcal{E}}_u$. We do this by showing $\tilde{\mathcal{E}}_u^c \subseteq \mathcal{E}_u^c$. To see this, notice that on $\tilde{\mathcal{E}}_u^c$, one has

$$\left(1-\tau\sqrt{r_1 k/n}\right)\|\beta_{\hat{F}_1}\| \leq \tilde{r}_2\sigma\tau\sqrt{\frac{|\hat{F}_1|}{n}}. \tag{36}$$

from (35). Assume that $\hat{F}_1$ is non-empty, since otherwise the claim is trivially true. Notice that since $n \geq (1+\delta) r_1 \bar{k}\tau^2$ from (15), one has $\tau\left(\bar{k} r_1/n\right)^{1/2} \leq 1/\sqrt{1+\delta}$. Now, since $k \leq \bar{k}$, the left side of (36) is non-negative. Thus, (36) can be re-expressed as,

$$\|\beta_{\hat{F}_1}\|^2 \leq (\sigma^2 r_2^2 f(\delta)\tau^2/n)|\hat{F}_1|,$$

which follows from noticing that $r_2 = \tilde{r}_2$, where $r_2$ is as in (13). Now, since $n \geq \sigma^2 r_2^2 f(\delta)\tau^2/\alpha$, the left side of the above is at most $\alpha|\hat{F}_1|$. Thus, $\sum_{j\in\hat{F}_1}\beta_j^2 \leq \alpha|\hat{F}_1|$ on $\tilde{\mathcal{E}}_u^c$, which implies that $\mathcal{E}_u \subseteq \tilde{\mathcal{E}}_u$. Consequently, $\mathbb{P}(\mathcal{E}_u) \leq \mathbb{P}(\mathcal{E}_{cond}) + 2k/p^{1+a}$. Accordingly, since $\mathbb{P}(\mathcal{E}) \leq \mathbb{P}(\mathcal{E}_u) + \mathbb{P}(\mathcal{E}_f)$, one has $\mathbb{P}(\mathcal{E}) \leq \mathbb{P}(\mathcal{E}_{cond}) + 2k/p^{1+a} + 2(k+1)/p^a$, which is equal to $p_{err,k}$. This completes the proof for the case $k \geq 1$.

For the case $k = 0$, we just need to show that the algorithm stops after the first step, in which case $\hat{S} = \emptyset$. This is immediately seen by noticing that for $k = 0$, one has that $\mathcal{Z}_{1j}$, for $j \in J$, are sub-Gaussian with mean 0 and scale 1. Correspondingly, from Lemma 13(b), the event $\{\max_{j\in J}|\mathcal{Z}_{1j}| > \tau\}$ has probability at most $p_{err,0} = 2/p^a$. ■

**Proof** [Proof of Lemma 9] Note that $\hat{\beta}_{ls} - \beta_{S_0}$ can be expressed as $Z = (X_{S_0}^{\mathrm{T}} X_{S_0})^{-1} X_{S_0}^{\mathrm{T}} \varepsilon$. Let $Z = (Z_j : j = 1, \ldots, k)$. Now, conditioned on $X_{S_0}$, each $Z_j$ is sub-Gaussian with mean 0 and scale $\sigma_j = \sigma \sqrt{e_j^{\mathrm{T}} (X_{S_0}^{\mathrm{T}} X_{S_0})^{-1} e_j}$. Here, $e_j$ is the $j$ th column of the size $k$ identity matrix. Correspondingly, from Lemma 13(b), one gets $\max_j |Z_j|$ is less than $(\max_j \sigma_j)\tau$, except on a set with probability $2k/p^{1+a}$. Finally, observe that on $\mathcal{E}_{cond}^c$, one has $e_j^{\mathrm{T}} (X_{S_0}^{\mathrm{T}} X_{S_0})^{-1} e_j \leq 1/(n\lambda_{min})$, since the maximum eigenvalue of $(X_{S_0}^{\mathrm{T}} X_{S_0}/n)^{-1}$ is at most $1/\lambda_{min}$. Thus, $\max_j \sigma_j \tau$ is at most $\sigma c_o \tau/\sqrt{n}$, with $c_0 = 1/\sqrt{\lambda_{min}}$. ■

**Proof** [Proof of Corollary 2] Take $\alpha(\delta) = \sigma^2/[(1+\delta)\bar{k}]$. Further, let $\xi(\delta) = \xi(\alpha(\delta), \delta)$, which, using $r_2^2 \geq r_1$ and $f(\delta) \geq 1$, can be written as,

$$\xi(\delta) = (1+\delta) f(\delta) r_2^2.$$

The function $(1+\delta)f(\delta)$, for $\delta > 0$, has its minimum at $\delta^* = 3$. Further, it is increasing and goes to infinity as $\delta$ tends to infinity. Now, using $\xi(\delta^*) = 16 r_2^2$, notice that $\xi(\delta^*)\bar{k}\tau^2 = \bar{\xi}\bar{k}\log p$. Correspondingly, since $n \geq \bar{\xi}\bar{k}\log p$, one gets that

$$n = \xi(\delta)\bar{k}\tau^2, \tag{37}$$

for some $\delta \geq \delta^*$. Consequently, from Theorem 1, one has,

$$\hat{S} \subseteq S_0 \quad \text{and} \quad \sum_{j \in \hat{F}} \beta_j^2 \leq \alpha(\delta)|\hat{F}|, \tag{38}$$

with probability at least $1 - p_{err,k}$. Use $f(\delta) \leq f(\delta^*) = 4$, to get from (37) that $n \leq (1+\delta) r \bar{k} \tau^2$. Correspondingly, $\alpha(\delta)$ is at most $r^2\sigma^2\mu_n^2$. Consequently, any $j$, with $|\beta_j| > r\sigma\sqrt{k}\mu_n$ cannot be in $\hat{F}$ since it would contradict the inequality in (38). Further, if $\beta_{min} > r\sigma\mu_n$, the inequality in (38) cannot hold if $\hat{F}$ is non-empty. In this case the algorithm recovers the entire support. ■

## 4. Proof Of Results In Section 2.2

**Proof** [Proof of Theorem 4] Once again, we first prove for the case $k \geq 1$. As before, we are interested in bounding the probability of $\mathcal{E}$, where $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$. Here $\mathcal{E}_1$ is the event that $\hat{S}$ is not contained in $S = S(\beta)$. Also, $\mathcal{E}_2$ is the event $\hat{S} \subseteq S$ and $\|\beta_{\hat{F}}\|^2 > \alpha|\hat{F}|$, where, here $\hat{F} = S - \hat{S}$ and $\hat{S} = \hat{S}(Y, X, \tau_1)$. Write $Y$ as $Y = X_S\beta_S + \tilde{\varepsilon}$, where $\tilde{\varepsilon} = X_{S^c}\beta_{S^c} + \varepsilon$. Analogous to before, we initially pretend that $X = X_S$ and $\beta = \beta_S$ and run the algorithm on the truncated problem to get residuals $\tilde{R}_0, \tilde{R}_1, \tilde{R}_2, \ldots, \tilde{R}_m$. These residuals are functions of $A = [X_S : \tilde{\varepsilon}]$. Further, as before, let $\mathcal{E}_u$ be the event that statement (30) is not met for this truncated problem. With $\hat{S}_1 = \hat{S}(Y, X_S, \tau_1)$ and $\hat{F}_1 = S - \hat{S}_1$, it is the event that $\|\beta_{\hat{F}_1}\|^2 > \alpha|\hat{F}_1|$. Similarly, we define $T_i$, $\tilde{T}_i$ as before, now with the maximum taken over $S$ instead of $S_0$. Further, define the event $\mathcal{E}_f$ analogous to before, with $\tau$ replaced by $\tau_1$. Using the same reasoning as in Theorem 1, one has $\mathcal{E} \subseteq \mathcal{E}_u \cup \mathcal{E}_f$. We first proceed to bound the probability of $\mathcal{E}_f$. Notice that unlike previously, the $X_j$'s, for $j \in S^c$, are not independent of the $\tilde{R}_i$'s. This makes bounding the probability of $\mathcal{E}_f$ more involved.

The following lemma will be useful, both in bounding $\mathbb{P}(\mathcal{E}_f)$ as well as $\mathbb{P}(\mathcal{E}_u)$. We denote as $\hat{\beta}_{ls}$ the least square estimate when $Y$ is regressed on $X_S$.

**Lemma 10** *Parts (i)-(iii) of this lemma demonstrate that requirements (i)-(iii) of Lemma 8 are satisfied with high probability.*

*(i) With $\lambda_{min}, \lambda_{max}$ as in (25), the following holds with probability at least 1 - 2/p:*

$$\lambda_{min}\|v\|^2 \leq \|X_S v\|^2/n \leq \lambda_{max}\|v\|^2 \quad \textit{for all} \quad v \in \mathbb{R}^k.$$

*(ii) Let $\lambda$ be as in (26). Then $\|\tilde{\varepsilon}\|^2/(n\sigma^2) \leq \lambda$, with probability at least $1 - 1/p$.*

*(iii) Let $\mathcal{E}_{ls} = \{\|\hat{\beta}_{ls} - \beta_S\|_\infty > \sigma c_0 \tau_1/\sqrt{n}\}$, where*

$$c_0 = (1-\omega)\left[\tilde{\nu_1} + \sqrt{\frac{1+\nu_1\bar{\eta}}{\lambda_{min}}}\right] \tag{39}$$

*Then $\mathbb{P}(\mathcal{E}^c_{cond} \cap \mathcal{E}_{ls}) \leq (\sqrt{2/\pi})k/(\tau p^{1+a})$, where $\mathcal{E}_{cond}$, here, is the event that (i) or (ii) above fails. From (i) and (ii) it has probability at most $3/p$.*

The above lemma is proved in Section 5. As mentioned before, the $X_j$'s, for $j \in S^c$, are not independent of the $\tilde{R}_i$'s. We get around this by finding the conditional distribution of each $X_j$ given $X_S$ and $\tilde{\varepsilon}$. Correspondingly, each $X_j$ may be represented as a linear combination of columns in $A = [X_S : \tilde{\varepsilon}]$ plus a noise vector, which we call $Z_j$. This noise term is independent of $A$ and hence $\tilde{R}_0, \tilde{R}_1, \ldots, \tilde{R}_m$.

Let $a_j = \Sigma_{SS}^{-1}\Sigma_{Sj}$ and

$$b_j = \frac{e_j^{\mathrm{T}}\Sigma_{S^c|S}\beta_{S^c}}{\sqrt{d}}, \tag{40}$$

where $e_j$ is the $j$th column of the size $p-k$ identity matrix and

$$d = \sigma^2 + \beta_{S^c}^{\mathrm{T}}\Sigma_{S^c|S}\beta_{S^c}. \tag{41}$$

The following lemma characterizes the conditional distribution of $X_j$ given $A$.

**Lemma 11** *Let $a_j, b_j$, for $j \in S^c$, be as above. Then we have the following:*

*(i) The distribution of $X_j$, for $j \in S^c$, may be represented as*

$$X_j \stackrel{\mathcal{D}}{=} X_S a_j + b_j W + Z_j \tag{42}$$

*where $W \sim N(0, I_n)$ and is independent of $X_S$. Further, $Z_j$ is independent of $[X_S : \tilde{\varepsilon}]$ and follows $N(0, \tilde{\sigma}_{jj} I_n)$, with $\tilde{\sigma}_{jj} \leq \sigma_{jj} = 1$.*

*(ii) Define, for $j \in S^c$ and $i = 1, \ldots, m+1$,*

$$V_{ji} = b_j W^{\mathrm{T}} \frac{\tilde{R}_{i-1}}{\|\tilde{R}_{i-1}\|} + E_{ji}, \tag{43}$$

*where $E_{ji} = Z_j^{\mathrm{T}}\tilde{R}_{i-1}/\|\tilde{R}_{i-1}\|$. Let,*

$$\tilde{\mathcal{E}}_f = \left\{\max_{1 \leq i \leq m+1,\, j \in S^c} |V_{ji}| > (1-\omega)\tau_1\right\}.$$

*Then $\mathbb{P}(\tilde{\mathcal{E}}_f) \leq 1/p + (\sqrt{2/\pi})(k+1)/(\tau p^a)$.*

The above lemma is proved in Section 5. We now show that $\mathcal{E}_f \subseteq \tilde{\mathcal{E}}_f$. To see this, notice that on $\tilde{\mathcal{E}}_f^c$ one has,

$$\begin{aligned}\tilde{T}_i &\leq (\max_{j\in S^c}\|a_j\|_1)T_i + (1-\omega)\tau_1,\\ &\leq \omega T_i + (1-\omega)\tau_1,\end{aligned} \tag{44}$$

for $i = 1,\ldots,m+1$. Here, the first inequality follows from using (42) and $\left|a_j^{\mathrm{T}} X_S^{\mathrm{T}} \tilde{R}_{i-1}/\|\tilde{R}_{i-1}\|\right| \leq \|a_j\|_1 T_i$, along with the fact that $|V_{ji}|$ is bounded by $(1-\omega)\tau_1$ on $\tilde{\mathcal{E}}_f^c$. The second inequality follows from (18). We now show that

$$\mathcal{E}' = \left\{\tilde{T}_i \leq \omega T_i + (1-\omega)\tau_1 \ \text{ for each } \ i \leq m+1\right\}$$

implies $\mathcal{E}_f^c$. To see this, for each $i$, consider two cases, viz. $T_i > \tau_1$ and $T_i \leq \tau_1$. From (44), in the first case one has $\tilde{T}_i < T_i$, and in the second case, one has $\tilde{T}_i \leq \tau_1$. Correspondingly, $\mathcal{E}'$ is contained in

$$\{\tilde{T}_i < T_i \text{ or } \tilde{T}_i \leq \tau_1 \quad \text{for each } i \leq m+1\},$$

which is $\mathcal{E}_f^c$. Consequently, $\mathcal{E}_f \subseteq \tilde{\mathcal{E}}_f$. Consequently, $\mathbb{P}(\mathcal{E}_f) \leq 1/p + (\sqrt{2/\pi})(k+1)/(\tau p^a)$ from Lemma 11.

What remains to be seen is that the probability of the event $\mathcal{E}_u$ can be bounded as before. For this we apply Lemma 8 once again. That conditions (i) - (iii), required for application of Lemma 8, are satisfied with high probability is proved parts (i)-(iii) of Lemma 10. Consequently, as before, if $\tilde{\mathcal{E}}_u = \mathcal{E}_{cond} \cup \mathcal{E}_{ls}$, where the sets on the right side are as in Lemma 10, one gets that on $\tilde{\mathcal{E}}_u^c$,

$$\left(1 - \tau_1\sqrt{r_1 k/n}\right)\|\beta_{\hat{F}_1}\| \leq \tilde{r}_2\sigma\tau_1\sqrt{\frac{|\hat{F}_1|}{n}}. \tag{45}$$

Here $\tilde{r}_2 = c_0 + \sqrt{r_1}$, where $c_0$ as in (39). Notice that $\tilde{r}_2 = r_2$, where $r_2$ as in (27). Now, once again use the fact that $n \geq (1+\delta)r_1 k\tau_1^2$ and $n \geq r_2^2 f(\delta)\sigma^2\tau_1^2/\alpha$, to get that (45) implies $\mathcal{E}_u^c$. Accordingly, $\mathbb{P}(\mathcal{E}_u) \leq \mathbb{P}(\tilde{\mathcal{E}}_u)$. Consequently, one has,

$$\begin{aligned}\mathbb{P}(\mathcal{E}) &\leq \mathbb{P}(\mathcal{E}_u \cup \mathcal{E}_f)\\ &\leq \mathbb{P}(\mathcal{E}_{cond}) + \mathbb{P}(\mathcal{E}_{cond}^c \cap \mathcal{E}_{ls}) + \mathbb{P}(\mathcal{E}_f),\end{aligned}$$

which is at most $\tilde{p}_{err,k} = 4/p + (\sqrt{2/\pi}/\tau)\left[(k+1)/p^a + k/p^{1+a}\right]$. This completes the proof for $k \geq 1$.

If $k = 0$, we will show that the probability that $\max_{j\in J}|Z_{1j}|$ exceeds $\tau_1$ is at most $\tilde{p}_{err,0}$. This would imply that the algorithm stops after one step and $\hat{S}$ is empty. Notice that $S^c = J$ and hence $\tilde{\varepsilon} = Y$. Consequently, $X_j \stackrel{\mathcal{D}}{=} \tilde{b}_j Y/\sigma_Y + Z_j$, where $Z_j \sim N(0,\tilde{\sigma}_j)$ is independent of $Y$, with $\tilde{\sigma}_j \leq 1$. Also, $\tilde{b}_j = e_j^{\mathrm{T}}\Sigma\beta/\sigma_Y$, where $\sigma_Y^2 = \mathrm{Var}(Y_1) = \sigma^2 + \beta^{\mathrm{T}}\Sigma\beta$. Correspondingly,

$$Z_{1j} \stackrel{\mathcal{D}}{=} \tilde{b}_j\|Y\|/\sigma_Y + Z_j^{\mathrm{T}}\frac{Y}{\|Y\|} \tag{46}$$

Using $\sigma_Y \geq \sigma$, one has $\tilde{b}_j \leq \nu_1\mu_n$. Further, using $\|Y\|/\sigma_Y \leq (1+\mu_n)$, with probability at least $1-1/p$ from Lemma 14, one has that the first term in the right side of (46) is at most $\nu_1\tau(1+\bar{k}^{-1/2})$ with

probability at least $1-1/p$. Further $|Z_j^{\mathrm{T}}Y/\|Y\||$, using the independence of $Z_j$ and $Y$, is less than $\tau$ for all $j$ with probability at least $1-\sqrt{2/\pi}/(\tau p^a)$ (Lemma 13 (b)). Denoting, $\tau_2=[\nu_1(1+\bar{k}^{-1/2})+1]\tau$, one sees $\max_{j\in J}|\mathcal{Z}_{1j}|\leq\tau_2$, with probability at least $1-\tilde{p}_{err,0}$. Notice that since $\tau_1\geq\tau_2$, the event $\max_{j\in J}|\mathcal{Z}_{1j}|\leq\tau_1$ also has probability at least $1-\tilde{p}_{err,0}$. This completes the proof. ■

**Proof** [Proof of Corollary 5] The proof is exactly similar to that of Corollary 2. As before, taking $\alpha(\delta)=\sigma^2/[(1+\delta)\bar{k}]$ and $\xi(\delta)=\xi(\alpha(\delta),\delta)$, we notice that $\rho^2\xi(\delta^*)\bar{k}\tau^2=\bar{\xi}\bar{k}\log p$, where $\delta^*=3$. Correspondingly, if $n\geq\bar{\xi}\bar{k}\log p$, one has $n=\rho^2\xi(\delta)\bar{k}\tau^2$ for some $\delta\geq\delta^*$ and hence,

$$\hat{S}\subseteq S \quad \text{and} \quad \sum_{j\in\hat{F}}\beta_j^2\leq\alpha(\delta)|\hat{F}|$$

with probability at least $1-\tilde{p}_{err,k}$, from Theorem 4. Further, $\alpha(\delta)$ is at most $r^2\sigma^2\mu_n^2$, using the same reasoning as before. The conclusions on recovering the large coefficients follow immediately from this. ■

**Proof** [Proof of Theorem 6] Notice that,

$$\|\hat{\beta}-\beta\|^2=\|\hat{\beta}_S-\beta_S\|^2+\|\hat{\beta}_{S^c}-\beta_{S^c}\|^2. \tag{47}$$

We apply the result of Corollary 5, to get that except on a set with probability $\tilde{p}_{err,k}$, one has $\hat{S}\subseteq S$. Correspondingly, the second term in (47) is simply $\|\beta_{S^c}\|^2$, which is equal to $\sum_{j\in S^c}\min\{\beta_j^2,\sigma^2\mu_n^2\}$.

Let's next concentrate on the first term in (47). Notice that since $\hat{S}\subseteq S$, one has $\hat{\beta}_S$ is same as the coefficient estimate one would get if the OMP were run on the truncated problem. Correspondingly, using part (b) of Lemma 8, with $\tau_0=\tau_1$ and $\tilde{r}_2=r_2$, one gets that

$$\|\hat{\beta}_S-\beta_S\|\leq\frac{r_2\sigma\tau_1\sqrt{k/n}}{1-\tau_1\sqrt{r_1k/n}}, \tag{48}$$

with probability at least $1-\tilde{p}_{err,k}$. Next, use the fact that $\tau_1\sqrt{k/n}\leq 1/(4r_2)$ using $\bar{\xi}\bar{k}\log p=16r_2^2\bar{k}\tau_1^2$. Consequently, the denominator in the right side of (48) is at least $1-\sqrt{r_1}/4r_2$. The latter is at least $3/4$ using $r_2\geq\sqrt{r_1}$. Thus,

$$\begin{aligned}\|\hat{\beta}_S-\beta_S\|&\leq\frac{4r_2\rho\sqrt{1+a}}{3}\sigma\sqrt{k}\mu_n,\\&=\sqrt{C}\sigma\sqrt{k}\mu_n,\end{aligned}$$

where $C=(4/9)r^2$. Correspondingly, from (47) one gets that,

$$\begin{aligned}\|\hat{\beta}-\beta\|^2&\leq C\sigma^2k\mu_n^2+\sum_{j\in S^c}\min\{\beta_j^2,\sigma^2\mu_n^2\}\\&\leq C\sum_{j=1}^{p}\min\{\beta_j^2,\sigma^2\mu_n^2\},\end{aligned}$$

where the last inequality from using $\sigma^2k\mu_n^2=\sum_{j\in S}\min\{\beta_j^2,\sigma^2\mu_n^2\}$, since $S=\{j:|\beta_j|>\sigma\mu_n\}$. ■

**Proof** [Proof of Corollary 7] For $k$-sparse $\beta$, once again let $S = \{j : |\beta_j| > \sigma\mu_n\}$. Now $\|\beta_{S^c}\|_1 \leq \eta\sigma\mu_n$, where $\eta = \bar{k}$, since there are at most $\bar{k}$ non-zero entries outside of $S$, with magnitude at most $\sigma\mu_n$. Now apply Theorem 6, with $\eta = \bar{k}$ (or $\bar{\eta} = 1$) to get the desired result. ■

## 5. Proof Of Results From Section 4

The following simple lemma will prove useful in proving Lemma 10.

**Lemma 12** *Let $\theta_n = \bar{k}^{1/2}\mu_n$. Conditions (17) - (19) imply the following:*
*(i) Let $d$ be as in (41). Then $d \leq \sigma^2(1+\nu_1\bar{\eta}\,\theta_n^2)$.*
*(ii) $\|\Sigma_{SS}g\|^2 \leq \sigma^2 s_{max}^2 \tilde{\nu_1}^2\theta_n^2$, where $g = \Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}$.*

**Remark:** Since we take $n > 2\bar{k}\log p$, we have $\theta_n \leq 1$. Accordingly, the above bound holds with $\theta_n$ replaced by 1.

**Proof** [Proof of Lemma 12] We first prove part (i). Recall that $d = \sigma^2 + \beta_{S^c}^{\mathrm{T}}\Sigma_{S^c|S}\beta_{S^c}$. Write $\beta_{S^c}^{\mathrm{T}}\Sigma_{S^c|S}\beta_{S^c}$ as $\sum_{j\in S^c}\beta_j e_j^{\mathrm{T}}\Sigma_{S^c|S}\beta_{S^c}$, which can be bounded by $(\|\Sigma_{S^c|S}\beta_{S^c}\|_\infty)\|\beta_{S^c}\|_1$, which is at most $\sigma\nu_1\bar{\eta}\,\theta_n^2$ from (19) and (10). This completes the proof.

For part (ii) use the fact that $\|\Sigma_{SS}g\|^2 \leq s_{max}^2\|g\|^2$ from (17) and $\|g\| \leq \sigma\sqrt{k}\tilde{\nu_1}\mu_n$ from (19), to complete the proof. ■

**Proof** [Proof of Lemma 10] We use a result in Szarek (1991) that gives tails bounds for the largest and smallest singular values of Gaussian random matrices. Let $U \in \mathbb{R}^{n\times k}$ be a matrix with i.i.d. standard Gaussian entries. Then, for $r > 0$, one has,

$$\mathbb{P}(\lambda_k\left(U/\sqrt{n}\right) > 1 + \sqrt{k/n} + r) \leq e^{-nr^2/2}$$

$$\mathbb{P}(\lambda_1\left(U/\sqrt{n}\right) < 1 - \sqrt{k/n} - r) \leq e^{-nr^2/2},$$

where $\lambda_k(.)$ and $\lambda_1(.)$ gives the largest and smallest singular values respectively, of an $n \times k$ matrix. Now, taking $r = \mu_n$, one has, using the above, that with probability at $1 - 2/p$ the following holds:

$$h_\ell\|v\|^2 \leq \frac{1}{n}\|Uv\|^2 \leq h_u\|v\|^2 \quad \text{for all} \quad v \in \mathbb{R}^k.$$

Now, notice that since $X_S \stackrel{\mathcal{D}}{=} U\Sigma_{SS}^{1/2}$, one has from the above that, with probability at least $1 - 2/p$,

$$h_\ell\|\Sigma_{SS}^{1/2}v\|^2 \leq \frac{1}{n}\|X_S v\|^2 \leq h_u\|\Sigma_{SS}^{1/2}v\|^2 \quad \text{for all} \quad v \in \mathbb{R}^k.$$

Correspondingly, from (17), since $s_{min} \leq \|\Sigma_{SS}^{1/2}v\|^2/\|v\|^2 \leq s_{max}$, which implies that, with probability at least $1 - 2/p$,

$$\lambda_{min}\|v\|^2 \leq \frac{1}{n}\|X_S v\|^2 \leq \lambda_{max}\|v\|^2 \quad \text{for all} \quad v \in \mathbb{R}^k,$$

where $\lambda_{min}$, $\lambda_{max}$ as in (26).

Before proving parts (ii) and (iii), observe that by conditioning on $X_S$, the distribution of $\tilde{\varepsilon}$ may be expressed as,

$$\tilde{\varepsilon} \stackrel{\mathcal{D}}{=} X_S g + \sqrt{d} W, \tag{49}$$

where $g = \Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}$ and $d$ as in (41). Here $W \sim N(0, I_n)$ and is independent of $X_S$.

For part (ii), notice that from the above $\tilde{\sigma}^2 := \mathrm{Var}(\tilde{\varepsilon}_1) = \|\Sigma_{SS} g\|^2 + d$, which is at most $\sigma^2(1 + s_{max}^2 \tilde{\nu_1}^2 + \nu_1\bar{\eta})$ from Lemma 12. Further, $\|\tilde{\varepsilon}\|^2/\tilde{\sigma}^2 \sim \mathcal{X}_n^2$. Now from Lemma 14, the probability of the event $\|\tilde{\varepsilon}\|^2/(n\tilde{\sigma}^2) > (1+\mu_n)^2$ is bounded $1/p$. Use $\mu_n \leq \bar{k}^{-1/2}$ and $\tilde{\sigma}^2 \leq \sigma^2(1 + s_{max}^2\tilde{\nu_1}^2 + \nu_1\bar{\eta})$, to get that $\mathbb{P}\left(\|\tilde{\varepsilon}\|^2/(n\sigma^2) > \lambda\right) \leq 1/p$, where $\lambda$ as in (26).

For part (iii), notice that $\hat{\beta}_{ls} - \beta_S = (X_S^{\mathrm{T}} X_S)^{-1} X_S^{\mathrm{T}} \tilde{\varepsilon}$, which using (49), can be expressed as,

$$\hat{\beta}_{ls} - \beta_S \stackrel{\mathcal{D}}{=} g + \sqrt{d}(X_S^{\mathrm{T}} X_S)^{-1} X_S^{\mathrm{T}} W.$$

Let $\tilde{\mathcal{E}}_{ls} = \{\sqrt{d}\|(X_S^{\mathrm{T}} X_S)^{-1} X_S^{\mathrm{T}} W\|_\infty > \sigma\sqrt{1+\nu_1\bar{\eta}}\tau/\sqrt{\lambda_{min} n}\}$. Now, since $W$ is independent of $X_S$, and $d \leq \sigma^2(1+\nu_1\bar{\eta})$, one can use the same logic as in the proof of Lemma 9 to get that, $\mathbb{P}(\mathcal{E}_{cond}^c \cap \tilde{\mathcal{E}}_{ls}) \leq \sqrt{2/\pi}k/(\tau p^{1+a})$. Further, $\|g\|_\infty \leq \sigma\tilde{\nu_1}\mu_n$ using (19), which, using $\mu_n \leq \tau/\sqrt{n}$, is at most $\sigma\tilde{\nu_1}\tau/\sqrt{n}$. Accordingly, on $\mathcal{E}_{cond}^c \cap \tilde{\mathcal{E}}_{ls}^c$, one has,

$$\begin{aligned}
\|\hat{\beta}_{ls} - \beta_S\|_\infty &\leq \sigma\left[\tilde{\nu_1} + \sqrt{\frac{1+\nu_1\bar{\eta}}{\lambda_{min}}}\right]\tau/\sqrt{n}, \\
&= \sigma\frac{c_0}{\sqrt{n}}\frac{\tau}{1-\omega},
\end{aligned}$$

where $c_0$ as in (39). Now use $\tau/(1-\omega) \leq \tau_1$, to get that $\mathbb{P}(\mathcal{E}_{cond}^c \cap \mathcal{E}_{ls}) \leq \sqrt{2/\pi}k/(\tau p^{1+a})$. This completes the proof of the lemma. ■

**Proof** [Proof of Lemma 11] We first prove part (i). Recall, from (49), one has, $\tilde{\varepsilon} \stackrel{\mathcal{D}}{=} X_S g + \sqrt{d}W$, where $g = (\Sigma_{SS})^{-1}\Sigma_{SS^c}\beta_{S^c}$ and $d$ as in (41). Further, $W$ is independent of $X_S$ and follows $N(0, I_n)$. Correspondingly, the conditional distribution of $X_j$ given $[X_S : W]$ may be expressed as,

$$X_j \stackrel{d}{=} X_S a_j + b_j W + Z_j$$

where $a_j = \mathrm{Cov}(X_{1,S}, X_{1j})[\mathrm{Var}(X_{1,S})]^{-1}$ and $b_j = \mathrm{Cov}(X_{1j}, W_1)$. Further, $Z_j \sim N(0, \tilde{\sigma}_{jj} I_n)$ and is independent of $X_S$ and $W$, with

$$\tilde{\sigma}_{jj} = \sigma_{jj} - a_j^{\mathrm{T}}\Sigma_{SS} a_j - b_j^2,$$

which is at most 1. Clearly, the expression for $a_j$ matches that given in the statement of the lemma. Further, from (49), one has that,

$$\mathrm{Cov}(X_{1j}, W_1) = \frac{1}{\sqrt{d}}\left[\mathrm{Cov}(X_{1j}, \tilde{\varepsilon}_1) - \mathrm{Cov}(X_{1j}, X_{1,S} g)\right].$$

Notice that $\mathrm{Cov}(X_{1j}, \tilde{\varepsilon}_1) = \Sigma_{jS^c}\beta_{S^c}$ and $\mathrm{Cov}(X_{1j}, X_{1,S}g) = \mathrm{Cov}(X_{1j}, X_{1,S})g$, which is $\Sigma_{jS}\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}$. Correspondingly, the numerator of the above is $e_j^{\mathrm{T}}\Sigma_{S^c|S}\beta_{S^c}$, and hence, the expression for $b_j$ given above matches that in (40).

We now prove part (ii) of Lemma 11. Firstly, notice that $\max_{j\in S^c}|b_j| \le \nu_1\mu_n$. This follows from observing that $d \ge \sigma^2$, from (41), and also the fact that $|e_j^T\Sigma_{S^c|S}\beta_{S^c}| \le \sigma\nu_1\mu_n$, for all $j \in S^c$, from (19).

Recall the statistic $V_{ji}$ given by (43). One sees that,

$$|V_{ji}| \le |b_j|\|W\| + \left|E_{ji}\right|.$$

Now $\|W\|^2 \sim \chi^2_n$. Correspondingly, from Lemma 14, the event $\{\|W\|/\sqrt{n} > (1+\mu_n)\}$ has probability at most $1/p$.

Further, $Z_j$'s are independent of $[X_S : \tilde{\varepsilon}]$ and, hence, are also independent of $\tilde{R}_0,\ldots,\tilde{R}_m$, since these residuals are functions of $[X_S : \tilde{\varepsilon}]$. Consequently, the $E_{ji}$'s are standard normal random variables; Indeed, conditional on the $\tilde{R}_i$'s, they follow $N(0,1)$, and hence, follow the same distribution unconditionally. Accordingly, using the same logic as in the proof of Theorem 1, the event

$$\left\{\max_{1\le i\le m+1,\, j\in S^c}\left|E_{ji}\right| > \tau\right\} \tag{50}$$

has probability bounded by $\sqrt{2/\pi}(k+1)/(\tau p^a)$.

Consequently, using the bounds on $|b_j|$ and the above, one gets that except on a set with probability $1/p + \sqrt{2/\pi}(k+1)/(\tau p^a)$, one has

$$\max_{1\le i\le m+1,\, j\in S^c}|V_{ji}| \le \nu_1\mu_n\sqrt{n}\,(1+\mu_n)+\tau.$$

Using $\tau \ge \mu_n\sqrt{n}$ and $\mu_n \le \bar{k}^{-1/2}$, the right side of the above is at most $(1-\omega)\tau_1$. This completes the proof of the lemma. ■

## 6. Conclusion

The paper analyzed variable selection for the OMP for random $X$ matrices. We analyzed performance with i.i.d sub-Gaussian designs, which has uses in compressed sensing. We remark that for these i.i.d designs, the analysis carries over for the hard thresholded version of the algorithm, in which, instead of choosing the $j$ which maximizes the $|\mathcal{Z}_{ij}|$'s, one chooses all $j$ satisfying $|\mathcal{Z}_{ij}| > \tau$. It is only when there is some correlation within the rows that we find it advantageous to choose the index which maximizes $|\mathcal{Z}_{ij}|$.

For Gaussian designs, with correlation within rows, we give much more general results. Apart from showing that results similar to that by Wainwright (2009), for exact support recovery, are also possible using the OMP, we show additional recovery properties by relaxing the assumption of exact sparsity to a more realistic assumption of a control over the $\ell_1$-norm of the smaller coefficients. Oracle inequalities for the coefficient estimate also followed easily as a consequence of these results.

As mentioned earlier, one drawback of the analysis is the crude manner in which the probability of event (50), that no terms outside of $S$ are selected, is bounded. This gives rise to the $\sqrt{2/\pi}(k+1)/(\tau p^a)$ term in the expression for $\tilde{p}_{err,k}$ (28), because of which $a$ has to be greater than 1 when $k$ is not negligible compared to $p$. In Fletcher and Rangan (2011), a more careful analysis had been carried out for exact recovery with i.i.d. designs and $\ell_0$-sparse vectors. Their analysis carries over, for the general case analyzed here, by noting that the random variables $E_{ji}$, for $i = 1,\ldots,m+1$,

defined in Lemma 11, has the same covariance structure as a normalized Brownian motion at times $t_1, \ldots, t_{m+1}$, where $t_i = \|\tilde{R}_{i-1}\|^2$. This should improve the probability of the event (50) to something closer to $1/p^a$.

For random designs, we measure the performance after averaging over the distribution of $X$. As mentioned before, this can be contrasted to another method, as done by Candès and Plan (2009) for the Lasso, in which a distribution is assigned to $\beta$ and the performance is measured after averaging over this distribution. Although these two methods do not imply each other, it is interesting to compare the average performance using both methods. To be consistent with their notation, let's assume that the entries of $X$ are scaled so that the columns have norm equal (or nearly equal) to one. Under a mild assumption on the incoherence, it is shown that for $\ell_0$-sparse vectors the support can be recovered, if

$$k = O(p/[|||X|||^2 \log p]), \tag{51}$$

where $|||X|||$ denotes the spectral norm of $X$. If $X$ has i.i.d $N(0,1/n)$ entries, then $|||X||| \approx \sqrt{p/n}$, so that the sparsity requirement (51) would translate to $k = O(n/\log p)$, which is of the same order as what we get here. However, the situation is different in the general case when the rows are i.i.d $N(0,\Sigma/n)$. Then $X$ may be expressed as $\tilde{X}\Sigma^{1/2}$, where $\tilde{X}$ has i.i.d $N(0,1/n)$ entries. Consider the example where $\Sigma_{ii} = 1$ and $\Sigma_{ij} = c/k$, when $i \neq j$, with $c$ appropriately chosen. In this case $|||X||| \approx c' p/\sqrt{nk}$. Consequently, (51) translates to assuming $n = \Omega(p \log p)$. Our results are better in this case, since we only require $\Omega(k \log p)$ observations even for such correlated designs.

An advantage of the work by Candès and Plan (2009) is its applicability to broad classes of deterministic designs. It is unclear at this stage whether such results also hold for the OMP.

## Appendix A. Tail Bounds

A random variable $Z$ is said to be sub-Gaussian with mean 0 and scale $\sigma > 0$, if $\mathrm{E}e^{tZ} \leq e^{t^2\sigma^2/2}$ for each $t \in \mathbb{R}$.

**Lemma 13** *Let $W = (W_j : 1 \leq j \leq n)^{\mathrm{T}}$, with each $W_j$ sub-Gaussian with mean 0 and scale $\sigma_j > 0$. Let $\sigma = \max_j\{\sigma_j\}$. The following hold.*

*(a) Let $h \in \mathbb{R}^n$, with $\|h\| \leq 1$. If the entries of $W$ are independent then $h^{\mathrm{T}}W$ is sub-Gaussian with mean 0 and scale $\sigma$.*

*(b) Let $\rho = \sigma\sqrt{2(1+a)\log p}$ with $a > 0$. Then $P(\max_j |W_j| > \rho) \leq 2n/p^{1+a}$. Further, if the $W_j \sim N(0,\sigma^2)$ then this probability can be bounded by $\sqrt{2/\pi}(\sigma n)/(\rho\, p^{1+a})$.*

**Proof** For part (a), we need to show that $\mathrm{E}\exp\{t\,h^{\mathrm{T}}W\} \leq \exp\{t^2\sigma^2/2\}$. To see this, notice that $\mathrm{E}\exp\{t\,h^{\mathrm{T}}W\} = \mathrm{E}\exp\left\{t^2\sum_{j=1}^n h_j^2\sigma_j^2/2\right\}$, using independence of $W_j$'s. The claim is proved by noticing that $\sum_{j=1}^n h_j^2\sigma_j^2/2 \leq \sigma^2$, using $\|h\| \leq 1$ and $\sigma_j \leq \sigma$.

For part (b), use a Chernoff bound, followed by optimizing the exponent to get that,

$$\mathbb{P}(|W_j| > \rho) \leq 2\exp\left(-\frac{\rho^2}{2\sigma^2}\right).$$

If the $W_j$'s were normal, standard tail bounds (Feller, 1950) reveals that the above bound can be improved to $(2/(\sqrt{2\pi}\rho))\exp\left(-\frac{\rho^2}{2\sigma^2}\right)$. Now use a union bound, along with the fact that $\exp\left(-\frac{\rho^2}{2\sigma^2}\right) =$

$1/p^{1+a}$, to prove the claim. ■

Next we give a simple lemma on chi-square tail bounds, which will be used repeatedly.

**Lemma 14** *Let $W$ follow $N(0, I_n)$. Then*

$$\mathbb{P}\left(\|W\|/\sqrt{n} \geq 1+\mu_n\right) \leq 1/p,$$

*where $\mu_n = \sqrt{(2\log p)/n}$.*

**Proof** Use the fact (Donoho, 2006b) that for $h > 0$, one has

$$\mathbb{P}\left(\|W\|/\sqrt{n} \geq 1+h\right) \leq e^{-nh^2/2}.$$

Substitute $h = \sqrt{(2\log p)/n}$ to get the result. ■

**Corollary 15** *Let $W_j \sim N(0, I_n)$. Then, with probability at least $1 - p^{a-1}$, for each $j = 1, \ldots, n$,*

$$\|W_j\|/\sqrt{n} \leq 1 + \frac{\sqrt{2a\log p}}{\sqrt{n}}.$$

**Proof** Substitute $h = \sqrt{2a\log p}/\sqrt{n}$ in Lemma 14 and take a union bound over the $j$ events. ■

## Appendix B. Proof Of Lemma 8

For convenience, let $S = \{1, \ldots, k\}$. Let $H_j$, $1 \leq j \leq k$ denote the columns of the $H$ matrix. Assume that the algorithm runs for $m$ steps and let $R_1, \ldots, R_{m-1}$ denote the associated residuals. Let $R_0 = Y$. Denote as $\hat{U}_{\mathcal{A}}$, the least square fit when $U$ is regressed on $H_{\mathcal{A}}$. We also denote as $u(i) = S - d(i)$, which corresponds to the terms in $S$ undetected after step $i$. We assume $u(0) = S$ and $\hat{U}_{d(0)} = 0$.

The following lemma is from Zhang (2009a).

**Lemma 16** *(Zhang, 2009a) For each $i$, with $0 \leq i < m$, if $|u(i)| > 0$, then*

$$\max_{j \in u(i)} \left| \frac{H_j^{\mathrm{T}} R_i}{\|H_j\|} \right| \geq \sqrt{\lambda_{min}} \frac{\|\hat{U}_{d(i)} - \hat{U}_S\|}{\sqrt{|u(i)|}},$$

The results is a consequence of Lemmas 6 and 7 in Zhang (2009a, page 566). Using his notation, in our case, $\lambda_{min} = \rho(\bar{F})$, $R_i = Y - X\beta^{(k-1)}$, $\hat{U}_{d(i)} = X\beta^{(k-1)}$, $\hat{U}_S = X\beta_X(\bar{F}, y)$ and $u(i) = \bar{F} - F^{(k-1)}$.

**Lemma 17** *For each $i$, with $0 \leq i \leq m$, one has*

$$\|R_i\|/\sqrt{n} \leq \sqrt{\tilde{\lambda}_{max}}(\|\varphi_{u(i)}\| + \sigma),$$

*where $\tilde{\lambda}_{max} = \max\{\lambda, \lambda_{max}\}$.*

**Proof** [Proof of 17] Write $R_i = (I - \mathcal{P}_i)U$, where here $\mathcal{P}_i$ is the projection matrix for column space of $H_{d(i)}$. Now $U = H_{d(i)}\varphi_{d(i)} + H_{u(i)}\varphi_{u(i)} + \varepsilon$ and $(I - \mathcal{P}_i)H_{d(i)} = 0$. Correspondingly, $R_i = (I - \mathcal{P}_i)[H_{u(i)}\varphi_{u(i)} + \varepsilon]$. Consequently, $\|R_i\| \leq \|H_{u(i)}\varphi_{u(i)}\| + \|\varepsilon\|$, since $\|(I - \mathcal{P}_i)x\| \leq \|x\|$ for any $x \in \mathbb{R}^n$. The result immediately follows from using $\|H_{u(i)}\varphi_{u(i)}\|/\sqrt{n} \leq \sqrt{\lambda_{max}}\|\varphi_{u(i)}\|$ and $\|\varepsilon\|/(\sqrt{n}\sigma) \leq \sqrt{\lambda}$. This completes the proof of the lemma. ■

Now use the fact that $\|H_j\| \geq \sqrt{n}\sqrt{\lambda_{min}}$, to get from Lemma 16 that,

$$\max_{j \in u(i)} \left|H_j^{\mathrm{T}} R_i\right| \geq \sqrt{\frac{n\rho_1}{|u(i)|}}\|\hat{U}_{d(i)} - \hat{U}_S\|,$$

where $\rho_1 = \lambda_{min}^2$. Consequently, using Lemma 17 and the above, one has that,

$$\max_{j \in u(i)} \left|H_j^{\mathrm{T}} \frac{R_i}{\|R_i\|}\right| \geq \sqrt{\frac{n\rho_2}{|u(i)|}} \frac{\|\hat{U}_{d(i)} - \hat{U}_S\|/\sqrt{n}}{\|\varphi_{u(i)}\| + \sigma},$$

where $\rho_2 = \rho_1/\tilde{\lambda}_{max}$. The algorithm continues as long as the left side of the above is at least $\tau_0$. Consequently, following the reasoning in Zhang (2009a), when the algorithm stops, one must have that either $|\hat{F}_2| = 0$ or the right side of the above, with $u(i)$ replaced by $\hat{F}_2$, is at most $\tau_0$. Let's assume that $|\hat{F}_2| > 0$, since otherwise we would have correctly decoded all terms. Correspondingly, we have,

$$\|\hat{U}_{\hat{S}} - \hat{U}_S\|/\sqrt{n} \leq \tau_0\sqrt{\frac{|\hat{F}_2|}{n\rho_2}}(\|\varphi_{\hat{F}_2}\| + \sigma) \tag{52}$$

when the algorithm stops. Now,

$$\|\varphi_{\hat{F}_2}\| \leq \sqrt{|\hat{F}_2|}\|\varphi - \hat{\varphi}_{ls}\|_\infty + \|\hat{\varphi}_{ls} - \hat{\varphi}\|. \tag{53}$$

To see this note that $\|\varphi_{\hat{F}_2}\|$ is bounded by the sum of $\|\varphi_{\hat{F}_2} - \hat{\varphi}_{ls,\hat{F}_2}\|$ and $\|\hat{\varphi}_{ls,\hat{F}_2}\|$, where $\hat{\varphi}_{ls,\hat{F}_2}$ is the sub-vector of $\hat{\varphi}_{ls}$ with indices in $\hat{F}_2$. The first term in the bound is at most $\sqrt{|\hat{F}_2|}\|\varphi - \hat{\varphi}_{ls}\|_\infty$, whereas the second term can be bounded by $\|\hat{\varphi}_{ls} - \hat{\varphi}\|$, since $\hat{\varphi}_j$ is zero for all indices $j$ in $\hat{F}_2$. Now, use the fact that $\|\hat{\varphi}_{ls} - \varphi\|_\infty$ is bounded by $c_0\sigma\tau_0/\sqrt{n}$ along with the fact that $\|\hat{U}_{\hat{S}} - \hat{U}_S\|/\sqrt{n} \geq \sqrt{\lambda_{min}}\|\hat{\varphi} - \hat{\varphi}_{ls}\|$, to get that from (52) and (53) that,

$$\|\varphi_{\hat{F}_2}\| \leq c_0\sigma\tau_0\sqrt{\frac{|\hat{F}_2|}{n}} + \tau_0\sqrt{r_1\frac{|\hat{F}_2|}{n}}(\|\varphi_{\hat{F}_2}\| + \sigma) \tag{54}$$

when the algorithm stops. Here we use that $r_1 = 1/(\lambda_{min}\rho_2)$. One gets from (54) that

$$\left(1 - \tau_0\sqrt{\frac{r_1|\hat{F}_2|}{n}}\right)\|\varphi_{\hat{F}_2}\| \leq \tilde{r}_2\sigma\tau_0\sqrt{|\hat{F}_2|}/\sqrt{n},$$

where $\tilde{r}_2 = c_0 + \sqrt{r_1}$ and $r_1 = 1/\rho$. Using $|\hat{F}_2| \leq k$, the term $\tau_0\sqrt{r_1|\hat{F}_2|/n}$ appearing in the left side of the above can be bounded by $\tau_0\sqrt{r_1 k/n}$. This leads us to (35), which completes the proof of part (a).

For part (b), notice that

$$\|\hat{\varphi} - \varphi\| \leq \sqrt{k}\|\hat{\varphi}_{ls} - \varphi\|_\infty + \|\hat{\varphi}_{ls} - \hat{\varphi}\|. \tag{55}$$

Now use,

$$\|\hat{\varphi}_{ls} - \hat{\varphi}\| \leq \tau_0\sqrt{r_1 k/n}(\|\varphi_{\hat{F}_2}\| + \sigma)$$

along with,

$$\|\varphi_{\hat{F}_2}\| \leq \frac{\tilde{r}_2\sigma\tau_0\sqrt{k/n}}{\left(1 - \tau_0\sqrt{r_1 k/n}\right)},$$

to get, after rearranging, that,

$$\|\hat{\varphi}_{ls} - \hat{\varphi}\| \leq \sigma\tau_0\sqrt{r_1 k/n}\,\frac{(c_0\tau_0\sqrt{k/n} + 1)}{1 - \tau_0\sqrt{r_1 k/n}}.$$

Now use $\|\hat{\varphi}_{ls} - \varphi\|_\infty \leq \sigma c_0\tau_0\sqrt{k/n}$, along with $\tilde{r}_2 = c_0 + \sqrt{r_1}$, to get from (55) and the above that,

$$\|\hat{\varphi} - \varphi\| \leq \frac{\tilde{r}_2\sigma\tau_0\sqrt{k/n}}{\left(1 - \tau_0\sqrt{r_1 k/n}\right)}.$$

This completes the proof of the lemma.

## Appendix C. Proof Of Lemma 3

For a matrix $A \in \mathbb{R}^{n\times m}$, and $a = 1$ or $\infty$, denote as $|||A|||_a = \sup_{v\neq 0}\|Av\|_a/\|v\|_a$. Recall that $|||A|||_1$ is the maximum of the $\ell_1$ norms of the columns, whereas $|||A|||_\infty$ is the maximum of the $\ell_1$ norms of the rows.

We first prove part (i). Use the fact (Cai and Wang, 2011, Lemma 2),

$$1 - \gamma(k-1) \leq s_{min} \leq s_{max} \leq 1 + \gamma(k-1).$$

Now $\gamma \leq \omega_0/(2k)$, since $k \leq \bar{k}$, and hence, the left side of the above is at least $1 - \omega_0/2$ and the right side is at most $1 + \omega_0/2$. Further (Tropp, 2004, Theorem 3.5),

$$\|\Sigma_{SS}^{-1}\Sigma_{Sj}\|_1 \leq \frac{\gamma k}{1 - \gamma(k-1)}.$$

The right side of the above is at most $\omega_0$. Correspondingly, we may take $\omega$ as $\omega_0$.

We next prove part (ii). Use the fact that,

$$\|\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}\|_\infty \leq |||\Sigma_{SS}^{-1}|||_\infty\|\Sigma_{SS^c}\beta_{S^c}\|_\infty. \tag{56}$$

Now as $\Sigma_{SS}^{-1}$ is symmetric, $|||\Sigma_{SS}^{-1}|||_\infty = |||\Sigma_{SS}^{-1}|||_1$; the latter is at most $1/(1-\gamma(k-1))$ from Tropp (2004, Theorem 3.5). Further, $\|\Sigma_{SS^c}\beta_{S^c}\|_\infty \leq \gamma\|\beta_{S^c}\|_1$, which is at most $\sigma\gamma\eta\mu_n$. Correspondingly, from (56), one gets

$$\|\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}\|_\infty \leq \sigma\frac{\gamma\bar{k}}{1 - \gamma(k-1)}\bar{\eta}\mu_n. \tag{57}$$

The right of the above is at most $\sigma\omega_0\bar{\eta}\mu_n$, using the bound on $\gamma$. Further,

$$\|\Sigma_{S^c|S}\|_\infty \leq \|\Sigma_{S^cS^c}\beta_{S^c}\|_\infty + \|\Sigma_{S^cS}\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}\|_\infty. \tag{58}$$

Now, $\|\Sigma_{S^cS^c}\beta_{S^c}\|_\infty \leq \|\beta_{S^c}\|_\infty + \|(\Sigma_{S^cS^c}-I)\beta_{S^c}\|_\infty$. Further, use $\|\beta_{S^c}\|_\infty \leq \sigma\nu\mu_n$ and $\|(\Sigma_{S^cS^c}-I)\beta_{S^c}\|_\infty \leq \gamma\|\beta_{S^c}\|_1$, the right side of which is at most $\sigma\gamma\eta\mu_n$. Also, the second term in (58) can be bounded as follows:

$$\|\Sigma_{S^cS}\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}\|_\infty \leq |||\Sigma_{S^cS}|||_\infty \|\Sigma_{SS}^{-1}\Sigma_{SS^c}\beta_{S^c}\|_\infty.$$

The first term in the right side product is bounded by $\gamma k$, whereas the second term, from (57), is bounded by $\sigma\omega_0\bar{\eta}\mu_n$. Correspondingly, one gets that

$$\|\Sigma_{S^c|S}\beta_{S^c}\|_\infty \leq \sigma\nu\mu_n + \sigma\gamma\eta\mu_n + \sigma\gamma\omega_0\eta\mu_n.$$

Further, using $\gamma\eta + \gamma\eta\omega_0 \leq 2\gamma\eta$, which is at most $\omega_0\bar{\eta}$, one gets the bound on $\|\Sigma_{S^c|S}\beta_{S^c}\|_\infty$.

For $k = 0$, one has $\|\Sigma_{S^cS^c}\beta_{S^c}\|_\infty \leq \nu + \omega_0\bar{\eta}$, which is at most $\nu + \omega_0\bar{\eta}$, from the bound derived above. This completes the proof of the lemma.

## References


R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. The Johnson-Lindenstrauss lemma meets compressed sensing. *Constructive Approximation*, 2007.

A.R. Barron and A. Joseph. Sparse superposition codes: Fast and reliable at rates approaching capacity with gaussian noise. Technical report, Yale University, 2010.

A.R. Barron and A. Joseph. Least squares superposition codes of moderate dictionary size, reliable at rates up to channel capacity. *IEEE Trans. Inform. Theory*, 58:2541 – 2557, 2012.

A.R. Barron, A. Cohen, W. Dahmen, and R.A. DeVore. Approximation and learning by greedy algorithms. *Ann. Statist.*, 36(1):64–94, 2008.

T.T. Cai and L. Wang. Orthogonal matching pursuit for sparse signal recovery with noise. *IEEE Trans. Inform. Theory*, 57(7):4680–4688, 2011.

E. Candès and T. Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *Ann. Statist.*, 35(6):2313–2351, 2007.

E.J. Candès and Y. Plan. Near-ideal model selection by l1 minimization. *Ann. Statist.*, 37(5A): 2145–2177, 2009.

E.J. Candès and T. Tao. Decoding by linear programming. *IEEE Trans. Inform. Theory*, 51(12): 4203–4215, 2005.

E.J. Candès and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inform. Theory*, 52(12):5406–5425, 2006.

D.L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, 2006a.

D.L. Donoho. For most large underdetermined systems of equations, the minimal l1-norm near-solution approximates the sparsest near-solution. *Communications On Pure And Applied Mathematics*, 59(7):907–934, 2006b.

D.L. Donoho, M. Elad, and V.N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Trans. Inform. Theory*, 52(1):6–18, 2006.

W. Feller. An Introduction To Probability Theory And Its Applications. Vol. i. 1950.

A.K. Fletcher and S. Rangan. Orthogonal matching pursuit: A Brownian motion analysis. *IEEE Trans. Sig. Processing*, (99):1–1, 2011.

C. Huang, G.H.L. Cheang, and A.R. Barron. Risk of penalized least squares, greedy selection and l1 penalization for flexible function libraries. *Technical Report*, 2008.

L. Jones. A simple lemma for optimization in a Hilbert space, with application to projection pursuit and neural net training. *Ann. Statist.*, 20:608–613, 1992.

W.S. Lee, P.L. Bartlett, and R.C. Williamson. Efficient agnostic learning of neural networks with bounded fan-in. *IEEE Trans. Inform. Theory*, 42(6):2118–2132, 1996.

S. Mallat and S.M.Z. Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Trans. Signal Processing*, 41:3397–3415, 1993.

N. Meinshausen and P. Buhlmann. High-dimensional graphs and variable selection with the lasso. *Ann. Statist.*, 34(3):1436–1462, 2006.

Y.C. Pati, R. Rezaiifar, and PS Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Conf. Rec. 27th Asilomar Conf. Sig., Sys. and Comput.*, pages 40–44. IEEE, 1993.

S.J. Szarek. Condition numbers of random matrices. *Journal of Complexity*, 7(2):131–149, 1991.

R. Tibshirani. Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, pages 267–288, 1996.

J.A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50(10):2231–2242, 2004.

J.A. Tropp. Just relax: Convex programming methods for identifying sparse signals in noise. *IEEE Trans. Inform. Theory*, 52(3):1030–1051, 2006.

J.A. Tropp and A.C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inform. Theory*, 53(12):4655–4666, 2007.

M.J. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using $\ell_1$-constrained quadratic programming (lasso). *IEEE Trans. Inform. Theory*, 55(5):2183–2202, 2009.

C.H. Zhang and J. Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *Ann. Statist.*, 36(4):1567–1594, 2008.

T. Zhang. On the consistency of feature selection using greedy least squares regression. *J. Mach. Learn. Res.*, 10:555–568, 2009a.

T. Zhang. Some sharp performance bounds for least squares regression with l1 regularization. *Ann. Statist.*, 37(5A):2109–2144, 2009b.

P. Zhao and B. Yu. On model selection consistency of lasso. *J. Mach. Learn. Res.*, 7:2541–2563, 2006.

Journal of Machine Learning Research 14 (2013) 1801-1835 Submitted 9/12; Revised 4/13; Published 7/13

# Random Walk Kernels and Learning Curves for Gaussian Process Regression on Random Graphs

**Matthew J. Urry** MATTHEW.URRY@KCL.AC.UK
**Peter Sollich** PETER.SOLLICH@KCL.AC.UK
*Department of Mathematics*
*King's College London*
*London, WC2R 2LS, U.K.*

**Editor:** Manfred Opper

## Abstract

We consider learning on graphs, guided by kernels that encode similarity between vertices. Our focus is on random walk kernels, the analogues of squared exponential kernels in Euclidean spaces. We show that on large, locally treelike graphs these have some counter-intuitive properties, specifically in the limit of large kernel lengthscales. We consider using these kernels as covariance functions of Gaussian processes. In this situation one typically scales the prior globally to normalise the average of the prior variance across vertices. We demonstrate that, in contrast to the Euclidean case, this generically leads to significant variation in the prior variance across vertices, which is undesirable from a probabilistic modelling point of view. We suggest the random walk kernel should be normalised locally, so that each vertex has the same prior variance, and analyse the consequences of this by studying learning curves for Gaussian process regression. Numerical calculations as well as novel theoretical predictions for the learning curves using belief propagation show that one obtains distinctly different probabilistic models depending on the choice of normalisation. Our method for predicting the learning curves using belief propagation is significantly more accurate than previous approximations and should become exact in the limit of large random graphs.

**Keywords:** Gaussian process, generalisation error, learning curve, cavity method, belief propagation, graph, random walk kernel

## 1. Introduction

*Gaussian processes (GPs)* have become a workhorse for probabilistic inference that has been developed in a wide range of research fields under various guises (see for example Kleijnen, 2009; Handcock and Stein, 1993; Neal, 1996; Meinhold and Singpurwalla, 1983). Their success and wide adoption can be attributed mainly to their intuitive nature and ease of use. They owe their intuitiveness to being one of a large family of kernel methods that implicitly map lower dimensional spaces with non-linear relationships to higher dimensional spaces where (hopefully) relationships are linear. This feat is achieved by using a kernel, which also encodes the types of functions that the GP prefers a priori. The ease of use of GPs is due to the simplicity of implementation, at least in the basic setting, where prior and posterior distributions are both Gaussian and can be written explicitly.

An important question for any machine learning method is how 'quickly' the method can generalise its prediction of a rule to the entire domain of the rule (i.e., how many examples are required to achieve a particular generalisation error). This is encapsulated in the *learning curve*, which traces average error versus number of examples. Learning curves have been studied for a variety of infer-

©2013 Matthew J. Urry and Peter Sollich.

ence methods and are well understood for parametric models (Seung et al., 1992; Amari et al., 1992; Watkin et al., 1993; Opper and Haussler, 1995; Haussler et al., 1996; Freeman and Saad, 1997) but rather less is known for non-parametric models such as GPs. In the case of GP regression, research has predominantly focused on leaning curves for input data from Euclidean spaces (Sollich, 1999a,b; Opper and Vivarelli, 1999; Williams and Vivarelli, 2000; Malzahn and Opper, 2003; Sollich, 2002; Sollich and Halees, 2002; Sollich and Williams, 2005), but there are many domains for which the input data has a discrete structure. One of the simplest cases is the one where inputs are vertices on a graph, with connections on the graph encoding similarity relations between different inputs. Examples could include the internet, social networks, protein networks and financial markets. Such discrete input spaces with graph structure are becoming more important, and therefore so is an understanding of GPs, and machine learning techniques in general, on these spaces.

In this paper we expand on earlier work in Sollich et al. (2009) and Urry and Sollich (2010) and focus on predicting the learning curves of GPs used for regression (where outputs are from the whole real line) on large sparse graphs, using the *random walk kernel* (Kondor and Lafferty, 2002; Smola and Kondor, 2003).

The rest of this paper will be structured as follows. In Section 2 we begin by analysing the random walk kernel, in particular with regard to the dependence on its lengthscale parameter, and study the approach to the fully correlated limit. With a better understanding of the random walk kernel in hand, we proceed in Section 3 to an analysis of the use of the random walk kernel for GP regression on graphs. We begin in Section 3.2 by looking at how kernel normalisation affects the prior probability over functions. We show that the more frequently used global normalisation of a kernel by its average prior variance is inappropriate for the highly location dependent random walk kernel, and suggest normalisation to uniform local prior variance as a remedy. To understand how this affects GP regression using random walk kernels quantitatively, we extend first in Section 3.4 an existing approximation to the learning curve in terms of kernel eigenvalues (Sollich, 1999a; Opper and Malzahn, 2002) to the discrete input case, allowing for arbitrary normalisation. This approximation turns out to be accurate only in the initial and asymptotic regimes of the learning curve.

The core of our analysis begins in Section 4 with the development of an improved learning curve approximation based on belief propagation. We first apply this, in Section 4.1, to the case of globally normalised kernels as originally proposed. The belief propagation analysis for global normalisation also acts as a useful warm-up for the extension to the prediction of learning curves for the locally normalised kernel setting, which we present in Section 4.3. In both sections we compare our predictions to numerical simulations, finding good agreement that improves significantly on the eigenvalue approximation. Finally, to emphasise the distinction between the use of globally and locally normalised kernels in GP regression, we study qualitatively the case of model mismatch, with a GP with a globally normalised kernel as the teacher and a GP with a locally normalised kernel as the student, or visa versa. The resulting learning curves show that the priors arising from the two different normalisations are fundamentally different; the learning curve can become non-monotonic and develop a maximum as a result of the mismatch. We conclude in Section 6 by summarising our results and discussing further potentially fruitful avenues of research.

### 1.1 Main Results

In this paper we will derive three key results; that normalisation of a kernel by its average prior variance leads to a complicated relationship between the prior variances and the local graph structure; that by fixing the scale to be equal everywhere using a local prescription $C_{ij} = \hat{C}_{ij}/\sqrt{\hat{C}_{ii}\hat{C}_{jj}}$ results in a fundamentally different probabilistic model; and that we can derive accurate predictions of the learning curves of Gaussian processes on graphs with a random walk kernel for both normalisations over a broad range of graphs and parameters. The last result is surprising since in continuous spaces this is only possible for a few very restrictive cases.

## 2. The Random Walk Kernel

A wide range of machine learning techniques like Gaussian processes capture prior correlations between points in an input space by mapping to a higher dimensional space, where correlations can be represented by a linear combination of 'features' (see, e.g., Rasmussen and Williams, 2005; Müller et al., 2001; Cristianini and Shawe-Taylor, 2000). Direct calculation of correlations in this high dimensional space is avoided using the 'kernel trick', where the kernel function implicitly calculates inner products in feature space. The widespread use of, and therefore extensive research in, kernel based machine learning has resulted in kernels being developed for a wide range of input spaces (see Genton, 2002, and references therein). We focus in this paper on the class of kernels introduced in Kondor and Lafferty (2002). These make use of the normalised graph Laplacian to define correlations between vertices of a graph.

We denote a generic graph by $G(\mathcal{V},\mathcal{E})$ with a vertex set $\mathcal{V} = \{1,\ldots,V\}$ and edge set $\mathcal{E}$. We encode the connection structure of $G$ using an adjacency matrix $\boldsymbol{A}$, where $A_{ij} = 1$ if vertex $i$ is connected to $j$, and 0 otherwise; we exclude self-loops so that $A_{ii} = 0$. We denote the number of edges connected to vertex $i$, known as the degree, by $d_i = \sum_j A_{ij}$ and define the degree matrix $\boldsymbol{D}$ as a diagonal matrix of the vertex degrees, that is, $D_{ij} = d_i\delta_{ij}$. The class of kernels created in Kondor and Lafferty (2002) is constructed using the *normalised Laplacian*, $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ (see Chung, 1996) as a replacement for the Laplacian in continuous spaces. Of particular interest is the diffusion kernel and its easier to calculate approximation, the random walk kernel. Both of these kernels can be viewed as an approximation to the ubiquitous squared exponential kernel that is used in continuous spaces. The direct graph equivalent of the squared exponential kernel is given by the *diffusion kernel* (Kondor and Lafferty, 2002). It is defined as

$$\boldsymbol{C} = \exp\left(-\frac{1}{2}\sigma^2\boldsymbol{L}\right), \quad \sigma > 0, \tag{1}$$

where $\sigma$ sets the length-scale of the kernel. Unlike in continuous spaces, the exponential in the diffusion kernel is costly to calculate. To avoid this, Smola and Kondor (2003) proposed as a cheaper approximation the *random walk kernel*

$$\boldsymbol{C} = \left(\boldsymbol{I} - a^{-1}\boldsymbol{L}\right)^p = \left((1-a^{-1})\boldsymbol{I} + a^{-1}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\right)^p, \quad a > 2, \quad p \in \mathbb{N}. \tag{2}$$

This gives back the diffusion kernel in the limit $a, p \to \infty$ whilst keeping $p/a = \sigma^2/2$ fixed. The random walk kernel derives its name from its use of random walks to express correlations between

vertices. Explicitly, a binomial expansion of Equation (2) gives

$$\begin{aligned} \boldsymbol{C} &= \sum_{q=0}^{p} \binom{p}{q} (1-a^{-1})^{p-q} (a^{-1})^{q} (\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2})^{q} \\ &= \boldsymbol{D}^{-1/2} \sum_{q=0}^{p} \binom{p}{q} (1-a^{-1})^{p-q} (a^{-1})^{q} (\boldsymbol{A}\boldsymbol{D}^{-1})^{q} \boldsymbol{D}^{1/2}. \end{aligned} \tag{3}$$

The matrix $\boldsymbol{A}\boldsymbol{D}^{-1}$ is a random walk transition matrix: $(\boldsymbol{A}\boldsymbol{D}^{-1})_{ij}$ is the probability of being at vertex $i$ after one random walk step starting from vertex $j$. Apart from the pre- and post-multiplication by $\boldsymbol{D}^{-1/2}$ and $\boldsymbol{D}^{1/2}$, the kernel $\boldsymbol{C}$ is therefore a $q$-step random walk transition matrix, averaged over the number of steps $q$ distributed as $q \sim \text{Binomial}(p, a^{-1})$. Equivalently one can interpret the random walk kernel as a $p$-step lazy random walk, where at each step the walker stays at the current vertex with probability $(1-a^{-1})$ and moves to a neighbouring vertex with probability $a^{-1}$.

Using either interpretation, one sees that $p/a$ is the lengthscale over which the random walk can diffuse along the graph, and hence the lengthscale describing the typical maximum range of the random walk kernel. In the limit of large $p$, where this lengthscale diverges, the kernel should represent full correlation across all vertices. One can see that this is the case by observing that for large $p$, a random walk on a graph will approach its stationary distribution, $\boldsymbol{p}_\infty \propto \boldsymbol{D}\boldsymbol{e}$, $\boldsymbol{e} = (1,\ldots,1)^T$. The $q$-step transition matrix for large $q$ is therefore $(\boldsymbol{A}\boldsymbol{D}^{-1})^q \approx \boldsymbol{p}_\infty \boldsymbol{e}^T = \boldsymbol{D}\boldsymbol{e}\boldsymbol{e}^T$, representing the fact that the random walk becomes stationary independently of the starting vertex. This gives, for $p \to \infty$, the kernel $\boldsymbol{C} \propto \boldsymbol{D}^{1/2}\boldsymbol{e}\boldsymbol{e}^T\boldsymbol{D}^{1/2}$, that is, $C_{ij} \propto d_i^{1/2} d_j^{1/2}$. This corresponds to full correlation across vertices as expected; explicitly, if $\boldsymbol{f}$ is a Gaussian process on the graph with covariance matrix $\boldsymbol{D}^{1/2}\boldsymbol{e}\boldsymbol{e}^T\boldsymbol{D}^{1/2}$, then $\boldsymbol{f} = v\boldsymbol{D}^{1/2}\boldsymbol{e}$ with $v$ a single Gaussian degree of freedom.

We next consider how random walk kernels on graphs approach the fully correlated case, and show that even for 'simple' graphs the convergence to this limit is non-trivial. Before we do so, we note an additional peculiarity of random walk kernels compared to their Euclidean counterparts: in addition to the maximum range lengthscale $p/a$ discussed so far, they have a diffusive lengthscale $\sigma = (2p/a)^{1/2}$, which is suggested for large $p$ and $a$ by the lengthscale of the corresponding diffusion kernel (1). This diffusive lengthscale will appear in our analysis of learning curves in the large $p$-limit Section 3.4.1.

### 2.1 The $d$-Regular Tree: A Concrete Example

To begin our discussion of the dependence of the random walk kernel on the lengthscale $p/a$, we first look at how this kernel behaves on a $d$-regular graph sampled uniformly from the set of all $d$-regular graphs. Here $d$-regular means that all vertices have degree $d_i = d$. For a large enough number of vertices $V$, typical cycles in such a $d$-regular graph are also large, of length $O(\log V)$, and can be neglected for calculation of the kernel when $V \to \infty$. We therefore begin by assuming the graph is an infinite tree, and assess later how the cycles that do exist on random $d$-regular graphs cause departures from this picture.

A $d$-regular tree is a graph where each vertex has degree $d$ with *no cycles*; it is unique up to permutations of the vertices. Since all vertices on the tree are equivalent, the random walk kernel $C_{ij}$ can only depend on the distance between vertices $i$ and $j$, that is, the smallest number of steps on the graph required to get from one vertex to the other. Denoting the value of a $p$-step lazy random walk kernel for vertices a distance $l$ apart by $C_{l,p}$, we can determine these values by recursion over

$p$ as follows:

$$\begin{aligned} C_{l,p=0} &= \delta_{l,0}, \qquad \gamma_{p+1} C_{0,p+1} = \left(1 - \frac{1}{a}\right) C_{0,p} + \frac{1}{a} C_{1,p}, \\ \gamma_{p+1} C_{l,p+1} &= \frac{1}{ad} C_{l-1,p} + \left(1 - \frac{1}{a}\right) C_{l,p} + \frac{d-1}{ad} C_{l+1,p} \quad l \geq 1. \end{aligned} \tag{4}$$

Here $\gamma_p$ is chosen to achieve the desired normalisation of the prior variance for every $p$. We will normalise so that $C_{0,p} = 1$.

Figure 1 (left) shows the results obtained by iterating Equation (4) numerically for a 3-regular tree with $a = 2$. As expected the kernel becomes longer-ranged initially as $p$ is increased, but seems to approach a non-trivial limiting form. This can be calculated analytically and is given by (see Appendix A)

$$C_{l,p\to\infty} = \left[1 + \frac{l(d-2)}{d}\right] \frac{1}{(d-1)^{l/2}}. \tag{5}$$

Equation (5) can be derived by taking the $\sigma^2 \to \infty$ limit of the integral expression for the diffusion kernel from Chung and Yau (1999) whilst preserving normalisation of the kernel (see Appendix A.1 for further details). Alternatively the result (5) can be obtained by rewriting the random walk in terms of shells, that is, grouping vertices according to distance $l$ from a chosen central vertex. The number of vertices in the $l$-th shell, or shell volume, is $v_l = d(d-1)^{l-1}$ for $l \geq 1$ and $v_0 = 1$. Introducing $R_{l,p} = C_{l,p}\sqrt{v_l}$, Equation (4) can be written in the form

$$\begin{aligned} R_{l,p=0} &= \delta_{l,0}, \qquad \gamma_{p+1} R_{0,p+1} = \left(1 - \frac{1}{a}\right) R_{0,p} + \frac{1}{a\sqrt{d}} R_{1,p}, \\ \gamma_{p+1} R_{l,p+1} &= \frac{\sqrt{d-1}}{ad} R_{l-1,p} + \left(1 - \frac{1}{a}\right) R_{l,p} + \frac{\sqrt{d-1}}{ad} R_{l+1,p} \quad l \geq 1. \end{aligned} \tag{6}$$

This is just the un-normalised diffusion equation for a biased random walk on a one dimensional lattice with a reflective boundary at 0. This has been solved in Monthus and Texier (1996), and mapping this solution back to $C_{l,p}$ gives (5) (see Appendix A.2 for further details).

To summarise thus far, the analysis on a $d$-regular tree shows that, for large $p$, the random walk kernel does not approach the expected fully correlated limit: because all vertices have the same degree this limit would correspond to $C_{l,p\to\infty} = 1$. On the other hand, on a $d$-regular graph with any finite number $V$ of vertices, the fully correlated limit must necessarily be approached as $p \to \infty$. As a large regular graph is locally treelike, the difference must arise from the existence of long cycles in a regular graph.

To estimate when the existence of cycles will start to affect the kernel, consider first a $d$-regular tree truncated at depth $l$. This will have $V = 1 + \sum_{i=1}^{l} d(d-1)^{i-1} = O(d(d-1)^{l-1})$ vertices. On a $d$-regular graph with the same number of vertices, we therefore expect to encounter cycles after a number of steps, taken along the graph, of order $l$. In the random walk kernel the typical number of steps is $p/a$, so effects of cycles should appear once $p/a$ becomes larger than

$$\frac{p}{a} \approx \frac{\log(V)}{\log(d-1)}. \tag{7}$$

Figure 1 (right) shows a comparison between $C_{1,p}$ as calculated from Equation (4) for a 3-regular tree and its analogue on random 3-regular graphs of finite size, which we call $K_{1,p}$. We define this

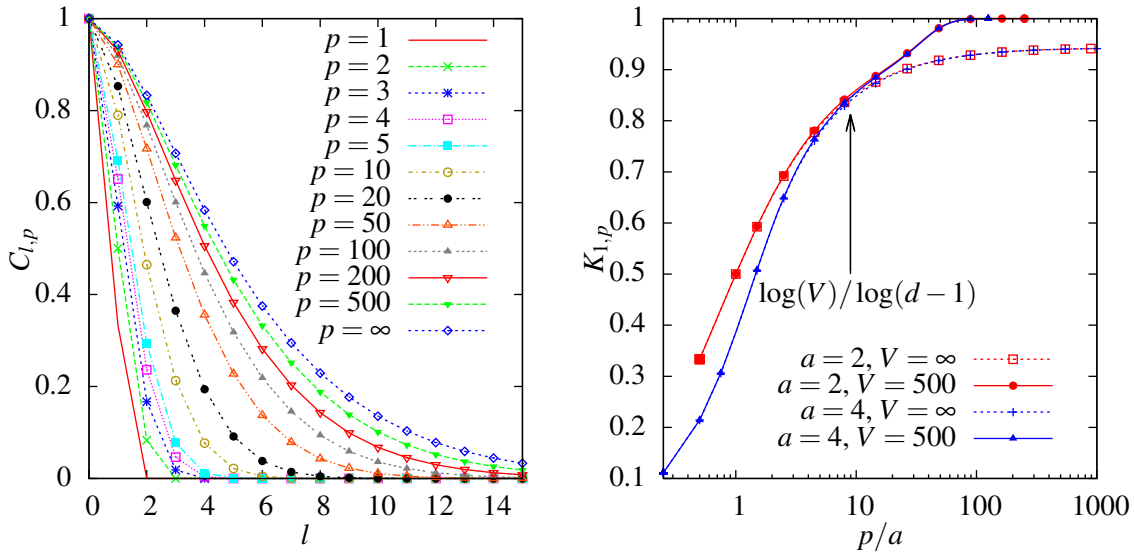



Figure 1: (Left) Random walk kernel $C_{l,p}$ on a 3-regular tree plotted against distance $l$ for increasing number of steps $p$ and $a = 2$. (Right) Comparison between numerical results for the average nearest neighbour kernel $K_{1,p}$ on random 3-regular graphs with the result $C_{1,p}$ on a 3-regular tree, calculated numerically by iteration of (4).

analogue as the average of $C_{ij}/\sqrt{C_{ii}C_{jj}}$ over all pairs of neighbouring vertices on a fixed graph, averaged further over a number of randomly generated regular graphs. The square root accounts for the fact that local kernel values $C_{ii}$ can vary slightly on a regular graph because of cycles, while they are the same for all vertices of a regular tree. Looking at Figure 1 (right) one sees that, as expected from the arguments above, the nearest neighbour kernel value for the 3-regular graph, $K_{1,p}$, coincides with its analogue $C_{1,p}$ on the 3-regular tree for small $p$. When $p/a$ crosses the threshold (7), cycles in the regular graph become important and the two curves separate. For larger $p$, the kernel value for neighbouring vertices approaches the fully correlated limit $K_{1,p} \to 1$ on a regular graph, while on a regular tree one has the non-trivial limit $C_{1,p} \to 2\sqrt{d-1}/d$ from (5).

In conclusion of our analysis of random walk kernels, we have seen that these kernels have an unusual dependence on their lengthscale $p/a$. In particular, kernel values for vertices a short distance apart can remain significantly below the fully correlated limit, even if $p/a$ is large. That limit is approached only once $p/a$ becomes larger than the graph size-dependent threshold (7), at which point cycles become important. We have focused here on random regular graphs, but the same qualitative behaviour should be observed also on graphs with a non-trivial distribution of vertex degrees $d_i$.

## 3. Learning Curves for Gaussian Process Regression

Having reached a better understanding of the random walk kernel we now study its application in machine learning. In particular we focus on the use of the random walk kernel for regression with Gaussian processes. We will begin, for completeness, with an introduction to GPs for regres-

sion. For a more comprehensive discussion of GPs for machine learning we direct the reader to Rasmussen and Williams (2005).

### 3.1 Gaussian Process Regression: Kernels as Covariance Functions

Gaussian process regression is a Bayesian inference technique that constructs a posterior distribution over a function space, $P(f|\boldsymbol{x},\boldsymbol{y})$, given training input locations $\boldsymbol{x} = (x_1,\ldots,x_N)^{\mathrm{T}}$ and corresponding function value outputs $\boldsymbol{y} = (y_1,\ldots,y_N)^{\mathrm{T}}$. The posterior is constructed from a prior distribution $P(f)$ over the function space and the likelihood $P(\boldsymbol{y}|f,\boldsymbol{x})$ to generate the observed output values from function $f$ by using Bayes' theorem

$$P(f|\boldsymbol{x},\boldsymbol{y}) = \frac{P(\boldsymbol{y}|f,\boldsymbol{x})P(f)}{\int \mathrm{d}f' P(\boldsymbol{y}|f',\boldsymbol{x})P(f')}.$$

In the GP setting the prior is chosen to be a Gaussian process, where any finite number of function values has a joint Gaussian distribution, with a covariance matrix with entries given by a *covariance function* or kernel $C(x,x')$ and with a mean vector with entries given by a *mean function* $\mu(x)$. For simplicity we will focus on zero mean GPs[1] and a Gaussian likelihood, which amounts to assuming that training outputs are corrupted by independent and identically distributed Gaussian noise. Under these assumptions all distributions are Gaussian and can be calculated explicitly. If we assume we are given training data $\{(x_\mu,y_\mu)|\mu = 1,\ldots,N\}$ where $y_\mu$ is the value of the target or 'teacher' function at input location $x_\mu$, corrupted by additive Gaussian noise with variance $\sigma^2$, the posterior distribution is then given by another Gaussian process with mean and covariance functions

$$\bar{f}(x) = \boldsymbol{k}(x)^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{y}, \tag{8}$$

$$\mathrm{Cov}(x,x') = C(x,x') - \boldsymbol{k}(x)^{\mathrm{T}}\boldsymbol{K}^{-1}\boldsymbol{k}(x'), \tag{9}$$

where $\boldsymbol{k}(x) = (C(x_1,x),\ldots,C(x_N,x))^{\mathrm{T}}$ and $K_{\mu\nu} = C(x_\mu,x_\nu) + \delta_{\mu\nu}\sigma^2$. With the posterior in the form of a Gaussian process, predictions are simple. Assuming a squared loss function, the optimal prediction of the outputs is given by $\bar{f}(x)$ and a measure of uncertainty in the prediction is provided by $\mathrm{Cov}(x,x)^{1/2}$.

Equations (8) and (9) illustrate that, in the setting of GP regression, kernels are used to change the type of function preferred by the Gaussian process prior, and correspondingly the posterior. The kernel can encode prior beliefs about smoothness properties, lengthscale and expected amplitude of the function we are trying to predict. Of particular importance for the discussion below, $C(x,x)$ gives the prior variance of the function $f$ at input $x$, so that $C(x,x)^{1/2}$ sets the typical function *amplitude* or *scale*.

### 3.2 Kernel Normalisation

Conventionally one fixes the desired scale of the kernel using a *global normalisation*: denoting the unnormalised kernel by $\hat{C}(x,x')$ one scales $C(x,x') = \hat{C}(x,x')/\kappa$ to achieve a desired average of $C(x,x)$ across input locations $x$. In Euclidean spaces one typically uses translationally invariant kernels like the squared exponential kernel. For these, $C(x,x)$ is the same for all input locations $x$ and so global normalisation is sufficient to fix a spatially uniform scale for the prior amplitude. In the case of kernels on graphs, on the other hand, the local connectivity structure around each vertex

1. In the discussion and analysis that follows, generalisation to non-zero mean GPs is straightforward.

can be different. Since information about correlations 'propagates' only along graph edges, graph kernels are not generally translation invariant. In particular, there can be large variation among the prior variances at different vertices. This is usually undesirable in a probabilistic model, unless one has strong prior knowledge to justify such variation. For the random walk kernel, the local prior variances are the diagonal entries of Equation (3). These are directly related to the probability of return of a lazy random walk on a graph, which depends sensitively on the local graph structure. This dependence is in general non-trivial, and not just expressible through, for example, the degree of the local vertex. It seems difficult to imagine a scenario where such a link between prior variances and local graph structures could be justified by prior knowledge.

To emphasise the issue, Figure 2 shows examples of distributions of local prior variances $C_{ii}$ for random walk kernels globally normalised to an average prior variance of unity.[2] The distributions are peaked around the desired value of unity but contain many 'outliers' from vertices with abnormally low or high prior variance. Figure 2 (left) shows the distribution of $C_{ii}$ for a large single instance of an Erdős-Rényi random graph (Erdős and Rényi, 1959). In such graphs, each edge is present independently of all others with some fixed probability, giving a Poisson distribution of degrees $p_\lambda(d) = \lambda^d \exp(-\lambda)/d!$; for the figure we chose average degree $\lambda = 3$. Figure 2 (right) shows analogous results for a generalised random graph with power law mixing distribution (Britton et al., 2006). Generalised random graphs are an extension of Erdős-Rényi random graphs where different edges are assigned different probabilities of being present. By appropriate choice of these probabilities (Britton et al., 2006), one can generate a degree distribution that is a superposition of Poisson distributions, $p(d) = \int \mathrm{d}\lambda\, p_\lambda(d) p(\lambda)$. We have taken a shifted Pareto distribution, $p(\lambda) = \alpha\lambda_m^\alpha/\lambda^{\alpha+1}$ with exponent $\alpha = 2.5$ and lower cutoff $\lambda_m = 2$ for the distribution of the means.

Looking first at Figure 2 (left), we know that large Erdős-Rényi graphs are locally tree-like and hence one might expect that this would lead to relatively uniform local prior variances. As shown in the figure, however, even for such tree-like graphs large variations can exist in the local prior variances. To give some specific examples, the large spike near 0 is caused by single disconnected vertices and the smaller spike at around 6.8 arises from two-vertex (single edge) disconnected subgraphs. Single vertex subgraphs have an atypically small prior variance since, for a single disconnected vertex $i$, before normalisation $C_{ii} = (1-a^{-1})^p$ which is the $q = 0$ contribution from Equation (3). Other vertices in the graph will get additional contributions from $q \geq 1$ and so have a larger prior variance. This effect will become more pronounced as $p$ is increased and the binomial weights assign less weight to the $q = 0$ term.

Somewhat surprisingly at first sight, the opposite effect is seen for two-vertex disconnected subgraphs as shown by the spike around $C_{ii} = 6.8$ in Figure 2 (left). For vertices on such subgraphs, $C_{ii} = \sum_{q=0}^{\lfloor p/2 \rfloor} \binom{p}{2q} a^{-2q}(1-a^{-1})^{p-2q}$, which is an atypically large return probability: after any even number of steps, the walker must always return to its starting vertex. A similar situation would occur on vertices at the centre of a star. This illustrates that local properties of a vertex alone, like its degree, do not constrain the prior variance. In a two-vertex disconnected subgraph both vertices have degree 1. But there will generically be other vertices of degree 1 that are dangling ends of a large connected graph component, and these will not have similarly elevated return probabilities. Thus, local graph structure is intertwined in a complex manner with local prior variance.

The black line in Figure 2 (left) shows theoretical predictions (see Section 4.2) for the prior variance distribution in the large graph limit. There is significant fine structure in the various peaks,

2. We use $C_{ii}$ again here, instead of $C(i,i)$ as in our general discussion of GPs; the subscript notation is more intuitive because the covariance function on a graph is just a $V \times V$ matrix.

on which theory and simulations agree well where the latter give reliable statistics. The decay from the mean is roughly exponential (see linear-log plot in inset), emphasizing that the distribution of local prior variances is not only rather broad but can also have large tails.

For the power law random graph, Figure 2 (right), the broad features of the distribution of local prior variances $C_{ii}$ are similar: a peak at the desired value of unity, overlaid by spikes which again come from single and two-vertex disconnected subgraphs. The inset shows that the tail beyond the mean is roughly exponential again, but with a slower decay; this is to be expected since power law graphs exhibit many more different local structures with a significantly larger probability than is the case for Erdős-Rényi graphs. Accordingly, the distribution of the $C_{ii}$ also has a larger standard deviation than for the Erdős-Rényi case. The maximum values of $C_{ii}$ that we see in these two specific graph instances follow the same trend, with $\max_i C_{ii} \approx 40$ for the power law graph and $\max_i C_{ii} \approx 15$ for the Erdős-Rényi graph. Such large values would constitute rather unrealistic prior assumptions about the scaling of the target function at these vertices.

To summarise, Figure 2 shows that after global normalisation a random walk kernel can retain a large spread in the local prior variances, with the latter depending on the graph structure in a complicated manner. We propose that to overcome this one should use a *local normalisation*. For a desired prior variance $c$ this means normalising according to $C_{ij} = c\hat{C}_{ij}/(\kappa_i\kappa_j)^{1/2}$ with local normalisation constants $\kappa_i = \hat{C}_{ii}$; here $\hat{C}_{ij}$ is the unnormalised kernel matrix as before. This guarantees that all vertices have exactly equal prior variance as in the Euclidean case, that is, all vertices have a prior variance of $c$. No uncontrolled local variation in the scaling of the function prior then remains, and the computational overhead of local over global normalisation is negligible. Graphically, if we were to normalise the kernel to unity according to the local prescription, a plot of prior variances like the one in Figure 2 would be a delta peak centred at 1.

The effect of this normalisation on the behaviour of GP regression is a key question for the remainder of this paper; numerical simulation results are shown in Section 3.3 below, while our theoretical analysis is described in Section 4.

### 3.3 Predicting the Learning Curve

The performance of non-parametric methods such as GPs can be characterised by studying the *learning curve*,

$$\varepsilon(N) = \left\langle \left\langle \left\langle \left\langle \frac{1}{V}\sum_{i=1}^{V} \left(g_i - \langle f_i \rangle_{\boldsymbol{f}|\boldsymbol{x},\boldsymbol{y}}\right)^2 \right\rangle_{\boldsymbol{y}|\boldsymbol{g},\boldsymbol{x}} \right\rangle_{\boldsymbol{g}} \right\rangle_{\boldsymbol{x}} \right\rangle_{\mathcal{G}},$$

defined as the average squared error between the student and teacher's predictions $\boldsymbol{f} = (f_1, \ldots, f_V)^{\mathrm{T}}$ and $\boldsymbol{g} = (g_1, \ldots, g_V)^{\mathrm{T}}$ respectively, averaged over the student's posterior distribution given the data $\boldsymbol{f}|\boldsymbol{x},\boldsymbol{y}$, the outputs given the teacher $\boldsymbol{y}|\boldsymbol{g},\boldsymbol{x}$, the teacher functions $\boldsymbol{g}$, and the input locations $\boldsymbol{x}$. This gives the average generalisation error as a function of the number of training examples. For simplicity we will assume that the input distribution is uniform across the vertices of the graph.

Because we are analysing GP regression on graphs, after the averages discussed so far the generalisation error will still depend on the structure of the specific graph considered. We therefore include an additional average, over all graphs in a random graph ensemble $\mathcal{G}$. We consider graph ensembles defined by the distribution of degrees $d_i$: we specify a degree sequence $\{d_1, \ldots, d_V\}$, or, for large $V$, equivalently a degree distribution $p(d)$, and pick uniformly at random any one of the graphs that has this degree distribution. The actual shape of the degree distribution is left arbitrary,

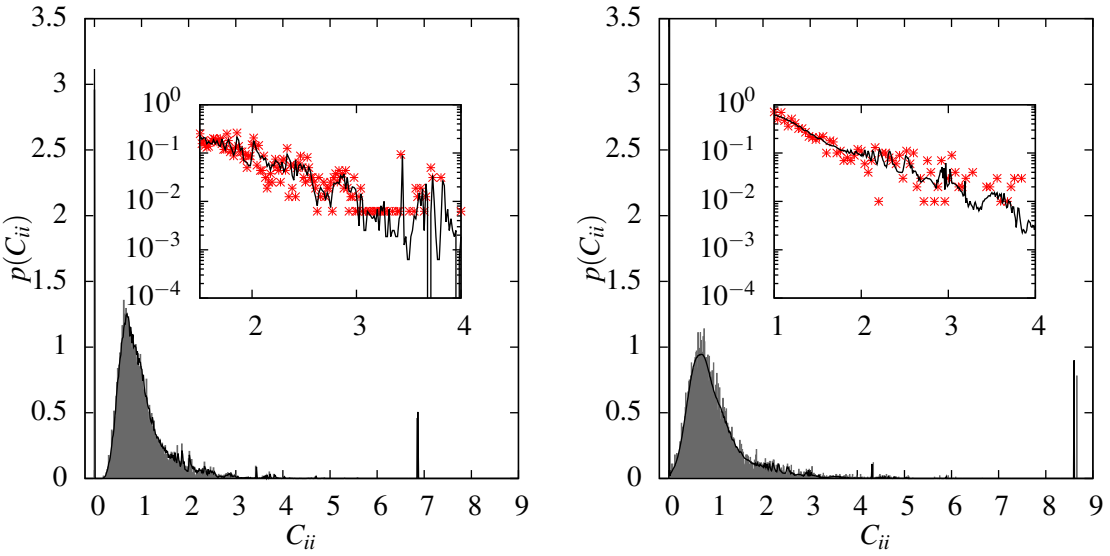



Figure 2: (Left) Grey: histogram of prior variances for the globally normalised random walk kernel with $a = 2$, $p = 10$ on a single instance of an Erdős-Rényi graph with mean degree $\lambda = 3$ and $V = 10000$ vertices. Black: prediction for this distribution in the large graph limit (see Section 4.2). Inset: Linear-log plot of the tail of the distribution. (Right) As (left) but for a power law generalised random graph with exponent 2.5 and cutoff 2.

as long as it has finite mean. Our analysis therefore has broad applicability, including in particular the graph types already mentioned above ($d$-regular graphs, where $p(d') = \delta_{dd'}$, Erdős-Rényi graphs, power law generalised random graphs).

For this paper, as is typical for learning curve studies, we will assume that teacher and student have the same prior distribution over functions, and likewise that the assumed Gaussian noise of variance $\sigma^2$ reflects the actual noise process corrupting the training data. This is known as the *matched case*.[3] Under this assumption the generalisation error becomes the Bayes error, which given that we are considering squared error simplifies to the posterior variance of the student averaged over data sets and graphs (Rasmussen and Williams, 2005). Since we only need the posterior variance we shift $\boldsymbol{f}$ so that the posterior mean is $\mathbf{0}$; $f_i$ is then just the deviation of the function value at vertex $i$ from the posterior mean. The Bayes error can now be written as

$$\varepsilon(N) = \left\langle \left\langle \left\langle \frac{1}{V} \sum_{i=1}^{V} f_i^2 \right\rangle_{\boldsymbol{f}|\boldsymbol{x}} \right\rangle_{\boldsymbol{x}} \right\rangle_{\mathcal{G}} . \tag{10}$$

Note that by shifting the posterior distribution to zero mean, we have eliminated the dependence on $\boldsymbol{y}$ in the above equation. That this should be so can also be seen from (9) for the posterior (co-)variance, which only depends on training inputs $\boldsymbol{x}$ but not the corresponding outputs $\boldsymbol{y}$.

3. The case of mismatch has been considered in Malzahn and Opper (2005) for fixed teacher functions, and for prior and noise level mismatch in Sollich (2002); Sollich and Williams (2005).

The averages in Equation (10) are in general difficult to calculate analytically, because the training input locations $\boldsymbol{x}$ enter in a highly nonlinear matter, see (9); only for very specific situations can exact results be obtained (Malzahn and Opper, 2005; Rasmussen and Williams, 2005). Approximate learning curve predictions have been derived, for Euclidean input spaces, with some degree of success (Sollich, 1999a,b; Opper and Vivarelli, 1999; Williams and Vivarelli, 2000; Malzahn and Opper, 2003; Sollich, 2002; Sollich and Halees, 2002; Sollich and Williams, 2005). We will show that in the case of GP regression for functions defined on graphs, learning curves can be predicted exactly in the limit of large random graphs. This prediction is broadly applicable because the degree distribution that specifies the graph ensemble is essentially arbitrary.

It is instructive to begin our analysis by extending a previous approximation seen in Sollich (1999a) and Malzahn and Opper (2005) to our discrete graph case. In so doing we will see explicitly how one may improve this approximation to fully exploit the structure of random graphs, using belief propagation or equivalently the *cavity method* (Mézard and Parisi, 2003). We will sketch the derivation of the existing approximation following the method of Malzahn and Opper (2005); the result given by Sollich (1999a) is included in this as a somewhat more restricted approximation. Both the approximate treatment and our cavity method take a statistical mechanics approach, so we begin by rewriting Equation (10) in terms of a *generating* or *partition function* $Z$

$$\varepsilon(N) = \left\langle \frac{1}{V} \sum_i \int \mathrm{d}\boldsymbol{f} P(\boldsymbol{f}|\boldsymbol{x}) f_i^2 \right\rangle_{\boldsymbol{x},\mathcal{G}} = -\lim_{\lambda \to 0} \frac{2}{V} \frac{\partial}{\partial \lambda} \langle \log(Z) \rangle_{\boldsymbol{x},\mathcal{G}}, \tag{11}$$

with

$$Z = \int \mathrm{d}\boldsymbol{f} \exp\left( -\frac{1}{2} \boldsymbol{f}^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{f} - \frac{1}{2\sigma^2} \sum_{\mu=1}^{N} f_{x_\mu}^2 - \frac{\lambda}{2} \sum_i f_i^2 \right).$$

In this representation the inputs $\boldsymbol{x}$ only enter $Z$ through the sum over $\mu$. We introduce $\gamma_i$ to count the number of examples at vertex $i$ so that $Z$ becomes

$$Z = \int \mathrm{d}\boldsymbol{f} \exp\left( -\frac{1}{2} \boldsymbol{f}^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{f} - \frac{1}{2} \boldsymbol{f}^{\mathrm{T}} \mathrm{diag}\left( \frac{\gamma_i}{\sigma^2} + \lambda \right) \boldsymbol{f} \right). \tag{12}$$

The average in Equation (11) of the logarithm of this partition function can still not be carried out in closed form. The approximation given by Malzahn and Opper (2005) and our present cavity approach diverge at this point. Section 3.4 discusses the existing approximation for the learning curve, applied to the case of regression on a graph. Section 4 then improves on this using the cavity method to fully exploit the graph structure.

### 3.4 Kernel Eigenvalue Approximation

The approach of Malzahn and Opper (2005) is to average $\log(Z)$ from (12) using the replica trick (Mézard et al., 1987). One writes $\langle \log Z \rangle_{\boldsymbol{x}} = \lim_{n \to 0} \frac{1}{n} \log \langle Z^n \rangle_{\boldsymbol{x}}$, performing the average $\langle Z^n \rangle_{\boldsymbol{x}}$ for integer $n$ and assuming that a continuation to $n \to 0$ is possible. The required $n$-th power of Equation (12) is given by

$$\langle Z^n \rangle_{\boldsymbol{x}} = \int \prod_{a=1}^{n} \mathrm{d}\boldsymbol{f}^a \left\langle \exp\left( -\frac{1}{2} \sum_a (\boldsymbol{f}^a)^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{f}^a - \frac{1}{2\sigma^2} \sum_{i,a} \gamma_i (f_i^a)^2 - \frac{\lambda}{2} \sum_{i,a} (f_i^a)^2 \right) \right\rangle_{\boldsymbol{x}},$$

where the replica index $a$ runs from 1 to $n$. Assuming as before that examples are generated independently and uniformly from $\mathcal{V}$, the data set average over $\boldsymbol{x}$ will, for large $V$, become equivalent to independent Poisson averages over $\gamma_i$ with mean $\nu = N/V$. Explicitly performing these averages gives

$$\langle Z^n \rangle_{\boldsymbol{x}} = \int \prod_{a=1}^{n} \mathrm{d}\boldsymbol{f}^a \exp\left( -\frac{1}{2}\sum_a (\boldsymbol{f}^a)^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{f}^a + \nu \sum_i \left( e^{-\sum_a (f_i^a)^2/2\sigma^2} - 1 \right) - \frac{\lambda}{2} \sum_{i,a} (f_i^a)^2 \right). \quad (13)$$

In order to evaluate (13) one has to find a way to deal with the exponential term in the exponent. Malzahn and Opper (2005) do this using a variational approximation for the distribution of the $\boldsymbol{f}^a$, of Gaussian form. Eventually this leads to the following eigenvalue learning curve approximation (see also Sollich, 1999a):

$$\varepsilon(N) = g\left( \frac{N}{\varepsilon(N) + \sigma^2} \right), \qquad g(h) = \sum_{\alpha=1}^{V} \left( \lambda_\alpha^{-1} + h \right)^{-1}. \quad (14)$$

The eigenvalues $\lambda_\alpha$ of the kernel are defined here from the eigenvalue equation[4] $(1/V)\sum_j C_{ij}\phi_j = \lambda\phi_i$. The Gaussian variational approach is evidently justified for large $\sigma^2$, where a Taylor expansion of the exponential term in (13) can be truncated after the quadratic term. For small noise levels, on the other hand, the Gaussian variational approach will in general not capture all the details of the fluctuations in the numbers of examples $\gamma_i$. This issue is expected to be most prominent for values of $\nu$ of order unity, where fluctuations in the number of examples are most relevant because some vertices will not have seen examples locally or nearby and will have posterior variance close to the prior variance, whereas those vertices with examples will have small posterior variance, of order $\sigma^2$. This effect disappears again for large $\nu$, where the $O(\sqrt{\nu})$ fluctuations in the number of examples at each vertex becomes relatively small. Mathematically this can be seen from the term proportional to $\nu$ in (13), which for large $\nu$ ensures that only values of $f_i^a$ with $\exp(-\sum_a (f_i^a)^2/2\sigma^2)$ close to 1 will contribute. A quadratic approximation is then justified even if $\sigma^2$ is not large.

Learning curve predictions from Equation (14) using numerically computed eigenvalues for the globally normalised random walk kernel are shown in Figure 3 as dotted lines for random regular (left), Erdős-Rényi (centre) and power law generalised random graphs (right). The predictions are compared to numerically simulated learning curves shown as solid lines, for a range of noise levels. Consistent with the discussion above, the predictions of the eigenvalue approximation are accurate where the Gaussian variational approach is justified, that is, for small and large $\nu$. Figure 3 also shows that the accuracy of the approximation improves as the noise level $\sigma^2$ becomes larger, again as expected by the nature of the Gaussian approximation.

### 3.4.1 LEARNING CURVES FOR LARGE $p$

Before moving on to the more accurate cavity prediction of the learning curves, we now look at how the learning curves for GP regression on graphs depend on the kernel lengthscale $p/a$. We focus for this discussion on random regular graphs, where the distinction between global and local normalisation is not important. In Section 2.1, we saw that on a large regular graph the random walk kernel approaches a non-trivial limiting form for large $p$, as long as one stays below the threshold

4. Here and below we consider the case of a uniform distribution of inputs across vertices, though the results can be generalised to the non-uniform case.

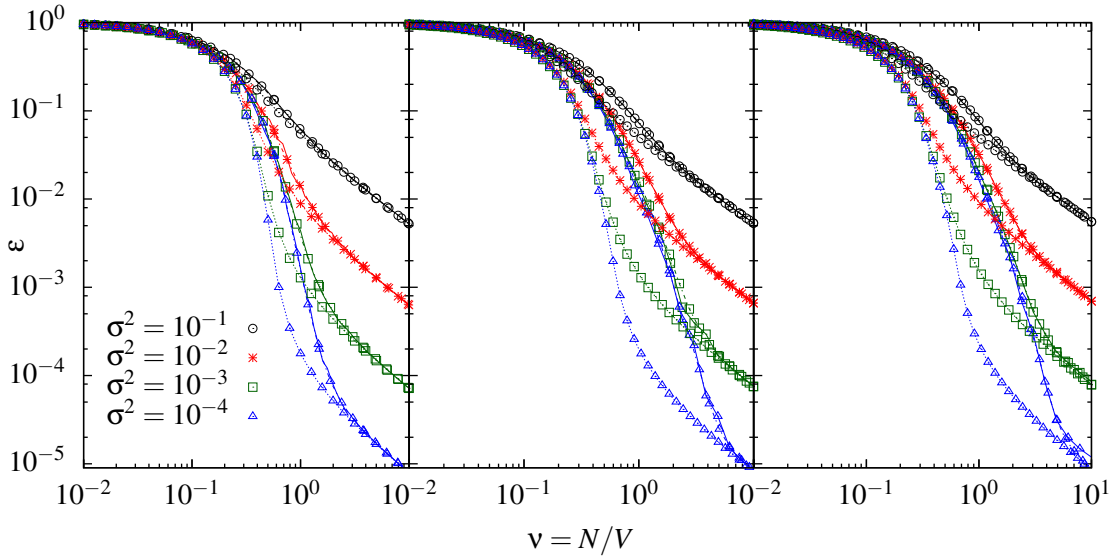



Figure 3: (Left) Learning curves for GP regression with globally normalised kernels with $p = 10$, $a = 2$ on 3-regular random graphs for a range of noise levels $\sigma^2$. Dotted lines: eigenvalue predictions (see Section 3.4), solid lines: numerically simulated learning curves for graphs of size $V = 500$, dashed lines: cavity predictions (see Section 4.1); note these are mostly visually indistinguishable from the simulation results. (Centre) As (left) for Erdős-Rényi random graphs with mean degree 3. (Right) As (left) for power law generalised random graphs with exponent 2.5 and cutoff 2.

(7) for $p$ where cycles become important. One might be tempted to conclude from this that also the learning curves have a limiting form for large $p$. This is too naive however, as one can see by considering, for example, the effect of the first example on the Bayes error. If the example is at vertex $i$, the posterior variance at vertex $j$ is, from (9), $C_{jj} - C_{ij}^2/(C_{ii} + \sigma^2)$. As the prior variances $C_{jj}$ are all equal, to unity for our chosen normalisation, this is $1 - C_{ij}^2/(1+\sigma^2)$. The reduction in the Bayes error is therefore $\varepsilon(0) - \varepsilon(1) = (1/V)\sum_j C_{ij}^2/(1+\sigma^2)$. As long as cycles are unimportant this is independent of the location of the example vertex $i$, and in the notation of Section 2.1 can be written as

$$\varepsilon(0) - \varepsilon(1) = \frac{1}{1+\sigma^2} \sum_{l=0}^{p} \nu_l C_{l,p}^2, \tag{15}$$

where $\nu_l$ is, as before, the number of vertices a distance $l$ away from vertex $i$, that is, $\nu_0 = 1$, $\nu_l = d(d-1)^{l-1}$ for $l \geq 1$. To evaluate (15) for large $p$, one cannot directly plug in the limiting kernel form (5): the 'shell volume' $\nu_l$ just balances the $l$-dependence of the factor $(d-1)^{-l/2}$ from $C_{l,p}$, so that one gets contributions from all distances $l$, proportional to $l^2$ for large $l$. Naively summing up to $l = p$ would give an initial decrease of the Bayes error growing as $p^3$. This is not correct; the reason is that while $C_{l,p}$ approaches the large $p$-limit (5) for any fixed $l$, it does so more and more slowly as $l$ increases. A more detailed analysis, sketched in Appendix A.2, shows that

for large $l$ and $p$, $C_{l,p}$ is proportional to the large $p$-limit $l(d-1)^{-l/2}$ up to a characteristic cutoff distance $l$ of order $p^{1/2}$, and decays quickly beyond this. Summing in (15) the contributions of order $l^2$ up to this distance predicts finally that the initial error decay should scale, non-trivially, as $p^{3/2}$.

We next show that this large $p$-scaling with $p^{3/2}$ is also predicted, for the entire learning curve, by the eigenvalue approximation (14). As before we consider $d$-regular random graphs. The required spectrum of kernel eigenvalues $\lambda_\alpha$ becomes identical, for large $V$, to that on a $d$-regular tree (McKay, 1981). Explicitly, if $\lambda_\alpha^L$ are the eigenvalues of the normalised graph Laplacian on a tree, then the kernel eigenvalues are $\lambda_\alpha = \kappa^{-1}V^{-1}(1-\lambda_\alpha^L/a)^p$. Here the factor $V^{-1}$ comes from the same factor in the kernel eigenvalue definition after (14), and $\kappa$ is the overall normalisation constant which enforces $\sum_\alpha \lambda_\alpha = V^{-1}\sum_j C_{jj} = 1$. The spectrum of the tree Laplacian is known (see McKay, 1981; Chung, 1996) and is given by

$$\rho(\lambda^L) = \begin{cases} \frac{\sqrt{\frac{4(d-1)}{d^2} - (\lambda^L - 1)^2}}{(2\pi/d)\lambda^L(2-\lambda^L)} & \lambda_- \le \lambda \le \lambda_+, \\ 0 & \text{otherwise,} \end{cases}$$

where $\lambda_\pm = 1 \pm \frac{2}{d}(d-1)^{1/2}$. (There are also two isolated eigenvalues at 0 and 2, which do not contribute for large $V$.)

We can now write down the function $g$ from (14), converting the sum over kernel eigenvalues to $V$ times an integral over Laplacian eigenvalues for large $V$. Dropping the $L$ superscript, the result is

$$g(h) = \int_{\lambda_-}^{\lambda_+} \mathrm{d}\lambda\, \rho(\lambda)[\kappa(1-\lambda/a)^{-p} + hV^{-1}]^{-1}. \tag{16}$$

The dependence on $hV^{-1}$ here shows that in the approximate learning curve (14), the Bayes error will depend only on $\nu = N/V$ as might have been expected. The condition for the normalisation factor $\kappa$ becomes simply $g(0) = 1$, or $\kappa^{-1} = \int \mathrm{d}\lambda\, \rho(\lambda)(1-\lambda/a)^p$.

So far we have written down how one would evaluate the eigenvalue approximation to the learning curve on large $d$-regular random graphs, for arbitrary kernel parameters $p$ and $a$. Now we want to consider the large $p$-limit. We show that there is then a *master curve* for the Bayes error against $\nu p^{3/2}$. This is entirely consistent with the $p^{3/2}$ scaling found above for the initial error decay. The intuition for the large $p$ analysis is that the factor $(1-\lambda/a)^p$ decays quickly as the Laplacian eigenvalue $\lambda$ increases beyond $\lambda_-$, so that only values of $\lambda$ near $\lambda_-$ contribute. One can then approximate

$$\left(1-\frac{\lambda}{a}\right)^p \approx \left(1-\frac{\lambda_-}{a}\right)^p \exp\left(-\frac{p(\lambda-\lambda_-)}{a-\lambda_-}\right).$$

Similarly one can replace $\rho(\lambda)$ by its leading square root behaviour near $\lambda_-$,

$$\rho(\lambda) = (\lambda-\lambda_-)^{1/2}\frac{(d-1)^{1/4}d^{5/2}}{\pi(d-2)^2}.$$

Substituting these approximations into (16) and introducing the rescaled integration variable $y = p(\lambda-\lambda_-)/(a-\lambda_-)$ gives

$$g(h) = r\kappa^{-1}(1-\lambda_-/a)^p \left(\frac{a-\lambda_-}{p}\right)^{3/2} F(h\kappa^{-1}V^{-1}(1-\lambda_-/a)^p),$$

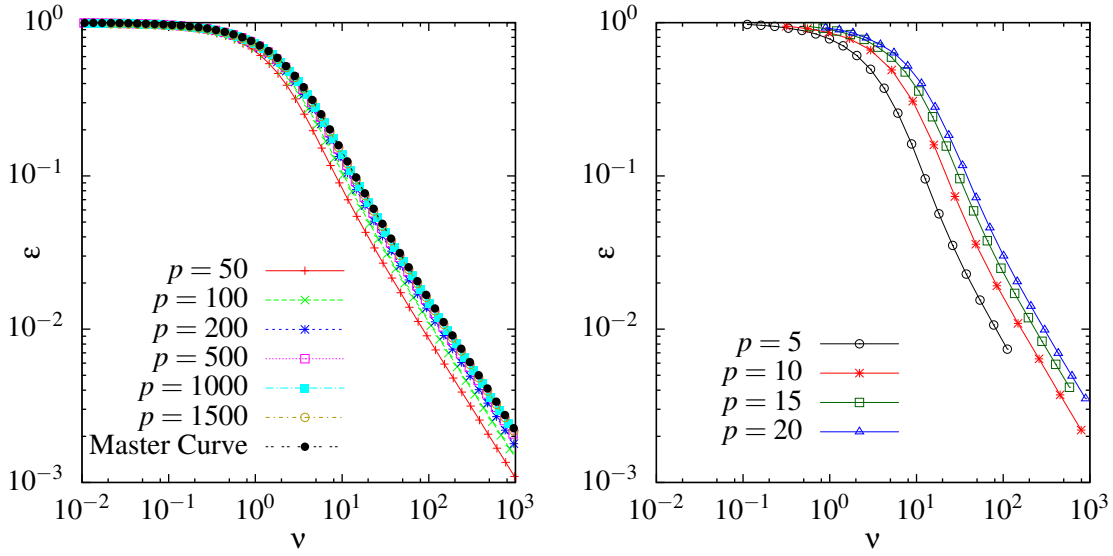



Figure 4: (Left) Eigenvalue approximation for learning curves on a random 3-regular graph, using a random walk kernel with $a = 2$, $\sigma^2 = 0.1$ and increasing values of $p$ as shown. Plotting against $\nu p^{3/2}$ shows that for large $p$ these rescaled curves approach the master curve predicted from (17), though this approach is slower in the tail of the curves. (Right) As (left), but for numerically simulated learning curves on graphs of size $V = 500$.

where $r = (d-1)^{1/4} d^{5/2}/(\pi(d-2)^2)$ and $F(z) = \int_0^\infty dy\, y^{1/2}(\exp(y)+z)^{-1}$. Since $g(0) = 1$, the prefactor must equal $1/F(0) = 2/\sqrt{\pi}$. This fixes the normalisation constant $\kappa$, and we can simplify to

$$g(h) = \frac{F(hV^{-1}c^{-1})}{F(0)}, \qquad c = rF(0)\left(\frac{a-\lambda_-}{p}\right)^{3/2}.$$

The learning curves for large $p$ are then predicted from (14) by solving

$$\varepsilon = F(\nu c^{-1}/(\varepsilon+\sigma^2))/F(0), \tag{17}$$

and depend clearly only on the combination $\nu c^{-1}$. Because $c$ is proportional to $p^{-3/2}$, this shows that learning curves for different $p$ should collapse onto a master curve when plotted against $\nu p^{3/2}$.

A plot of the scaling of the eigenvalue learning curve approximations onto the master curve is shown in Figure 4 (left). As can be seen, large values of $p$ are required in order to get a good collapse in the tail of the learning curve prediction, whereas in the initial part the $p^{3/2}$ scaling is accurate already for relatively small $p$.

Finally, Figure 4 (right) shows that the predicted $p^{3/2}$-scaling of the learning curves is present not only within the eigenvalue approximation, but also in the actual learning curves. Figure 4 (right) displays numerically simulated learning curves for $p = 5, 10, 15$ and 20, against the rescaled number of examples $\nu p^{3/2}$ as before. Even for these comparatively small values of $p$ one sees that the rescaled learning curves approach a master curve.

## 4. Exact Learning Curves: Cavity Method

So far we have discussed the eigenvalue approximation of GP learning curves, and how it deviates from numerically exact simulated learning curves. As discussed in Section 3.4, the deficiencies of the eigenvalue approximation can be traced back to the fact that the fluctuations in the number of training examples seen at each vertex of the graph cannot be accounted for in detail. If in the average over data sets these fluctuations could be treated exactly, one would hope to obtain exact, or at least very accurate, learning curve predictions. In this section we show that this is indeed possible in the case of a random walk kernel, for both global and local normalisations. We derive our prediction using belief propagation or, equivalently, the cavity method (Mézard and Parisi, 2003). The approach relies on the fact that the local structure of the graph on which we are learning is tree-like. This local tree-like structure always occurs in large random graphs sampled uniformly from an ensemble specified by an arbitrary but fixed degree distribution, which is the scenario we consider here. We will see that already for moderate graph sizes of $V = 500$, our predictions are nearly indistinguishable from numerical simulations.

In order to apply the cavity method to the problem of predicting learning curves we must first rewrite the partition function (12) in the form of a graphical model. This means that the function being integrated over to obtain $Z$ must consist of factors relating only to individual vertices, or to pairs of neighbouring vertices. The inverse of the covariance matrix in (12) creates factors linking vertices at arbitrary distances along the graph, and so must be eliminated before the cavity method can be applied. We begin by assuming a general form for the normalisation of $\hat{C}$ that encompasses both local and global normalisation and set $\boldsymbol{C} = \mathcal{K}^{-1/2}[(1-a^{-1})\boldsymbol{I} + a^{-1}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}]^p\mathcal{K}^{-1/2}$ with $\mathcal{K}_{ij} = \kappa_i\delta_{ij}$. To eliminate interactions across the entire graph we first Fourier transform the prior term $\exp(-\frac{1}{2}\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{f})$ in (12), introduce Fourier variables $\boldsymbol{h}$, and then integrate out the remaining terms with respect to $\boldsymbol{f}$ to give

$$Z \propto \prod_i \left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-1/2} \int \mathrm{d}\boldsymbol{h} \exp\left(-\frac{1}{2}\boldsymbol{h}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{h} - \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}\mathrm{diag}\left(\left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-1/2}\right)\boldsymbol{h}\right).$$

The coupling between different vertices in (4) is now through $\boldsymbol{C}$ so still links vertices up to distance $p$. To reduce these remaining interactions to ones among nearest neighbours only, one exploits the binomial expansion of the random walk kernel given in (3). Defining $p$ additional variables at each vertex as $\boldsymbol{h}^q = \mathcal{K}^{1/2}(\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2})^q\mathcal{K}^{-1/2}\boldsymbol{h}$, $q = 1,\ldots,p$, and abbreviating $c_q = \binom{p}{q}(1-a^{-1})^{p-q}(a^{-1})^q$, the interaction term $\boldsymbol{h}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{h}$ turns into a local term $\sum_{q=0}^p c_q(\boldsymbol{h}^0)^{\mathrm{T}}\mathcal{K}^{-1}\boldsymbol{h}^q$. (Here we have, for the sake of uniformity, written $\boldsymbol{h}^0$ instead of $\boldsymbol{h}$.) Of course the interactions have only been 'hidden' in the $\boldsymbol{h}^q$, but the key point is that the definition of these additional variables can be enforced recursively, via $\boldsymbol{h}^q = \mathcal{K}^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\mathcal{K}^{-1/2}\boldsymbol{h}^{q-1}$. We represent this definition via a Dirac delta function (for each $q = 1,\ldots,p$) and then Fourier transform the latter, with conjugate variables $\hat{\boldsymbol{h}}^q$, to get

$$Z \propto \prod_i \left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-1/2} \int \prod_{q=0}^p \mathrm{d}\boldsymbol{h}^q \prod_{q=1}^p \mathrm{d}\hat{\boldsymbol{h}}^q \exp\left(-\frac{1}{2}(\boldsymbol{h}^0)^{\mathrm{T}}\mathrm{diag}\left(\left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-1}\right)\boldsymbol{h}^0 - \frac{1}{2}\sum_{q=0}^p c_q(\boldsymbol{h}^0)^{\mathrm{T}}\mathcal{K}^{-1}\boldsymbol{h}^q + \mathrm{i}\sum_{q=1}^p(\hat{\boldsymbol{h}}^q)^{\mathrm{T}}\left(\boldsymbol{h}^q - \mathcal{K}^{1/2}\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\mathcal{K}^{-1/2}\boldsymbol{h}^{q-1}\right)\right). \quad (18)$$

Because the graph adjacency matrix $\boldsymbol{A}$ now appears at most linearly in the exponent, all interactions are between nearest neighbours only. We have thus expressed our $Z$ as the partition function of a (complex-valued) graphical model.

### 4.1 Global Normalisation

We can now apply belief propagation to the calculation of marginals for the above graphical model. We focus first on the simpler case of a globally normalised kernel where $\kappa_i = \kappa$ for all $i$. Rescaling each $h_i^q$ to $d_i^{1/2}\kappa^{1/2}h_i^q$ and $\hat{h}_i^q$ to $d_i^{1/2}\hat{h}_i^q/\kappa^{1/2}$ we are left with

$$Z \propto \prod_i \left(\frac{\gamma_i}{\sigma^2}+\lambda\right)^{-1/2} \int \prod_{q=0}^{p} \mathrm{d}\boldsymbol{h}^q \prod_{q=1}^{p} \mathrm{d}\hat{\boldsymbol{h}}^q \prod_i \exp\left(-\frac{1}{2}\sum_{q=0}^{p} c_q h_i^0 h_i^q d_i - \frac{1}{2}\frac{(h_i^0)^2\kappa d_i}{\gamma_i/\sigma^2+\lambda} + \mathrm{i}\sum_{q=1}^{p} d_i \hat{h}_i^q h_i^q\right) \prod_{(i,j)} \exp\left(-\mathrm{i}\sum_{q=1}^{p}\left(\hat{h}_i^q h_j^{q-1} + \hat{h}_j^q h_i^{q-1}\right)\right), \quad (19)$$

where the interaction terms coming from the adjacency matrix, $\boldsymbol{A}$, have been written explicitly as a product over distinct graph edges $(i,j)$.

To see how the Bayes error (10) can be obtained from this partition function, we differentiate $\log(Z)$ with respect to $\lambda$ as prescribed by (11) to get

$$\varepsilon(\nu) = \lim_{\lambda\to 0} \frac{1}{V}\sum_i \frac{1}{\gamma_i/\sigma^2+\lambda}\left(1 - \frac{d_i\kappa\langle (h_i^0)^2\rangle}{\gamma_i/\sigma^2+\lambda}\right). \quad (20)$$

In order to calculate the Bayes error we therefore require specifically the marginal distributions of $h_i^0$. These can be calculated using the cavity method: for a large random graph with arbitrary fixed degree sequence the graph is locally tree-like, so that if vertex $i$ were eliminated the corresponding subgraphs (locally trees) rooted at the neighbours $j \in \mathcal{N}(i)$ of $i$ would become approximately independent. The resulting cavity marginals created by removing $i$, which we denote $P_j^{(i)}(\boldsymbol{h}_j, \hat{\boldsymbol{h}}_j|\boldsymbol{x})$, can then be calculated iteratively within these subgraphs using the update equations

$$P_j^{(i)}(\boldsymbol{h}_j, \hat{\boldsymbol{h}}_j|\boldsymbol{x}) \propto \exp\left(-\frac{1}{2}\sum_{q=0}^{p} c_q d_j h_j^0 h_j^q - \frac{1}{2}\frac{d_j\kappa(h_j^0)^2}{\gamma_j/\sigma^2+\lambda} + \mathrm{i}\sum_{q=1}^{p} d_j \hat{h}_j^q h_j^q\right) \int \prod_{k\in\mathcal{N}(j)\setminus i} \mathrm{d}\boldsymbol{h}_k \mathrm{d}\hat{\boldsymbol{h}}_k \exp\left(-\mathrm{i}\sum_{q=1}^{p}(\hat{h}_j^q h_k^{q-1} + \hat{h}_k^q h_j^{q-1})\right) P_k^{(j)}(\boldsymbol{h}_k, \hat{\boldsymbol{h}}_k|\boldsymbol{x}). \quad (21)$$

where $\boldsymbol{h}_j = (h_j^0, \ldots, h_j^p)^{\mathrm{T}}$ and $\hat{\boldsymbol{h}}_j = (\hat{h}_j^1, \ldots, \hat{h}_j^p)^{\mathrm{T}}$. In terms of the sum-product formulation of belief propagation, the cavity marginal on the left is the message that vertex $j$ sends to the factor in $Z$ for edge $(i,j)$ (Bishop, 2007).

One sees that the cavity update Equations (21) are solved self-consistently by complex-valued Gaussian distributions with mean zero and covariance matrices $\boldsymbol{V}_j^{(i)}$. This Gaussian character of the solution was of course to be expected because in (19) we have a Gaussian graphical model. By performing the Gaussian integrals in the cavity update equations explicitly, one finds for the corresponding updates of the covariance matrices the rather simple form

$$\boldsymbol{V}_j^{(i)} = (\boldsymbol{O}_j - \sum_{k\in\mathcal{N}(j)\setminus i} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X})^{-1}, \quad (22)$$

where we have defined the $(2p+1)\times(2p+1)$ matrices

$$\boldsymbol{O}_j = d_j \left(\begin{array}{c|ccc|ccc} c_0+\frac{\kappa}{\gamma_j/\sigma^2+\lambda} & \frac{c_1}{2} & \dots & \frac{c_p}{2} & 0 & \dots & 0 \\ \frac{c_1}{2} & & & & -\mathrm{i} & & \\ \vdots & & & & & \ddots & \\ \frac{c_p}{2} & & & & & & -\mathrm{i} \\ \hline 0 & -\mathrm{i} & & & & & \\ \vdots & & \ddots & & & \mathbf{0}_{p,p} & \\ 0 & & & -\mathrm{i} & & & \end{array}\right), \quad \boldsymbol{X} = \left(\begin{array}{ccc|ccc} & & & \mathrm{i} & & \\ & \mathbf{0}_{p+1,p+1} & & & \ddots & \\ & & & & & \mathrm{i} \\ \hline & & & 0 & \dots & 0 \\ \mathrm{i} & & 0 & & & \\ & \ddots & \vdots & & \mathbf{0}_{p,p} & \\ & \mathrm{i} & 0 & & & \end{array}\right). \quad (23)$$

At first glance (22) becomes singular for $\gamma_j = 0$; however this is easily avoided. We introduce $\boldsymbol{O}_j - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X} = \boldsymbol{M}_j + [d_j\kappa/(\gamma_j/\sigma^2+\lambda)]\boldsymbol{e}_0\boldsymbol{e}_0^T$ with $\boldsymbol{e}_0^T = (1,0,\dots,0)$ so that $\boldsymbol{M}_j$ contains all the non-singular terms. We may then apply the Woodbury identity (Hager, 1989) to write the matrix inverse in a form where the $\lambda \to 0$ limit can be taken without difficulties:

$$\left(\boldsymbol{O}_j - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k^{(j)}\boldsymbol{X}\right)^{-1} = \boldsymbol{M}_j^{-1} - \frac{\boldsymbol{M}_j^{-1}\boldsymbol{e}_0\boldsymbol{e}_0^T\boldsymbol{M}_j^{-1}}{(\gamma_j/\sigma^2+\lambda)/(d_j\kappa) + \boldsymbol{e}_0^T\boldsymbol{M}_j^{-1}\boldsymbol{e}_0}.$$

In our derivation so far we have assumed a fixed graph, we therefore need to translate these equations to the setting we ultimately want to study, that is, an ensemble of large random graphs. This ensemble is characterised by the distribution $p(d)$ of the degrees $d_i$, so that every graph that has the desired degree distribution is assigned equal probability. Instead of individual cavity covariance matrices $\boldsymbol{V}_j^{(i)}$, one must then consider their probability distribution $W(\boldsymbol{V})$ across all edges of the graph. Picking at random an edge $(i,j)$ of a graph, the probability that vertex $j$ will have degree $d_j$ is then $p(d_j)d_j/\bar{d}$, because such a vertex has $d_j$ 'chances' of being picked. (The normalisation factor is the average degree $\bar{d} = \sum_i p(d_i)d_i$.) Using again the locally treelike structure, the incoming (to vertex $j$) cavity covariances $\boldsymbol{V}_k^{(j)}$ will be independent and identically distributed samples from $W(\boldsymbol{V})$. Thus a fixed point of the cavity update equations corresponds to a fixed point of an update equation for $W(\boldsymbol{V})$:

$$W(\boldsymbol{V}) = \sum_d \frac{p(d)d}{\bar{d}} \left\langle \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k W(\boldsymbol{V}_k)\, \delta\left(\boldsymbol{V} - \left(\boldsymbol{O} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1}\right)\right\rangle_\gamma . \quad (24)$$

Since the vertex label is now arbitrary, we have omitted the index $j$. The average in (24) is over the distribution of the number of examples $\gamma \equiv \gamma_j$ at vertex $j$. As before we assume for simplicity that examples are drawn with uniform input probability across all vertices, so that the distribution of $\gamma$ is simply $\gamma \sim \text{Poisson}(\nu)$ in the limit of large $N$ and $V$ at fixed $\nu = N/V$.

In general Equation (24)—which can also be formally derived using the replica approach (see Urry and Sollich, 2012)—cannot be solved analytically, but we can tackle it numerically using population dynamics (Mézard and Parisi, 2001). This is an iterative technique where one creates a population of covariance matrices and for each iteration updates a random element of the population according to the delta function in (24). The update is calculated by sampling from the degree distribution $p(d)$ of local degrees, the Poisson distribution of the local number of examples $\nu$ and

from the distribution $W(\boldsymbol{V}_k)$ of 'incoming' covariance matrices, the latter being approximated by uniform sampling from the current population.

Once we have $W(\boldsymbol{V})$, the Bayes error can be found from the graph ensemble version of Equation (11). This is obtained by inserting the explicit expression for $\langle (h_i^0)^2 \rangle$ in terms of the cavity marginals of the neighbouring vertices, and replacing the average over vertices with an average over degrees $d$:

$$\varepsilon(\nu) = \lim_{\lambda \to 0} \sum_d p(d) \left\langle \frac{1}{\gamma/\sigma^2 + \lambda} \left( 1 - \frac{d\kappa}{\gamma/\sigma^2 + \lambda} \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k W(\boldsymbol{V}_k) \, (\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X})^{-1}_{00} \right) \right\rangle_\gamma . \quad (25)$$

The number of examples at the vertex $\gamma$ is once more to be averaged over $\gamma \sim \mathrm{Poisson}(\nu)$. The subscript '00' indicates the top left element of the matrix, which determines the variance of $h^0$.

To be able to use Equation (25), we again need to rewrite it into a form that remains explicitly non-singular when $\gamma = 0$ and $\lambda \to 0$. We separate the $\gamma$-dependence of the matrix inverse again and write, in slightly modified notation as appropriate for the graph ensemble case, $\boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X} = \boldsymbol{M}_d + [d\kappa/(\gamma/\sigma^2 + \lambda)]\boldsymbol{e}_0\boldsymbol{e}_0^T$, where $\boldsymbol{e}_0^T = (1, 0, \ldots, 0)$. The 00 element of the matrix inverse appearing above can then be expressed using the Woodbury formula (Hager, 1989) as

$$\boldsymbol{e}_0^{\mathrm{T}} \left( \boldsymbol{O} - \sum_{k=1}^{d} \boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X} \right)^{-1} \boldsymbol{e}_0 = \boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{M}_d^{-1} \boldsymbol{e}_0 - \frac{\boldsymbol{e}_0^{\mathrm{T}} \boldsymbol{M}_d^{-1} \boldsymbol{e}_0 \boldsymbol{e}_0^T \boldsymbol{M}_d^{-1} \boldsymbol{e}_0}{(\gamma/\sigma^2 + \lambda)/(d\kappa) + \boldsymbol{e}_0^T \boldsymbol{M}_d^{-1} \boldsymbol{e}_0}.$$

The $\lambda \to 0$ limit can now be taken, with the result

$$\varepsilon(\nu) = \left\langle \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_k W(\boldsymbol{V}_k) \, \frac{1}{\gamma/\sigma^2 + d\kappa (\boldsymbol{M}_d^{-1})_{00}} \right\rangle_\gamma . \quad (26)$$

This has a simple interpretation: the cavity marginals of the neighbours provide an effective Gaussian prior for each vertex, whose inverse variance is $d\kappa(\boldsymbol{M}^{-1})_{00}$.

The self-consistency Equation (24) for $W(\boldsymbol{V})$ and the expression (26) for the resulting Bayes error allow us to predict learning curves as a function of the number of examples per vertex, $\nu$, for *arbitrary degree distributions* $p(d)$ of our random graph ensemble. For large graphs the predictions should become exact. It is worth stressing that such exact learning curve predictions have previously only been available in very specific, noise-free, GP regression scenarios, while our result for GP regression on graphs is applicable to a broad range of random graph ensembles, with arbitrary noise levels and kernel parameters.

We note briefly that for graphs with isolated vertices ($d = 0$), one has to be slightly careful: already in the definition of the covariance function (2) one should replace $\boldsymbol{D} \to \boldsymbol{D} + \delta\boldsymbol{I}$ to avoid division by zero, taking $\delta \to 0$ at the end. For $d = 0$ one then finds in the expression (26) that $(\boldsymbol{M}^{-1})_{00} = 1/(c_0\delta)$, where $c_0$ is defined before (18). As a consequence, $\kappa(\delta + d)(\boldsymbol{M}^{-1})_{00} = \kappa\delta(\boldsymbol{M}^{-1})_{00} = \kappa/c_0$. This is to be expected since isolated vertices each have a separate Gaussian prior with variance $c_0/\kappa$.

Equations (24) and (26) still require the normalisation constant, $\kappa$. The simplest way to calculate this is to run the population dynamics once for $\kappa = 1$ and $\nu = 0$, that is, an unnormalised kernel and no training data. The result for $\varepsilon$ then just gives the average (over vertices) prior variance. With $\kappa$ set to this value, one can then run the population dynamics for any $\nu$ to obtain the Bayes error prediction for GP regression with a globally normalised kernel.

Comparisons between the cavity prediction for the learning curves, numerically exact simulated learning curves and the results of the eigenvalue approximation are shown in Figure 3 (left, centre and right), for regular, Erdős-Rényi and generalised random graphs with power law degree distributions respectively. As can be seen the cavity predictions greatly outperform the eigenvalue approximation and are accurate along the whole length of the curve. This confirms our expectation that the cavity approach will become exact on large graphs, although it is remarkable that the agreement is quantitatively so good already for graphs with only five hundred vertices.

### 4.2 Predicting Prior Variances

As a by-product of the cavity analysis for globally normalised kernels we note that in the cavity form of the Bayes error in Equation (26), the fraction $(\gamma/\sigma^2 + d\kappa(\boldsymbol{M}_d^{-1})_{00})^{-1}$ is the local Bayes error, that is, the local posterior variance. By keeping track of individual samples for this quantity from the population dynamics approach, we can thus predict the distribution of local posterior variances. If we set $\nu = 0$, then this becomes the distribution of prior variances. The cavity approach therefore gives us, without additional effort, a prediction for this distribution.

We can now go back to Section 3.2 and compare the cavity predictions to numerically simulated distributions of prior variances. The cavity predictions for these distributions are shown by the black lines in Figure 2. The cavity approach provides, in particular, detailed information about the tail of the distributions as shown in the insets. There is good agreement between the predictions and the numerical simulations, both regarding the general shape of the variance distributions and the fine structure with a number of non-trivial peaks and troughs. The residual small shifts between the predictions and the numerical results for a single instance of a large graph are most likely due to finite size effects: in a finite graph, the assumption of a tree-like local structure is not exact because there can be rare short cycles; also, the long cycles that the cavity method ignores because their length diverges logarithmically with $V$ will have an effect when $V$ is finite.

### 4.3 Local Normalisation

We now extend the cavity analysis for the learning curves to the case of locally normalised random walk kernels, which, as argued above, provide more plausible probabilistic models. In this case the diagonal entries of the normalisation matrix $\mathcal{K}$ are defined as

$$\kappa_i = \int \mathrm{d}\boldsymbol{f}\, f_i^2 P(\boldsymbol{f}),$$

where $P(\boldsymbol{f})$ is the GP prior with the unnormalised kernel $\hat{\boldsymbol{C}}$. This makes clear why the locally normalised kernel case is more challenging technically: we cannot calculate the normalisation constants once and for all for a given random graph ensemble and set of kernel parameters $p$ and $a$ as we did for $\kappa$ in the globally normalised scenario. Instead we have to account for the dependence of the $\kappa_i$ on the specific graph instance.

On a single graph instance, this stumbling block can be overcome as follows. One iterates the cavity updates (22) for the unnormalised kernel and without the training data (i.e., setting $\kappa = 1$ and $\gamma_i = 0$). The local Bayes error at vertex $i$, given by the $i$-th term in the sum from (20), then gives us $\kappa_i$. Because $\gamma_i = 0$, one has to use the Woodbury trick to get well-behaved expressions in the limit where the auxiliary parameter $\lambda \to 0$, as explained after (25).

Once the $\kappa_i$ have been determined in this way, one can use them for predicting the Bayes error for the scenario we really want to study, that is, using a locally normalised kernel and incorporating the training data. The relevant partition function is the analogue of (18) for local normalisation. Dropping the prefactors, the resulting $Z$ can be written as

$$Z \propto \int \prod_{q=0}^{p} \mathrm{d}\boldsymbol{h}^q \prod_{q=1}^{p} \mathrm{d}\hat{\boldsymbol{h}}^q \prod_i \exp\left( -\frac{1}{2}\sum_{q=0}^{p} c_q d_i h_i^0 h_i^q - \frac{1}{2}\frac{d_i \kappa_i (h_i^0)^2}{\gamma_i/\sigma^2 + \lambda} + \mathrm{i}\sum_{q=1}^{p} d_i \hat{h}_i^q h_i^q \right) \prod_{(i,j)} \exp\left( -\mathrm{i}\sum_{q=1}^{p} (\hat{h}_i^q h_j^{q-1} + \hat{h}_j^q h_i^{q-1}) \right),$$

where we have rescaled $h_i^q$ to $d_i^{1/2}\kappa_i^{1/2} h_i^q$ and $\hat{h}_i^q$ to $d_i^{1/2}\kappa_i^{-1/2}\hat{h}_i^q$. Given that the $\kappa_i$ have already been determined, this is a graphical model for which marginals can be calculated by iterating to a fixed point the equations for the cavity marginals:

$$P_{\mathrm{loc},j}^{(i)}(\boldsymbol{h}_j, \hat{\boldsymbol{h}}_j | \boldsymbol{x}) \propto \exp\left( -\frac{1}{2}\sum_{q=0}^{p} c_q d_j h_j^0 h_j^q - \frac{1}{2}\frac{d_j \kappa_j (h_j^0)^2}{\gamma_j/\sigma^2 + \lambda} + \mathrm{i}\sum_{q=1}^{p} d_j \hat{h}_j^q h_j^q \right) \int \prod_{k \in \mathcal{N}(j)\setminus i} \mathrm{d}\boldsymbol{h}_k \mathrm{d}\hat{\boldsymbol{h}}_k \exp\left( -\mathrm{i}\sum_{q=1}^{p} (\hat{h}_j^q h_k^{q-1} + \hat{h}_k^q h_j^{q-1}) \right) P_{\mathrm{loc},k}^{(j)}(\boldsymbol{h}_k, \hat{\boldsymbol{h}}_k | \boldsymbol{x}). \quad (27)$$

As in Section 4.1 these update equations are solved by cavity marginals of complex Gaussian form, and so we can simplify them to updates for the covariance matrices:

$$\boldsymbol{V}_{\mathrm{loc},j}^{(i)} = \left( \boldsymbol{O}_{\mathrm{loc},j} - \sum_{k \in \mathcal{N}(j)\setminus i} \boldsymbol{X} \boldsymbol{V}_{k,\mathrm{loc}}^{(j)} \boldsymbol{X} \right)^{-1}. \quad (28)$$

Here $\boldsymbol{X}$ is defined as in Equation (23) and $\boldsymbol{O}_{\mathrm{loc},j}$ is the obvious analogue of $\boldsymbol{O}_j$ also defined in Equation (23); specifically, $\kappa$ is replaced by $\kappa_j$. Once the update equations have converged, one can calculate the Bayes error from a similarly adapted version of (20).

The above procedure for a single fixed graph now has to be extended to the case of an ensemble of large random graphs characterised by some degree distribution $p(d)$. The outcome of the first round of cavity updates, for the unnormalised kernel without training data, is then represented by a distribution of cavity covariances $\boldsymbol{V}$, while the second one gives a distribution of cavity covariances $\boldsymbol{V}_{\mathrm{loc}}$ for the locally normalised kernel, with training data included. Importantly, these message distributions are coupled to each other via the graph structure, so we need to look at the joint distribution $W(\boldsymbol{V}_{\mathrm{loc}}, \boldsymbol{V})$.

Detailed analysis using the replica method (Urry and Sollich, 2012) shows that the correct fixed point equation updates the $\boldsymbol{V}$-messages as in the globally normalised case with $\gamma = 0$. The second set of local covariances, $\boldsymbol{V}_{\mathrm{loc}}$, are then updated according to (28), with a normaliser calculated using the marginals from the $d-1$ $\boldsymbol{V}$-covariances and an additional 'counterflow' covariance generated from $W(\boldsymbol{V}) = \int d\boldsymbol{V}_{\mathrm{loc}} W(\boldsymbol{V}_{\mathrm{loc}}, \boldsymbol{V})$, subject to the constraint that the local marginals of the neighbours are consistent. We find in practice that the consistency constraint can be dropped and the fixed point

equation for the distribution of the two sets of messages can be approximated by

$$W(\boldsymbol{V}_{\rm loc},\boldsymbol{V}) = \left\langle \sum_d \frac{p(d)d}{\bar{d}} \int \prod_{k=1}^{d-1} \mathrm{d}\boldsymbol{V}_k \mathrm{d}\boldsymbol{V}_{{\rm loc},k} \mathrm{d}\boldsymbol{V}_d \prod_{k=1}^{d-1} W(\boldsymbol{V}_{{\rm loc},k},\boldsymbol{V}_k) W(\boldsymbol{V}_d) \right.$$
$$\left. \delta\left(\boldsymbol{V}_{\rm loc} - \left(\boldsymbol{O}_{{\rm loc},j} - \sum_{k=1}^{d-1} \boldsymbol{X}\boldsymbol{V}_{k,{\rm loc}}^{(j)}\boldsymbol{X}\right)^{-1}\right) \delta\left(\boldsymbol{V} - \left(\boldsymbol{O} - \sum_{k=1}^{d-1}\boldsymbol{X}\boldsymbol{V}_k\boldsymbol{X}\right)^{-1}\right) \right\rangle_\gamma . \quad (29)$$

One sees that if one marginalises over $\boldsymbol{V}_{\rm loc}$, then one obtains exactly the same condition on $W(\boldsymbol{V})$ as before in the globally normalised kernel case (but with $\kappa = 1$ and $\nu = 0$), see (24). This reflects the fact that the cavity updates for the first set of messages on a single graph do not rely on any information about the second set. The first delta function in (29) corresponds to the fixed point condition for this second set of cavity updates. This condition depends, via the value of the local $\kappa$, on the $\boldsymbol{V}$-cavity covariances:

$$\kappa = \frac{1}{d(\boldsymbol{M}_d^{-1})_{00}}. \quad (30)$$

It may seem unusual that $d$ copies of $\boldsymbol{V}$ enter here; $\boldsymbol{V}_d$ represents the cavity covariance from the first set that is *received* from the vertex to which the new message $\boldsymbol{V}_{\rm loc}$ is being *sent*. While this counterflow appears to run against the basic construction of the cavity or belief propagation method, it makes sense here because the first set of cavity messages (or equivalently the distribution $W(\boldsymbol{V})$) reaches a fixed point that is independent of the second set, so the counterflow of information is only apparent. The reason why knowledge about $\boldsymbol{V}_d$ is needed in the update is that $\kappa$ is the variance of a full marginal rather than a cavity marginal.

Similarly to the case of global normalisation, (29) can be solved by looking for a fixed point of $W(\boldsymbol{V}_{\rm loc},\boldsymbol{V})$ using population dynamics. Updates are made by first updating $\boldsymbol{V}$ using Equation (21) and then updating $\boldsymbol{V}_{\rm loc}$ using (27) with $\kappa \equiv \kappa_i$ replaced by (30).

Once a fixed point has been calculated for the covariance distribution we apply the Woodbury formula to (20) in a similar manner to Section 4.1 to give the prediction for the learning curve for GP regression with a locally normalised kernel. The result for the Bayes error becomes

$$\varepsilon = \left\langle \sum_d p(d) \int \prod_{k=1}^{d} \mathrm{d}\boldsymbol{V}_{{\rm loc},k} \mathrm{d}\boldsymbol{V}_k \prod_{k=1}^{d} W(\boldsymbol{V}_{{\rm loc},k},\boldsymbol{V}_k) \, \frac{1}{\gamma/\sigma^2 + (\boldsymbol{M}_{d,{\rm loc}}^{-1})_{00}/(\boldsymbol{M}_d^{-1})_{00}} \right\rangle_\gamma .$$

Learning curve predictions for GPs with locally normalised kernels as they result from the cavity approach described above are shown in Figure 5. The figure shows numerically simulated learning curves and the cavity prediction, both for Erdős-Rényi random graphs (left) and power law generalised random graphs (centre) of size $V = 500$. As for the globally normalised case one sees that the cavity predictions are quantitatively very accurate even with the simplified update Equation (29). They capture all aspects of learning curve both qualitatively and quantitatively, including, for example, the shoulder in the curves from disconnected single vertices, a feature discussed in more detail below.

The fact that the cavity predictions of the learning curve for a locally normalised kernel are indistinguishable from the numerically simulated learning curves in Figure 5 leads us to believe that the simplification made by dropping the consistency requirement in (29) is in fact exact. This is further substantiated by looking not just at the average of the posterior variance over vertices, which

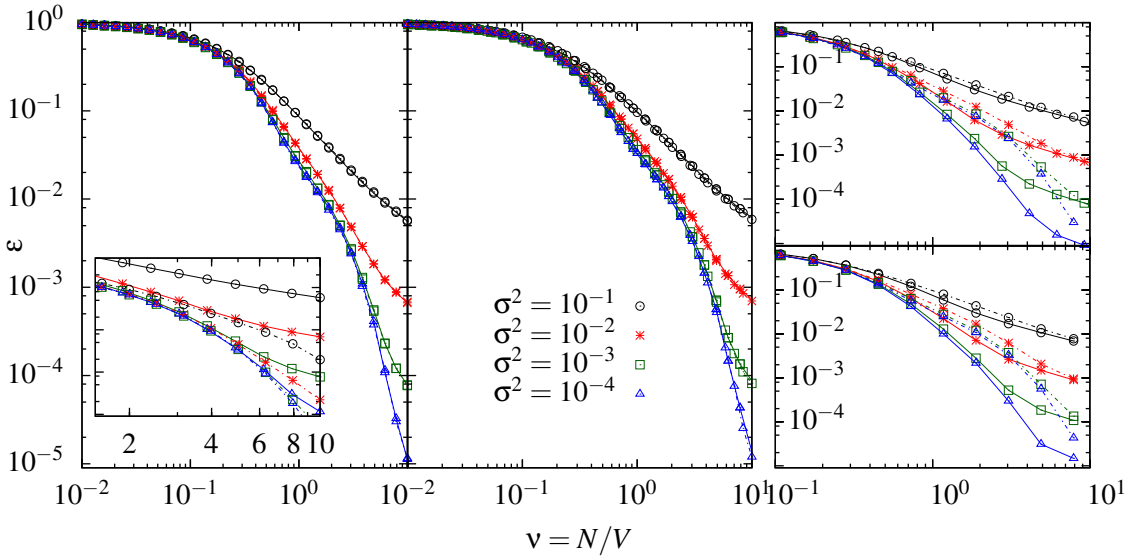



Figure 5: (Left) Learning curves for GP regression with locally normalised kernels with $p = 10$, $a = 2$ on Erdős-Rényi random graphs with mean degree 3, for a range of noise levels $\sigma^2$. Solid lines: numerically simulated learning curves for graphs of size $V = 500$, dashed lines: cavity predictions (see Section 4.1); note these are mostly visually indistinguishable from the simulation results. (Left inset) Dotted lines show single vertex contributions to the learning curve (solid line). (Centre) As (left) for power law generalised random graphs with exponent 2.5 and cut off 2. (Right top) Comparison between learning curves for locally (dashed line) and globally (solid line) normalised kernels for Erdős-Rényi random graphs. (Right bottom) As (right top) for power law random graphs.

is the Bayes error, but its distribution across vertices. As shown in Figure 6, the cavity predictions for this distribution are in very good agreement with the results of numerical simulations. This holds not only for the two values of $\nu$ shown, but along the entire learning curve.

## 5. A Qualitative Comparison of Learning with Locally and Globally Normalised Kernels

The cavity approach we have developed gives very accurate predictions for learning curves for GP regression on graphs using random walk kernels. This is true for both global and local normalisations of the kernel. We argued in Section 3.2 that the local normalisation is much more plausible as a probabilistic model, because it avoids variability in the local prior variances that is non-trivially related to the local graph structure and so difficult to justify from prior knowledge. We now compare what the qualitative effects of the two different normalisations are on GP learning.

It is not a simple matter to say which kernel is 'better', the locally or globally normalised one. Since we have dealt with the matched case, where for each kernel the target functions are sampled from a GP prior with that kernel as covariance function, it would not make sense to say the better kernel is the one that gives the lower Bayes error for given number of examples, as the Bayes error

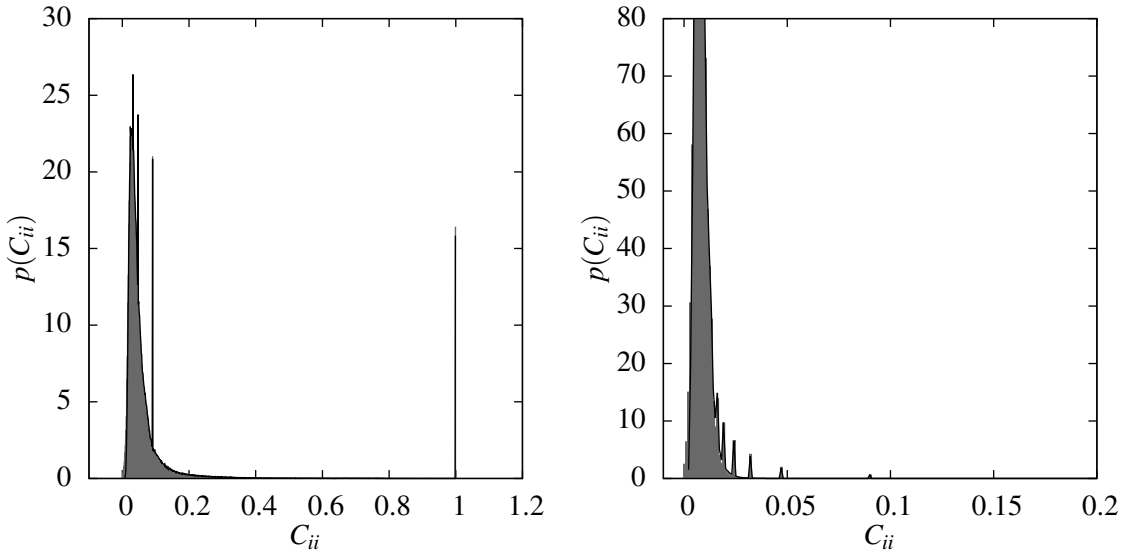



Figure 6: (Left) Grey: histogram of posterior variances at $\nu = 1.172$ for the locally normalised random walk kernel with $a = 2$, $p = 10$, averaged over ten samples each of teacher functions, data and Erdős-Rényi graphs with mean degree $\lambda = 3$ and $V = 1000$ vertices. Black: cavity prediction for this distribution in the large graph limit. (Right) As (left) but for $\nu = 6.210$.

reflects both the complexity of the target function and the success in learning it. A more definite answer could be obtained only empirically, by running GP regression with local and global kernel normalisation on the same data sets and comparing the prediction errors and also the marginal data likelihood. The same approach could also be tried with synthetic data sets generated from GP priors that are mismatched to both priors we have considered, defined by the globally and locally normalised kernel, though justifying what is a reasonable choice for the prior of the target function would not be easy.

While a detailed study along the lines above is outside the scope of this paper, we can nevertheless at least qualitatively study the effect of the kernel normalisation, to understand to what extent the corresponding priors define significantly different probabilistic models. Figure 5 (right top and bottom) overlays the learning curves for global and local kernel normalisations, for an Erdős-Rényi and a power law generalised random graph respectively. There are qualitative differences in the shapes of the learning curves, with the ones for the locally normalised kernel exhibiting a shoulder around $\nu = 2$. This shoulder is due to the proper normalisation of isolated vertices to unit prior variance; by contrast, as shown earlier in Figure 2 (left), global normalisation gives too small a prior variance to such vertices. The inset in Figure 5 (left) shows the expected learning curve contributions from all locally normalised isolated vertices (single vertex subgraphs) as dotted lines. After the GP learns the rest of the graph to a sufficient accuracy, the single vertex error dominates the learning curve until these vertices have typically seen at least one example. Once this point has been passed, the dominant error comes once more from the giant connected component of the graph, and the GP

learns in a similar manner to the globally normalised case. A similar effect, although not plotted, is seen for the generalised random graph case.

We can extend the scope of this qualitative comparison by examining how a student GP with a kernel with one normalisation performs when learning from a teacher with a kernel with the other normalisation. This is a case of *model mismatch*; our theory so far does not extend to this scenario, but we can obtain learning curves by numerical simulation. Figure 7 (left) shows the case of GP students with a globally normalised kernel learning from a teacher with a locally normalised kernel on an Erdős-Rényi graph. The learning curves for the mismatched scenario are very different from those for the matched case (Figures 3 and 5), showing an increase in error as $\nu$ approaches unity. The resulting maximum in the learning curve again emphasises that the two choices of normalisation produce distinctly different probabilistic models. Similar behaviour can be observed for the case of power law generalised random graphs, shown in Figure 7 (right) and for the case of GP students with a locally normalised kernel learning from a teacher with a globally normalised kernel, shown in Figure 8. In all cases close inspection (see Appendix B) shows that the error maximum is caused by 'dangling edges' of the graph, that is, chains of vertices (with degree two) extending away from the giant graph component and terminating in a vertex of degree one.

As a final qualitative comparison between globally and locally normalised kernels, Figure 9 shows the variance of local posterior variances. This measures how much the local Bayes error typically varies from vertex-to-vertex, as a function of the data set size. Plausibly one would expect that this error variance is low initially when prediction on all vertices is equally uncertain. For large data sets the same should be true because errors on all vertices are then low. In an intermediate regime the error variance should become larger because examples will have been received on or near some vertices but not others. As Figure 9 shows, for kernels with local normalisation we find exactly this scenario, both for Erdős-Rényi and power law random graphs. The error variance is low for small $\nu = N/V$, increasing to a peak at $\nu \approx 0.2$ and finally decreasing again.

These results can now be contrasted with those for globally normalised kernels, also displayed in Figure 9. Here the error variance is largest at $\nu = 0$ and decays from there. This means that the initial variance in the local prior variances is so large that any effects from the uneven distribution of example locations in any given data set remain sub-dominant throughout. We regard this as another indication of the probabilistically implausible character of the large spread of prior variances caused by global normalisation.

## 6. Conclusions and Further Work

In this paper we studied random walk kernels and their application to GP regression. We began, in Section 2, by studying the random walk kernel, with a focus on applying this to $d$-regular trees and graphs. We showed that the kernel exhibits a rather subtle approach to the fully correlated limit; this limit is reached only beyond a graph-size dependent threshold for the kernel range $p/a$, where cycles become important. If $p/a$ is large but below this threshold, the kernel reaches a non-trivial limiting shape.

In Section 3 we moved on to the application of random walk kernels to GP regression. We showed, in Section 3.2, that the more typical approach to normalisation, that is, scaling the kernel globally to a desired average prior variance, results in a large spread of local prior variances that is related in a complicated manner to the graph structure; this is undesirable in a prior. We suggested

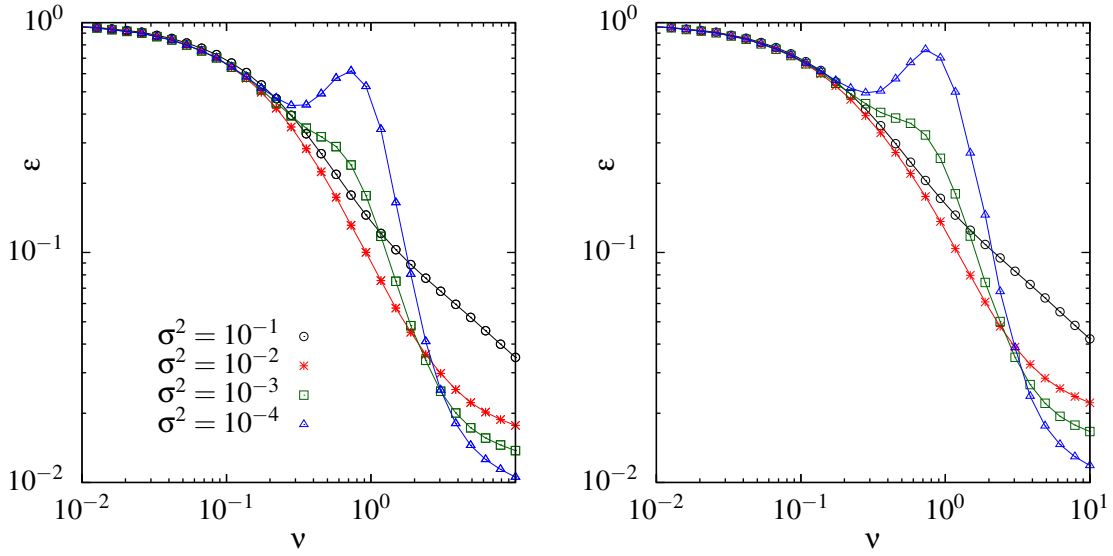



Figure 7: (Left) Numerically simulated learning curves for a GP with a globally normalised kernel with $p = 10$ and $a = 2$, on Erdős-Rényi random graphs with mean degree 3 for a range of noise levels. The teacher GP has a locally normalised kernel with the same parameters. (Right) As (left) but for power law generalised random graphs with exponent 2.5 and cutoff 2.

as a simple remedy to perform local normalisation, where the raw kernel is normalised by its local prior variance so that the prior variance becomes the same at every vertex.

In order to get a deeper understanding of the performance of GPs with random walk kernels we then studied the learning curves, that is, the mean Bayes error as a function of the number of examples. We began in section 3.4 by applying a previous approximation due to Sollich (1999a) and Malzahn and Opper (2005) to the case of discrete inputs that are vertices on a graph. We demonstrated numerically that this approximation is accurate only for small and large number of training examples per vertex, $\nu = N/V$, while it fails in the crossover between these two regimes. The outline derivation of this approximation suggested how one might improve it: one has to exploit fully the structure of random graphs, using the cavity method, thus avoiding approximating the average over data sets. In Section 4 we implemented this programme, beginning in Section 4.1 with the case of global normalisation. We showed that by Fourier transforming the prior and introducing $2p$ additional variables at each vertex one can rewrite the partition function in terms of a complex-valued Gaussian graphical model, where the marginals that are required to calculate the Bayes error can be found using the cavity method, or equivalently belief propagation. In Section 4.3 we tackled the more difficult scenario of a locally normalised kernel. This required two sets of cavity equations. The first serves to calculate the local normalisation factors. The second one then combines these with the information about the data set to find the local marginal distributions. One might be tempted to consider applying our methods to a lattice so that one could make an estimate of the learning curves for the continuous limit, that is, regression with a squared exponential kernel

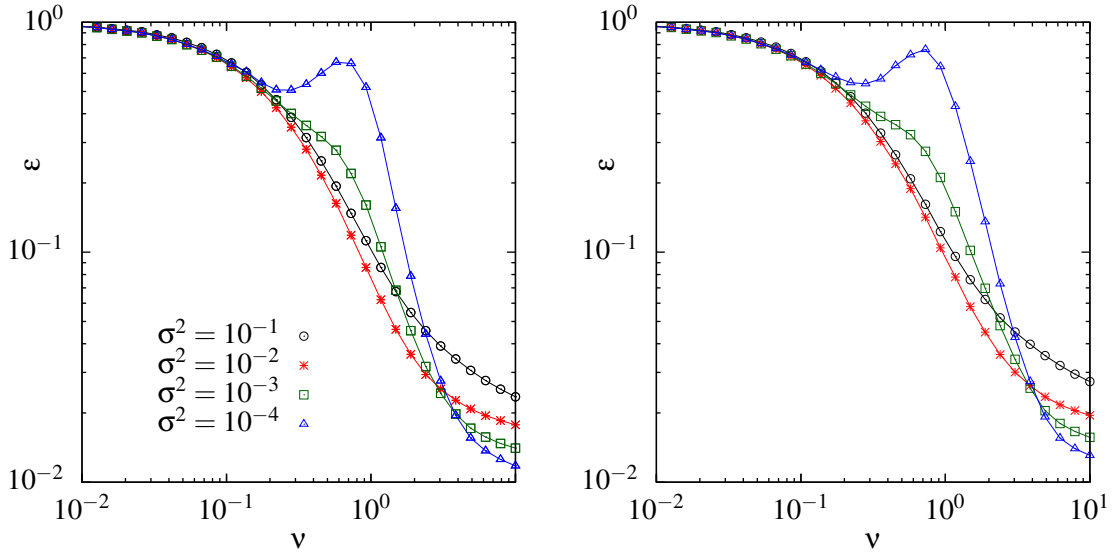



Figure 8: (Left) Numerically simulated learning curves for a GP with a locally normalised kernel with $p = 10$ and $a = 2$, on Erdős-Rényi random graphs with mean degree 3 for a range of noise levels. The teacher GP has a globally normalised kernel with the same parameters. (Right) As (left) but for power law generalised random graphs with exponent 2.5 and cutoff 2.

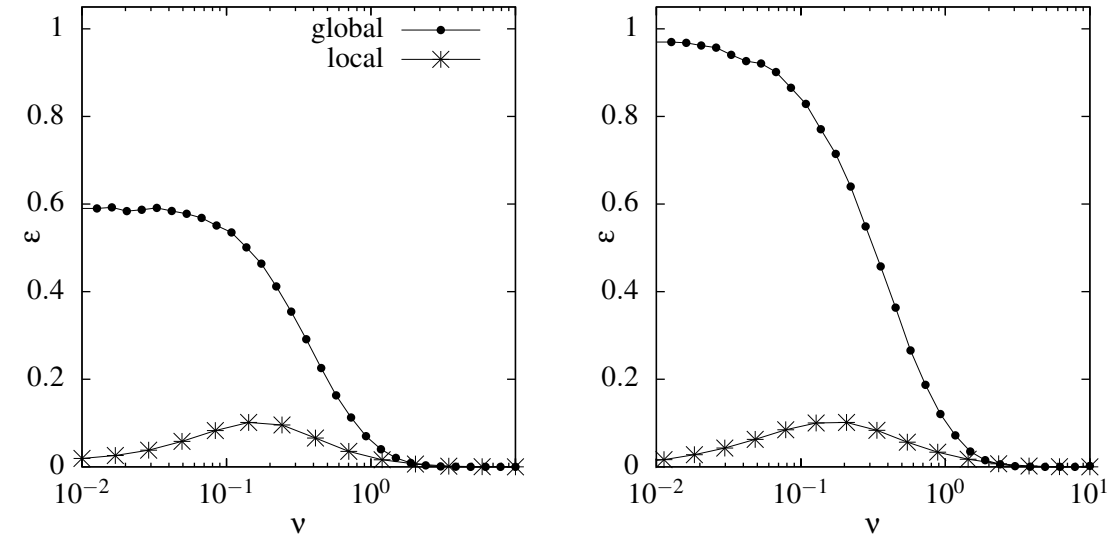



Figure 9: (Left) Error variance for GP regression with locally (stars) and globally (circles) normalised kernels against ν, on Erdős-Rényi random graphs, and for matched learning with $p = 10$, $a = 2$, $\sigma^2 = 0.1$. (Right) As (left) but for power law generalised random graphs with exponent 2.5 and cutoff 2.

for inputs in $\mathbb{R}^2$ or similar. Sadly, however, since the cavity method requires graphs to be treelike this is not possible.

Finally in Section 5 we qualitatively compared GPs with kernels that are normalised globally and locally. We showed that learning curves are indeed qualitatively different. In particular, local normalisation leads to a shoulder in the learning curves owing to the correct normalisation of the single vertex disconnected graph components. We also considered numerically calculated mismatch curves. The mismatch caused a maximum to appear in the learning curves, as a result of the large differences in the teacher and student priors. Lastly we looked at the variance among the local Bayes errors, for GPs with both globally and locally normalised kernels. Plausibly, locally normalised kernels lead to this error variance being maximal for intermediate data set sizes. This reflects the variation in number of examples seen at or near individual vertices. For globally normalised kernels, the error variance inherited from the spread of local prior variances is always dominant, obscuring any signatures from the changing 'coverage' of the graph by examples.

In further work we intend to extend the cavity approximation of the learning curves to the case of mismatch, where teacher and student have different kernel hyperparameters. It would also be interesting to apply the cavity method to the learning curves of GPs with random walk kernels on more general random graphs, like those considered in Rogers et al. (2010) and Kühn and van Mourik (2011). This would enable us to consider graphs exhibiting some community structure. Looking further ahead, preliminary work has shown that it should be possible to extend the cavity learning curve approximation to the problem of graph mismatch, where the student has incomplete information about the graph structure of the teacher.

## Appendix A. Random Walk Kernel on $d$-Regular Trees

We detail here how to derive the limiting kernel from (5) in Section 2.1 and how to calculate the scaling of the learning curve stated in Section 3.4.1.

### A.1 Large-$p$ Limit from Heat Kernel Results

In Chung and Yau (1999) the authors considered heat kernels on graphs. The heat kernel for a graph with normalised Laplacian $\boldsymbol{L}$ (see Section 2 in the main text) is given by

$$\boldsymbol{H} = \exp(-t\boldsymbol{L}), \quad t > 0,$$

This is exactly the diffusion kernel given in (1) with $t = \frac{1}{2}\sigma^2$. Chung and Yau (1999) gave results for heat kernels on $d$-regular trees. Their method involves treating the tree as a covering of a path, which is closely related to the mapping onto a one-dimensional lattice discussed in Section 2.1. The eigenvectors and eigenvalues of $\boldsymbol{L}$ are then found within this mapping. If we call the unnormalised form of the random walk kernels we considered in the main text again $\hat{\boldsymbol{C}} = \left(\boldsymbol{I} - a^{-1}\boldsymbol{L}\right)^p$, then we can directly use the results of Chung and Yau (1999), simply by modifying the function that is applied to each eigenvalue $\lambda$ of $\boldsymbol{L}$ from $\exp(-t\lambda)$ to $(1-\lambda/a)^p$. This gives, for the unnormalised

random walk kernel element between two vertices on the tree at distance $l$,

$$\begin{aligned}\hat{C}_{0,p} &= \frac{2(d-1)d}{\pi}\int_0^\pi \mathrm{d}x\,\frac{[1-(1-2\cos(x)\sqrt{d-1}/d)/a]^p\sin^2(x)}{d^2-4(d-1)\cos^2(x)},\\ \hat{C}_{l\geq 1,p} &= \frac{2}{\pi(d-1)^{l/2-1}}\int_0^\pi \mathrm{d}x\,[1-(1-2\cos(x)\sqrt{d-1}/d)/a]^p\\ &\qquad\times\frac{\sin(x)[(d-1)\sin((l+1)x)-\sin((l-1)x)]}{d^2-4(d-1)\cos^2(x)}.\end{aligned} \tag{31}$$

In the limit $p\to\infty$, the factor in square brackets in both expressions means that only values of $x$ up to $O(p^{-1/2})$ contribute significantly to the integral. One can then expand $\sin(x)\approx x$ linearly everywhere, and similarly use $\cos(x)\approx 1$. Calculating $\hat{C}_{l,p}$ in this way for $p\to\infty$ gives the result (5) for $C_{l,p}=\hat{C}_{l,p}/\hat{C}_{0,p}$.

### A.2 Scaling Form for Large $p$

In Section 3.4.1 we derived the learning curve decay due to the first example, given in (15). As explained there, to understand how this behaves for large $p$ one needs to know how $C_{l,p}$ approaches its limiting form (5). In the outline derivation in the previous subsection we saw that in order to obtain this limiting form one expands not just $\sin(x)$, but also $\sin((l-1)x)$ and $\sin((l+1)x)$ linearly because $x=O(p^{-1/2})$ is small. For the two sine factors this will break down once $lp^{-1/2}$ is of order one, that is, for large enough $l$. This suggests looking at $C_{l,p}$ as function of $l'=lp^{-1/2}$. To focus on the relevant part of the integral one can similarly transform the integration variable to $x'=xp^{1/2}$. For $p\to\infty$ at constant $l'$, the integral for $\hat{C}_{l,p}$ in (31) then becomes proportional to

$$(d-1)^{-l/2}\int_0^\infty \mathrm{d}x'\,\mathrm{e}^{-x'^2\sqrt{d-1}/(da)}x'(d-2)\sin(l'x').$$

Integration by parts now gives $C_{l,p}\propto\hat{C}_{l,p}\propto(d-1)^{-l/2}l'\exp(-\frac{(l')^2d}{2\sqrt{d-1}})$. Comparing with the large-$l$ limit of (5), $C_{l,p}\propto(d-1)^{-l/2}l'$, shows that the effect of finite $p$ is contained in the exponential (Gaussian) cutoff factor which becomes significant for $l'$ of order unity, that is, $l=O(p^{1/2})$, as stated in the main text.

Visually, the cutoff effects for large $l$ and $p$ can be seen more clearly by plotting not $C_{l,p}$ directly but rather $\hat{R}_{l,p}$, the unormalised version of $R_{l,p}$ as defined in Section 3.4.1. As $v_l=d(d-1)^{l-1}$ for $l\geq 1$, the additional $\sqrt{v_l}$ factor just removes the decay with $(d-1)^{-l/2}$ from $C_{l,p}$. From the above discussion, we then expect to find for large $l$ that $\hat{R}_{l,p}\propto l'\exp(-\frac{(l')^2d}{2\sqrt{d-1}})$. A plot of numerical results for a suitably normalised version of $\hat{R}_{l,p}$ (see below), plotted against $lp^{-1/2}$ for increasing $p$, is shown in Figure 10. There is a clear trend towards the predicted asymptotic behaviour for large $p$ (dashed line).

It may be somewhat surprising that the cutoff lengthscale that appears in the analysis above is $l=O(p^{1/2})$, which for large $p$ is much smaller than the typical kernel range $p/a$. To understand this intuitively, one can go back to (6) for $R_{l,p}$ and use that this is essentially unnormalised diffusion. Taking $\hat{R}_{l,p}$ as the unnormalised version of $R_{l,p}$ again and letting $\rho_{l,p}=\hat{R}_{l,p}\gamma^{-p}$ with $\gamma=(1-1/a)+2\sqrt{d-1}/(ad)$ to re-establish normalisation, one finds that $\rho_{l,p}$ evolves almost according to a diffusion process in 'time' $p$, except that probability conservation is broken at $l=0$ and $l=1$. In the (leaky) diffusion process $\rho_{l,p}$ the typical lengthscale $l$ should then scale with time as $p^{1/2}$, exactly

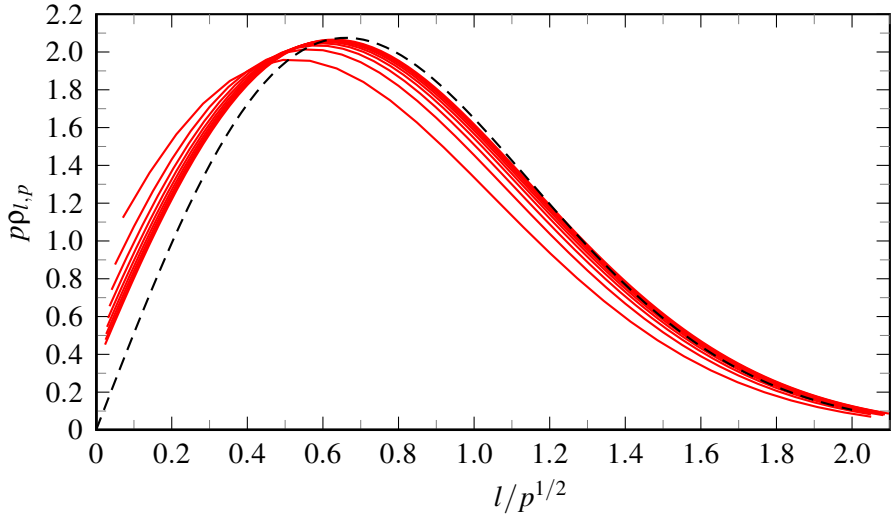



Figure 10: Plot demonstrating the $p \to \infty$ scaling from of $p\rho_{l,p}$, the normalised version of $\hat{R}_{l,p}$. Dashed line shows the analytically found functional from of the scaling.

as we found above. This diffusion interpretation can also be used to reproduce the quantitative scaling form, by adapting the methods of Monthus and Texier (1996) where the authors study a one dimensional walk with a reflective boundary to compute the large-$p$ scaling. The normalised version of $\hat{R}_{l,p}$ shown in Figure 10 is in fact $p\rho_{l,p}$.

## Appendix B. Analysis of the Mismatch Learning Curves

In this appendix we suggest a way of understanding the mismatch maximum seen in the learning curves plotted in Figure 7 (left) and (right) and Figure 8 (left) and (right). We begin by considering the mean of the GP posterior, that is, the prediction function. If we arrange the predictive means at each vertex into a vector $\bar{\boldsymbol{f}}$, then from (8) this can be written as

$$\bar{\boldsymbol{f}} = \boldsymbol{K}_V \boldsymbol{K}^{-1} \boldsymbol{y},$$

where $(K_V)_{j\nu} = C_{j,x_\nu}$ for $\nu = 1,\ldots,N$ and $j = 1,\ldots,V$ and $K_{\mu\nu} = C_{x_\mu,x_\nu} + \sigma^2\delta_{\mu,\nu}$ for $\mu,\nu = 1,\ldots,N$. (Recall that $x_\mu$ is the location of the $\mu$-th training input, which on a graph is just a label in the range $\{1,\ldots,V\}$.) We can rewrite this in the form $\bar{\boldsymbol{f}} = \boldsymbol{M}\boldsymbol{z}$ where

$$\boldsymbol{M} = \boldsymbol{K}_V \boldsymbol{K}^{-1/2},$$
$$\boldsymbol{z} = \boldsymbol{K}^{-1/2} \boldsymbol{y}.$$

One sees that $\boldsymbol{M}$ represents teacher-independent aspects of $\bar{\boldsymbol{f}}$. The vector of 'pseudo-training outputs' $\boldsymbol{z}$ has been defined so that in the *matched* case, it obeys $\langle \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}} \rangle = \boldsymbol{I}$. We think of $\boldsymbol{z}$ as pseudo-training outputs because if its components $z_\mu$ are sampled as independent and identically distributed unit variance Gaussian random variables, then $\bar{\boldsymbol{f}} = \boldsymbol{M}\boldsymbol{z}$ will have the correct distribution of mean prediction vectors. The columns $\boldsymbol{m}_1,\ldots,\boldsymbol{m}_N$ of $\boldsymbol{M} = \begin{pmatrix} \boldsymbol{m}_1 & \cdots & \boldsymbol{m}_N \end{pmatrix}$ represent the 'effective

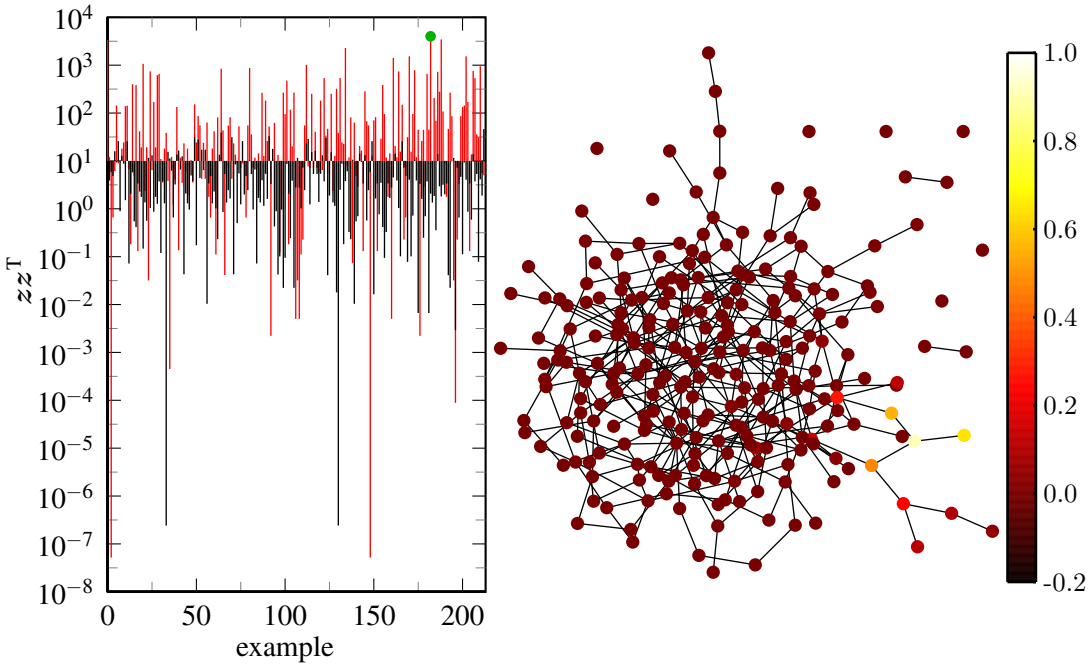



Figure 11: (Left) Black lines show diagonal elements of $zz^{\mathrm{T}}$ for a typical draw from the Erdős-Rényi random graph ensemble with average degree 3 in a matched scenario with 250 vertices, $p = 10$, $a = 2$ and $\sigma^2 = 0.0001$. Red lines show $zz^{\mathrm{T}}$ for the case where student is globally normalised and the teacher is locally normalised. (Right) A plot of the 'prediction vector' for the largest spike in $zz^{\mathrm{T}}$ (indicated by a green circle in the left hand plot) under mismatch.

prediction vectors' conjugate to the $z_\mu$. In the matched case each $\bar{f}_i^2$ will then be, on average, a sum of squares of the corresponding entries in the effective prediction vectors.

For the case of mismatch, for example, because student and teacher use differently normalised kernels, the only change is in the statistics of $z$. Their covariance matrix $\langle zz^{\mathrm{T}} \rangle$ will no longer simply be the identity matrix, and so act as a re-weighting of the prediction vectors. To understand our numerical simulations, we considered a value of $\nu$ around the maximum in the mismatched learning curve, listed the largest (diagonal) entries of $zz^{\mathrm{T}}$ and plotted their corresponding prediction vectors. These prediction vectors were generally localised around dangling ends of the graph, substantiating our claim in the main text that it is from these graph regions that the major contribution to the learning curve maximum arises. Typical plots of $zz^T$ and the prediction vector corresponding to the largest spike in $zz^{\mathrm{T}}$ for an instance of an Erdős-Rényi graph with 250 vertices are shown in Figures 11 and 12 for both a locally normalised teacher and globally normalised student and a globally normalised teacher and locally normalised student respectively. Both these plots clearly show the largest spike corresponding to a prediction vector localised on a dangling edge in the graph. Similar plots of the other large spikes give prediction vectors localised on other dangling edges.

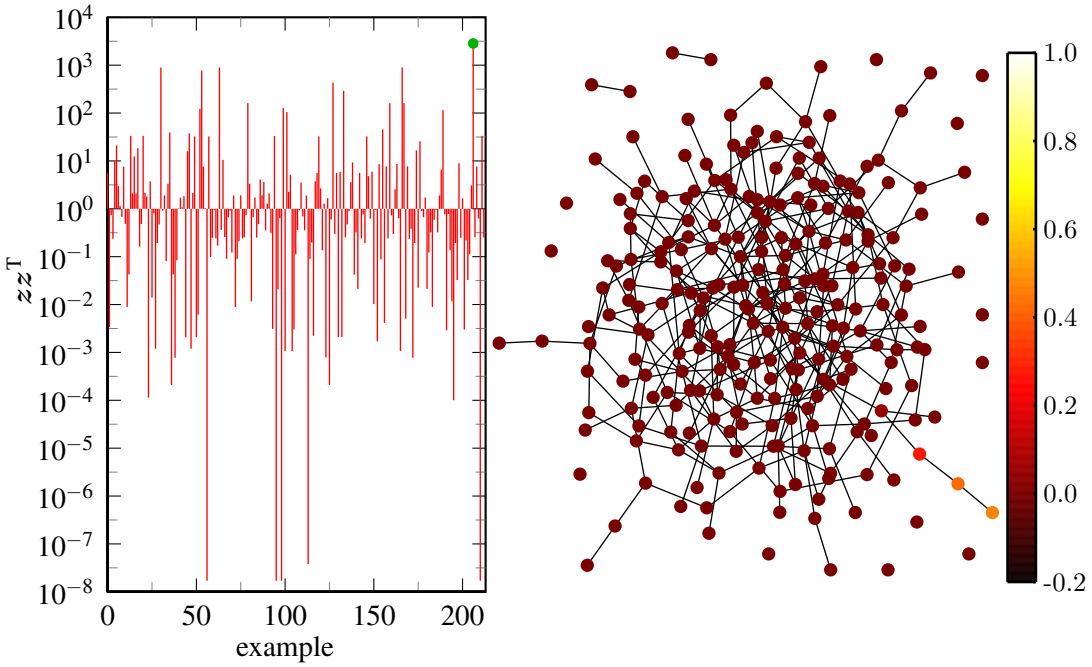



Figure 12: Analogue of Figure 11 for a locally normalised student and globally normalised teacher

## References

S. Amari, N. Fujita, and S. Shinomoto. Four types of learning curves. *Neural Comput.*, 4(4): 605–618, 1992.

C. Bishop. *Pattern Recognition And Machine Learning*. Springer, New York, NY, USA., 2nd edition, 2007.

T. Britton, M. Deijfen, and A. Martin-Loeff. Generating simple random graphs with prescribed degree distribution. *J. Stat. Phys.*, 124(6):1377–1397, 2006.

F. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series In Mathematics*. American Mathematical Society, Dec 1996.

F. K. Chung and S. T. Yau. Coverings, heat kernels and spanning trees. *Electronic Journal Of Combinatorics*, 6:R12, 1999.

N. Cristianini and J. Shawe-Taylor. *An Introduction To Support Vector Machines And Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

P. Erdős and A. Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

J. Freeman and D. Saad. Dynamics of on-line learning in radial basis networks. *Phys. Rev. E*, 56 (1):907–918, Jul 1997.

M. G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal Of Machine Learning Research*, 2(2):299–312, 2002.

W.W. Hager. Updating the inverse of a matrix. *SIAM Rev.*, 31(2):221–239, 1989.

M. S. Handcock and M. L. Stein. A Bayesian analysis of Kriging. *Technometrics*, 35(4):403–410, 1993.

D. Haussler, M. Kearns, H. S. Seung, and N. Tishby. Rigorous learning curve bounds from statistical mechanics. *Mach. Learn.*, 25(2–3):195–236, 1996.

J. P. C. Kleijnen. Kriging metamodeling in simulation: A review. *European Journal Of Operational Research*, 192(3):707 – 716, 2009.

R. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In C. Sammut and Hoffmann A. G., editors, *Proceedings Of The 19th International Conference On Machine Learning (ICML)*, pages 315–322, San Francisco, CA, USA., 2002. Morgan Kaufmann.

R. Kühn and J. van Mourik. Spectra of modular and small-world matrices. *J. Phys. A Math. Gen.*, 44(16):165205, 2011.

D. Malzahn and M. Opper. Learning curves and bootstrap estimates for inference with Gaussian processes: A statistical mechanics study. *Complexity*, 8(4):57–63, 2003.

D. Malzahn and M. Opper. A statistical physics approach for the analysis of machine learning algorithms on real data. *J. Stat. Mech. Theor. Exp.*, 2005(11):P11001, 2005.

B. D. McKay. The expected eigenvalue distribution of a large regular graph. *Linear Algebra And Its Applications*, 40:203–216, 1981.

J. R. Meinhold and N. D. Singpurwalla. Understanding the Kalman filter. *The American Statistician*, 37(2):123–127, 1983.

M. Mézard and G. Parisi. The Bethe lattice spin glass revisited. *Eur. Phys. J. B*, 20(2):217–233, 2001.

M. Mézard and G. Parisi. The cavity method at zero temperature. *J. Stat. Phys.*, 111(1–2):1–34, 2003.

M. Mézard, G. Parisi, and M. Virasoro. *Spin Glass Theory And Beyond*, volume 9 of *World Scientific Lecture Notes In Physics*. World Scientific, Singapore, 1987.

C. Monthus and C. Texier. Random walk on the Bethe lattice and hyperbolic Brownian motion. *J. Phys. A Math. Gen.*, 29:2399–2409, 1996.

K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions On Neural Networks*, 12(2):181–201, 2001.

R. M. Neal. *Bayesian Learning For Neural Networks*, volume 118 of *Lecture Notes In Statistics*. Springer-Verlag, Berlin, Germany, 1996.

M. Opper and D. Haussler. Bounds for predictive errors in the statistical mechanics of supervised learning. *Phys. Rev. Lett.*, 75(20):3772–3775, 1995.

M. Opper and D. Malzahn. A variational approach to learning curves. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances In Neural Information Processing Systems*, volume 14, pages 463–469, Cambridge, MA, USA., 2002. The MIT Press.

M. Opper and F. Vivarelli. General bounds on Bayes errors for regression with Gaussian processes. In M. S. Kearns, S. A. Solla, and D A Cohn, editors, *Advances In Neural Information Processing Systems*, volume 11, pages 302–308, Cambridge, MA, USA., 1999. The MIT Press.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes For Machine Learning*. MIT Press, Nov 2005.

T. Rogers, C. Vicente, K. Takeda, and I. Castillo. Spectral density of random graphs with topological constraints. *J. Phys. A Math. Gen.*, 43(19):195002, 2010.

H. S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Phys. Rev. A*, 45(8):6056–6091, 1992.

A. Smola and R. Kondor. Kernels and regularization on graphs. In B Schölkopf and M. K. Warmuth, editors, *Lect. Notes Artif. Int.*, volume 2777, pages 144–158, 2003.

P. Sollich. Learning curves for Gaussian processes. In M. S. Kearns, S. A. Solla, and D A Cohn, editors, *Advances In Neural Information Processing Systems*, volume 11, pages 344–350, Cambridge, MA, USA., 1999a. The MIT Press.

P. Sollich. Approximate learning curves for Gaussian processes. In *Ninth International Conference On Artificial Neural Networks (ICANN99)*, pages 437–442, Edison, NJ, USA., 1999b. Institute of Electrical Engineers Inspec inc.

P. Sollich. Gaussian process regression with mismatched models. In S Becker T G Dietterich and Z Ghahramani, editors, *Advances In Neural Information Processing Systems*, volume 14, pages 519–526, Cambridge, MA, USA., 2002. The MIT Press.

P. Sollich and A. Halees. Learning curves for Gaussian process regression: Approximations and bounds. *Neural Comput.*, 14(6):1393–1428, 2002.

P. Sollich and C. K. I. Williams. Using the equivalent kernel to understand Gaussian process regression. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances In Neural Information Processing Systems*, volume 17, pages 1313–1320, Cambridge, MA, USA., 2005. The MIT Press.

P. Sollich, M. J. Urry, and C. Coti. Kernels and learning curves for Gaussian process regression on random graphs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances In Neural Information Processing Systems*, volume 22, pages 1723–1731, Red Hook, NY, USA., 2009. Curran Associates Inc.

M. J. Urry and P. Sollich. Exact learning curves for Gaussian process regression on large random graphs. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances In Neural Information Processing Systems*, volume 23, pages 2316–2324, Red Hook, NY, USA., 2010. Curran Associates Inc.

M. J. Urry and P. Sollich. Replica theory for learning curves for gaussian processes on random graphs. *J. Phys. A Math. Gen.*, 45(42):425005, 2012.

T. L. H. Watkin, A. Rau, and M. Biehl. The statistical mechanics of learning a rule. *Rev. Mod. Phys.*, 65(2):499–556, 1993.

C. Williams and F. Vivarelli. Upper and lower bounds on the learning curve for Gaussian processes. *Mach. Learn.*, 40:77–102, 2000.

Journal of Machine Learning Research 14 (2013) 1837-1864 Submitted 11/12; Revised 5/13; Published 7/13

# Distributions of Angles in Random Packing on Spheres

**Tony Cai** TCAI@WHARTON.UPENN.EDU
*Statistics Department*
*The Wharton School*
*University of Pennsylvania*
*Philadelphia, PA 19104, USA*

**Jianqing Fan** JQFAN@PRINCETON.EDU
*Department of Operation Research and Financial Engineering*
*Princeton University*
*Princeton, NJ 08540, USA*

**Tiefeng Jiang** TJIANG@STAT.UMN.EDU
*School of Statistics*
*University of Minnesota*
*Minneapolis, MN 55455, USA*

**Editor:** Tong Zhang

## Abstract

This paper studies the asymptotic behaviors of the pairwise angles among $n$ randomly and uniformly distributed unit vectors in $\mathbb{R}^p$ as the number of points $n \to \infty$, while the dimension $p$ is either fixed or growing with $n$. For both settings, we derive the limiting empirical distribution of the random angles and the limiting distributions of the extreme angles. The results reveal interesting differences in the two settings and provide a precise characterization of the folklore that "all high-dimensional random vectors are almost always nearly orthogonal to each other". Applications to statistics and machine learning and connections with some open problems in physics and mathematics are also discussed.

**Keywords:** random angle, uniform distribution on sphere, empirical law, maximum of random variables, minimum of random variables, extreme-value distribution, packing on sphere

## 1. Introduction

The distribution of the Euclidean and geodesic distances between two random points on a unit sphere or other geometric objects has a wide range of applications including transportation networks, pattern recognition, molecular biology, geometric probability, and many branches of physics. The distribution has been well studied in different settings. For example, Hammersley (1950), Lord (1954), Alagar (1976) and García-Pelayo (2005) studied the distribution of the Euclidean distance between two random points on the unit sphere $\mathbb{S}^{p-1}$. Williams (2001) showed that, when the underlying geometric object is a sphere or an ellipsoid, the distribution has a strong connection to the neutron transport theory. Based on applications in neutron star models and tests for random number generators in $p$-dimensions, Tu and Fischbach (2002) generalized the results from unit spheres to more complex geometric objects including the ellipsoids and discussed many applications. In general, the angles, areas and volumes associated with random points, random lines and random planes

©2013 Tony Cai, Jianqing Fan and Tiefeng Jiang.

appear in the studies of stochastic geometry, see, for example, Stoyan, et al. (1995) and Kendall and Molchanov (2010).

In this paper we consider the empirical law and extreme laws of the pairwise angles among a large number of random unit vectors. More specifically, let $\mathbf{X}_1, \cdots, \mathbf{X}_n$ be random points independently chosen with the uniform distribution on $\mathbb{S}^{p-1}$, the unit sphere in $\mathbb{R}^p$. The $n$ points $\mathbf{X}_1, \cdots, \mathbf{X}_n$ on the sphere naturally generate $n$ unit vectors $\overrightarrow{\mathbf{OX}}_i$ for $i = 1, 2 \cdots, n$, where $\mathbf{O}$ is the origin. Let $0 \le \Theta_{ij} \le \pi$ denote the angle between $\overrightarrow{\mathbf{OX}}_i$ and $\overrightarrow{\mathbf{OX}}_j$ for all $1 \le i < j \le n$. In the case of a fixed dimension, the global behavior of the angles $\Theta_{ij}$ is captured by its empirical distribution

$$\mu_n = \frac{1}{\binom{n}{2}} \sum_{1 \le i < j \le n} \delta_{\Theta_{ij}}, \ n \ge 2. \tag{1}$$

When both the number of points $n$ and the dimension $p$ grow, it is more appropriate to consider the normalized empirical distribution

$$\mu_{n,p} = \frac{1}{\binom{n}{2}} \sum_{1 \le i < j \le n} \delta_{\sqrt{p-2}(\frac{\pi}{2} - \Theta_{ij})}, \ n \ge 2, \ p \ge 3. \tag{2}$$

In many applications it is of significant interest to consider the extreme angles $\Theta_{\min}$ and $\Theta_{\max}$ defined by

$$\Theta_{\min} = \min\{\Theta_{ij};\ 1 \le i < j \le n\}; \tag{3}$$

$$\Theta_{\max} = \max\{\Theta_{ij};\ 1 \le i < j \le n\}. \tag{4}$$

We will study both the empirical distribution of the angles $\Theta_{ij}$, $1 \le i < j \le n$, and the distributions of the extreme angles $\Theta_{\min}$ and $\Theta_{\max}$ as the number of points $n \to \infty$, while the dimension $p$ is either fixed or growing with $n$.

The distribution of minimum angle of $n$ points randomly distributed on the $p$-dimensional unit sphere has important implications in statistics and machine learning. It indicates how strong spurious correlations can be for $p$ observations of $n$-dimensional variables (Fan et al., 2012). It can be directly used to test isotropic of the distributions (see Section 4). It is also related to regularity conditions such as the Incoherent Condition (Donoho and Huo, 2001), the Restricted Eigenvalue Condition (Bickel et al., 2009), the $\ell_q$-Sensitivity (Gautier and Tsybakov, 2011) that are needed for sparse recovery. See also Section 5.1.

The present paper systematically investigates the asymptotic behaviors of the random angles $\{\Theta_{ij}; 1 \le i < j \le n\}$. It is shown that, when the dimension $p$ is fixed, as $n \to \infty$, the empirical distribution $\mu_n$ converges to a distribution with the density function given by

$$h(\theta) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \cdot (\sin\theta)^{p-2}, \ \theta \in [0, \pi].$$

On the other hand, when the dimension $p$ grows with $n$, it is shown that the limiting normalized empirical distribution $\mu_{n,p}$ of the random angles $\Theta_{ij}$, $1 \le i < j \le n$ is Gaussian. When the dimension is high, most of the angles are concentrated around $\pi/2$. The results provide a precise description of this concentration and thus give a rigorous theoretical justification to the folklore that "all high-dimensional random vectors are almost always nearly orthogonal to each other," see, for example,

Diaconis and Freedman (1984) and Hall et al. (2005). A more precise description is given in Proposition 5 later in terms of the concentration rate.

In addition to the empirical law of the angles $\Theta_{ij}$, we also consider the extreme laws of the random angles in both the fixed and growing dimension settings. The limiting distributions of the extremal statistics $\Theta_{\max}$ and $\Theta_{\min}$ are derived. Furthermore, the limiting distribution of the sum of the two extreme angles $\Theta_{\min} + \Theta_{\max}$ is also established. It shows that $\Theta_{\min} + \Theta_{\max}$ is highly concentrated at $\pi$.

The distributions of the minimum and maximum angles as well as the empirical distributions of all pairwise angles have important applications in statistics. First of all, they can be used to test whether a collection of random data points in the $p$-dimensional Euclidean space follow a spherically symmetric distribution (Fang et al., 1990). The natural test statistics are either $\mu_n$ or $\Theta_{\min}$ defined respectively in (1) and (3). The statistic $\Theta_{\min}$ also measures the maximum spurious correlation among $n$ data points in the $p$-dimensional Euclidean space. The correlations between a response vector with $n$ other variables, based on $n$ observations, are considered as spurious when they are smaller than a certain upper quantile of the distribution of $|\cos(\Theta_{\min})|$ (Fan and Lv, 2008). The statistic $\Theta_{\min}$ is also related to the bias of estimating the residual variance (Fan et al., 2012). More detailed discussion of the statistical applications of our studies is given in Section 4.

The study of the empirical law and the extreme laws of the random angles $\Theta_{ij}$ is closely connected to several deterministic open problems in physics and mathematics, including the general problem in physics of finding the minimum energy configuration of a system of particles on the surface of a sphere and the mathematical problem of uniformly distributing points on a sphere, which originally arises in complexity theory. The extreme laws of the random angles considered in this paper is also related to the study of the coherence of a random matrix, which is defined to be the largest magnitude of the Pearson correlation coefficients between the columns of the random matrix. See Cai and Jiang (2011, 2012) for the recent results and references on the distribution of the coherence. Some of these connections are discussed in more details in Section 5.

This paper is organized as follows. Section 2 studies the limiting empirical and extreme laws of the angles $\Theta_{ij}$ in the setting of the fixed dimension $p$ as the number of points $n$ going to $\infty$. The case of growing dimension is considered in Section 3. Their applications in statistics are outlined in Section 4. Discussions on the connections to the machine learning and some open problems in physics and mathematics are given in Section 5. The proofs of the main results are relegated in Section 6.

## 2. When The Dimension $p$ Is Fixed

In this section we consider the limiting empirical distribution of the angles $\Theta_{ij}$, $1 \le i < j \le n$ when the number of random points $n \to \infty$ while the dimension $p$ is fixed. The case where both $n$ and $p$ grow will be considered in the next section. Throughout the paper, we let $\mathbf{X}_1$, $\mathbf{X}_2$, $\cdots$, $\mathbf{X}_n$ be independent random points with the uniform distribution on the unit sphere $\mathbb{S}^{p-1}$ for some fixed $p \ge 2$.

We begin with the limiting empirical distribution of the random angles.

**Theorem 1 (Empirical Law for Fixed $p$)** *Let the empirical distribution $\mu_n$ of the angles $\Theta_{ij}$, $1 \le i < j \le n$, be defined as in* (1)*. Then, as $n \to \infty$, with probability one, $\mu_n$ converges weakly to the*

*distribution with density*

$$h(\theta) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \cdot (\sin\theta)^{p-2}, \ \theta \in [0,\pi]. \tag{5}$$

In fact, $h(\theta)$ is the probability density function of $\Theta_{ij}$ for any $i \neq j$ ($\Theta_{ij}$'s are identically distributed). Due to the dependency of $\Theta_{ij}$'s, some of them are large and some are small. Theorem 1 says that the average of these angles asymptotically has the same density as that of $\Theta_{12}$.

Notice that when $p = 2$, $h(\theta)$ is the uniform density on $[0,\pi]$, and when $p > 2$, $h(\theta)$ is unimodal with mode $\theta = \pi/2$. Theorem 1 implies that most of the angles in the total of $\binom{n}{2}$ angles are concentrated around $\pi/2$. This concentration becomes stronger as the dimension $p$ grows since $(\sin\theta)^{p-2}$ converges to zero more quickly for $\theta \neq \pi/2$. In fact, in the extreme case when $p \to \infty$, almost all of $\binom{n}{2}$ angles go to $\pi/2$ at the rate $\sqrt{p}$. This can be seen from Theorem 4 later.

It is helpful to see how the density changes with the dimension $p$. Figure 1 plots the function

$$\begin{aligned} h_p(\theta) &= \frac{1}{\sqrt{p-2}} h\Big(\frac{\pi}{2} - \frac{\theta}{\sqrt{p-2}}\Big) \\ &= \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})\sqrt{p-2}} \cdot \Big(\cos\frac{\theta}{\sqrt{p-2}}\Big)^{p-2}, \quad \theta \in [0,\pi] \end{aligned} \tag{6}$$

which is the asymptotic density of the normalized empirical distribution $\mu_{n,p}$ defined in (2) when the dimension $p$ is fixed. Note that in the definition of $\mu_{n,p}$ in (2), if "$\sqrt{p-2}$" is replaced by "$\sqrt{p}$", the limiting behavior of $\mu_{n,p}$ does not change when both $n$ and $p$ go to infinity. However, it shows in our simulations and the approximation (7) that the fitting is better for relatively small $p$ when "$\sqrt{p-2}$" is used.

Figure 1 shows that the distributions $h_p(\theta)$ are very close to normal when $p \geq 5$. This can also be seen from the asymptotic approximation

$$h_p(\theta) \propto \exp\Big((p-2)\log\big\{\cos\big(\frac{\theta}{\sqrt{p-2}}\big)\big\}\Big) \approx e^{-\theta^2/2}. \tag{7}$$

We now consider the limiting distribution of the extreme angles $\Theta_{\min}$ and $\Theta_{\max}$.

**Theorem 2 (Extreme Law for Fixed $p$)** *Let $\Theta_{\min}$ and $\Theta_{\max}$ be defined as in* (3) *and* (4) *respectively. Then, both $n^{2/(p-1)}\Theta_{\min}$ and $n^{2/(p-1)}(\pi - \Theta_{\max})$ converge weakly to a distribution given by*

$$F(x) = \begin{cases} 1 - e^{-Kx^{p-1}}, & \text{if } x \geq 0; \\ 0, & \text{if } x < 0, \end{cases} \tag{8}$$

*as $n \to \infty$, where*

$$K = \frac{1}{4\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p+1}{2})}. \tag{9}$$

The above theorem says that the smallest angle $\Theta_{\min}$ is close to zero, and the largest angle $\Theta_{\max}$ is close to $\pi$ as $n$ grows. This makes sense from Theorem 1 since the support of the density function $h(\theta)$ is $[0,\pi]$.

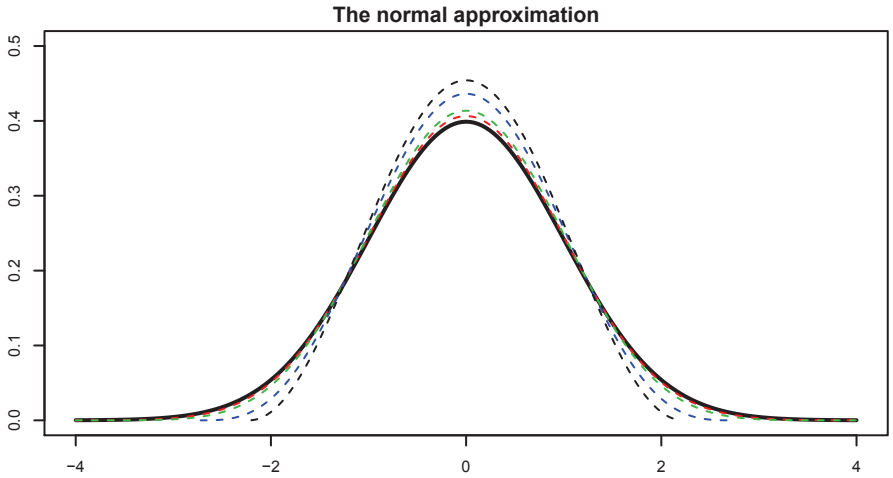



Figure 1: Functions $h_p(\theta)$ given by (6) for $p = 4, 5, 10$ and 20. They are getting closer to the normal density (thick black) as $p$ increases.

In the special case of $p = 2$, the scaling of $\Theta_{\min}$ and $\pi - \Theta_{\max}$ in Theorem 2 is $n^2$. This is in fact can also be seen in a similar problem. Let $\xi_1, \cdots, \xi_n$ be i.i.d. $U[0,1]$-distributed random variables with the order statistics $\xi_{(1)} \le \cdots \le \xi_{(n)}$. Set $W_n := \min_{1 \le i \le n-1}(\xi_{(i+1)} - \xi_{(i)})$, which is the smallest spacing among the observations of $\xi_i$'s. Then, by using the representation theorem of $\xi_{(i)}$'s through i.i.d. random variables with exponential distribution $\mathrm{Exp}(1)$ (see, for example, Proposition 4.1 from Resnick (2007)), it is easy to check that $n^2 W_n$ converges weakly to $\mathrm{Exp}(1)$ with the probability density function $e^{-x} I(x \ge 0)$.

To see the goodness of the finite sample approximations, we simulate 200 times from the distributions with $n = 50$ for $p = 2, 3$ and 30. The results are shown respectively in Figures 2–4. Figure 2 depicts the results when $p = 2$. In this case, the empirical distribution $\mu_n$ should approximately be uniformly distributed on $[0, \pi]$ for most of realizations. Figure 2 (a) shows that it holds approximately truly for $n$ as small as 50 for a particular realization (It indeed holds approximately for almost all realizations). Figure 2(b) plots the average of these 200 distributions, which is in fact extremely close to the uniform distribution on $[0, \pi]$. Namely, the bias is negligible. For $\Theta_{\min}$, according to Theorem 1, it should be well approximated by an exponential distribution with $K = 1/(2\pi)$. This is verified by Figure 2(c), even when sample size is as small as 50. Figure 2(d) shows the distribution of $\Theta_{\min} + \Theta_{\max}$ based on the 200 simulations. The sum is distributed tightly around $\pi$, which is indicated by the red line there.

The results for $p = 3$ and $p = 30$ are demonstrated in Figures 3 and 4. In this case, we show the empirical distributions of $\sqrt{p-2}(\pi/2 - \Theta_{ij})$ and their asymptotic distributions. As in Figure 1, they are normalized. Figure 3(a) shows a realization of the distribution and Figure 3(b) depicts the average of 200 realizations of these distributions for $p = 3$. They are very close to the asymptotic distribution, shown in the curve therein. The distributions of $\Theta_{\min}$ and $\Theta_{\max}$ are plotted in Figure 3(c). They concentrate respectively around 0 and $\pi$. Figure 3(d) shows that the sum is concentrated symmetrically around $\pi$.

When $p = 30$, the approximations are still very good for the normalized empirical distributions. In this case, the limiting distribution is indistinguishable from the normal density, as shown in Figure 1. However, the distribution of $\Theta_{\min}$ is not approximated well by its asymptotic counterpart,

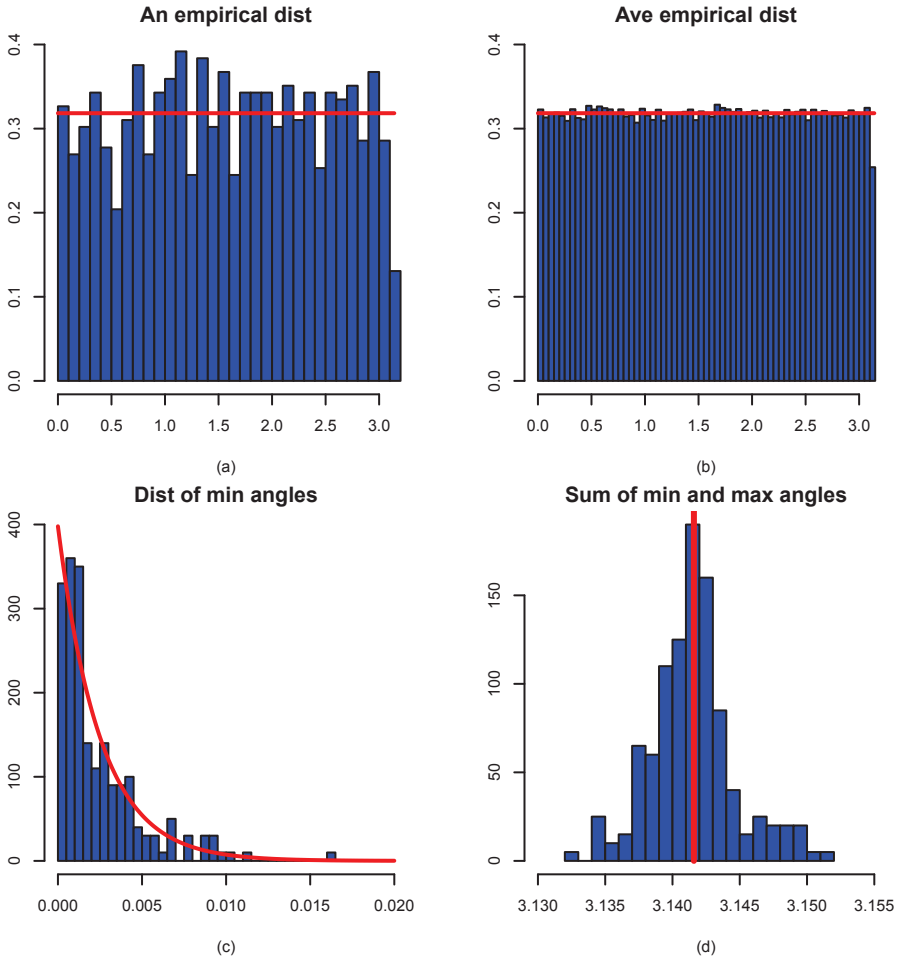



Figure 2: Various distributions for $p = 2$ and $n = 50$ based on 200 simulations. (a) A realization of the empirical distribution $\mu_n$; (b) The average distribution of 200 realizations of $\mu_n$; (c) the distribution of $\Theta_{\min}$ and its asymptotic distribution $\exp(-x/(2\pi))/(2\pi)$; (d) the distribution of $\Theta_{\min} + \Theta_{\max}$; the vertical line indicating the location $\pi$.

as shown in Figure 4(c). In fact, $\Theta_{\min}$ does not even tends to zero. This is not entirely surprising since $p$ is comparable with $n$. The asymptotic framework in Section 3 is more suitable. Nevertheless, $\Theta_{\min} + \Theta_{\max}$ is still symmetrically distributed around $\pi$.

The simulation results show that $\Theta_{\max} + \Theta_{\min}$ is very close to $\pi$. This actually can be seen trivially from Theorem 2: $\Theta_{\min} \to 0$ and $\Theta_{\max} \to \pi$ in probability as $p \to \infty$. Hence, the sum goes to $\pi$ in probability. An interesting question is: how fast is this convergence? The following result answers this question.

**Theorem 3 (Limit Law for Sum of Largest and Smallest Angles)** *Let* $\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_n$ *be independent random points with the uniform distribution on* $\mathbb{S}^{p-1}$ *for some fixed* $p \geq 2$. *Let* $\Theta_{\min}$ *and* $\Theta_{\max}$ *be defined as in* (3) *and* (4) *respectively. Then,* $n^{2/(p-1)}\left(\Theta_{\max} + \Theta_{\min} - \pi\right)$ *converges weakly to the distribution of* $X - Y$, *where* $X$ *and* $Y$ *are i.i.d. random variables with distribution function* $F(x)$ *given in (8).*

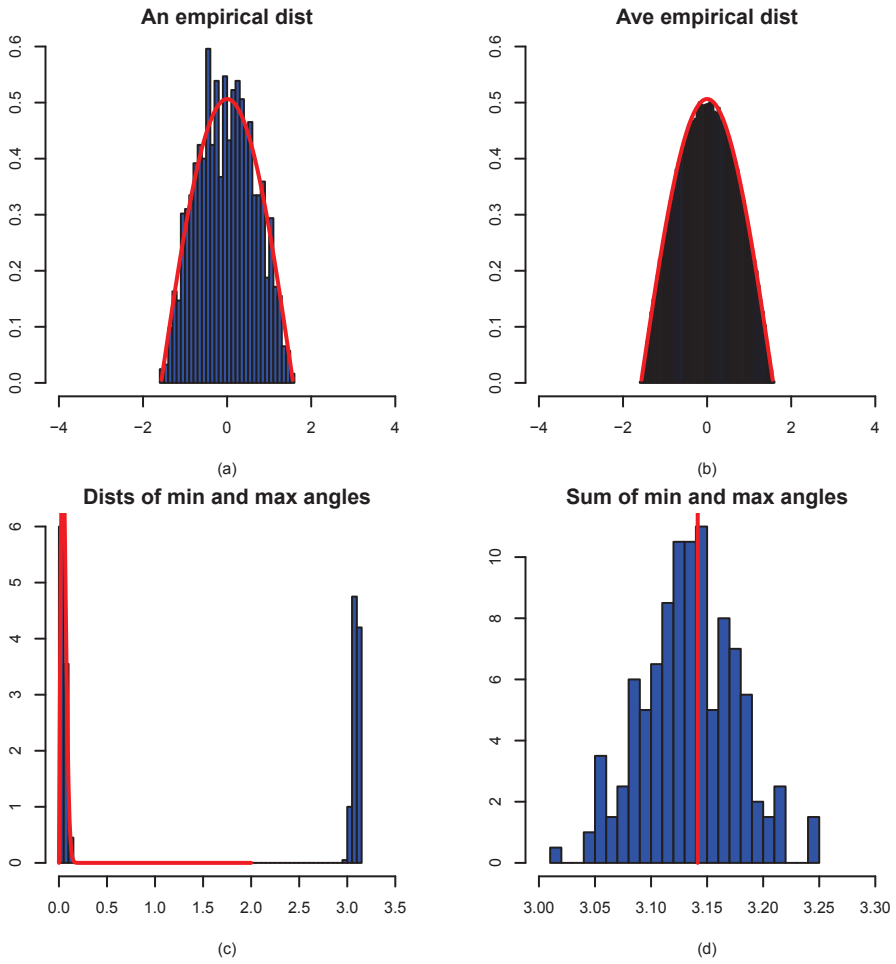



Figure 3: Various distributions for $p = 3$ and $n = 50$ based on 200 simulations. (a) A realization of the normalized empirical distribution $\mu_{n,p}$ given by (2); (b) The average distribution of 200 realizations of $\mu_{n,p}$; (c) the distribution of $\Theta_{\min}$ and its asymptotic distribution; (d) the distribution of $\Theta_{\min} + \Theta_{\max}$; the vertical line indicating the location $\pi$.

It is interesting to note that the marginal distribution of $\Theta_{\min}$ and $\pi - \Theta_{\max}$ are identical. However, $n^{2/(p-1)}\Theta_{\min}$ and $n^{2/(p-1)}(\pi - \Theta_{\max})$ are asymptotically independent with non-vanishing limits and hence their difference is non-degenerate. Furthermore, since $X$ are $Y$ are i.i.d., $X - Y$ is a symmetric random variable. Theorem 3 suggests that $\Theta_{\max} + \Theta_{\min}$ is larger or smaller than $\pi$ "equally likely". The symmetry of the distribution of $\Theta_{\max} + \Theta_{\min}$ has already been demonstrated in Figures 2–4.

## 3. When Both $n$ and $p$ Grow

We now turn to the case where both $n$ and $p$ grow. The following result shows that the empirical distribution of the random angles, after suitable normalization, converges to a standard normal distribution. This is clearly different from the limiting distribution given in Theorem 1 when the dimension $p$ is fixed.

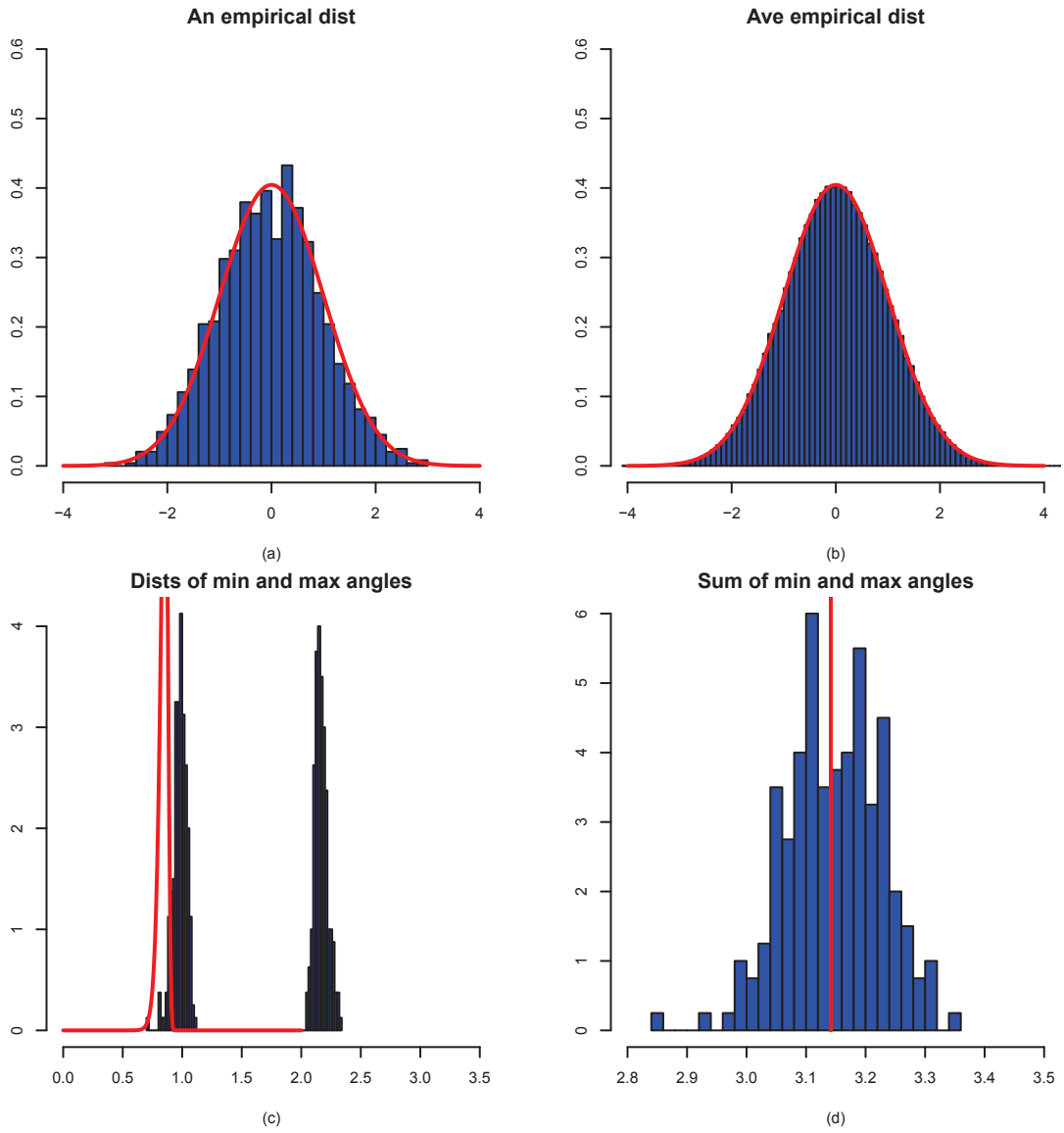



Figure 4: Various distributions for $p = 30$ and $n = 50$ based on 200 simulations. (a) A realization of the normalized empirical distribution $\mu_{n,p}$ given by (2); (b) The average distribution of 200 realizations of $\mu_{n,p}$; (c) the distribution of $\Theta_{\min}$ and its asymptotic distribution; (d) the distribution of $\Theta_{\min} + \Theta_{\max}$; the vertical line indicating the location $\pi$.

**Theorem 4 (Empirical Law for Growing $p$)** *Let $\mu_{n,p}$ be defined as in* (2). *Assume* $\lim_{n\to\infty} p_n = \infty$. *Then, with probability one, $\mu_{n,p}$ converges weakly to $N(0,1)$ as $n \to \infty$.*

Theorem 4 holds regardless of the speed of $p$ relative to $n$ when both go to infinity. This has also been empirically demonstrated in Figures 2–4 (see plots (a) and (b) therein). The theorem implies that most of the $\binom{n}{2}$ random angles go to $\pi/2$ very quickly. Take any $\gamma_p \to 0$ such that $\sqrt{p}\gamma_p \to \infty$ and denote by $N_{n,p}$ the number of the angles $\Theta_{ij}$ that are within $\gamma_p$ of $\pi/2$, that is, $|\frac{\pi}{2} - \Theta_{ij}| \le \gamma_p$. Then $N_{n,p}/\binom{n}{2} \to 1$. Hence, most of the random vectors in the high-dimensional Euclidean spaces

are nearly orthogonal. An interesting question is: Given two such random vectors, how fast is their angle close to $\pi/2$ as the dimension increases? The following result answers this question.

**Proposition 5** *Let* $\mathbf{U}$ *and* $\mathbf{V}$ *be two random points on the unit sphere in* $\mathbb{R}^p$. *Let* $\Theta$ *be the angle between* $\overrightarrow{\mathbf{OU}}$ *and* $\overrightarrow{\mathbf{OV}}$. *Then*

$$P(|\Theta - \frac{\pi}{2}| \geq \varepsilon) \leq K\sqrt{p}(\cos\varepsilon)^{p-2}$$

*for all* $p \geq 2$ *and* $\varepsilon \in (0, \pi/2)$, *where* $K$ *is a universal constant.*

Under the spherical invariance one can think of $\Theta$ as a function of the random point $\mathbf{U}$ only. There are general concentration inequalities on such functions, see, for example, Ledoux (2005). Proposition 5 provides a more precise inequality.

One can see that, as the dimension $p$ grows, the probability decays exponentially. In particular, take $\varepsilon = \sqrt{(c\log p)/p}$ for some constant $c > 1$. Note that $\cos\varepsilon \leq 1 - \varepsilon^2/2 + \varepsilon^4/24$, so

$$P\left(|\Theta - \frac{\pi}{2}| \geq \sqrt{\frac{c\log p}{p}}\right) \leq K\sqrt{p}\left(1 - \frac{c\log p}{2p} + \frac{c^2\log^2 p}{24p^2}\right)^{p-2} \leq K' p^{-\frac{1}{2}(c-1)}$$

for all sufficiently large $p$, where $K'$ is a constant depending only on $c$. Hence, in the high dimensional space, the angle between two random vectors is within $\sqrt{(c\log p)/p}$ of $\pi/2$ with high probability. This provides a precise characterization of the folklore mentioned earlier that "all high-dimensional random vectors are almost always nearly orthogonal to each other".

We now turn to the limiting extreme laws of the angles when both $n$ and $p \to \infty$. For the extreme laws, it is necessary to divide into three asymptotic regimes: sub-exponential case $\frac{1}{p}\log n \to 0$, exponential case $\frac{1}{p}\log n \to \beta \in (0,\infty)$, and super-exponential case $\frac{1}{p}\log n \to \infty$. The limiting extreme laws are different in these three regimes.

**Theorem 6 (Extreme Law: Sub-Exponential Case)** *Let* $p = p_n \to \infty$ *satisfy* $\frac{\log n}{p} \to 0$ *as* $n \to \infty$. *Then*

(i). $\max_{1\leq i<j\leq n} |\Theta_{ij} - \frac{\pi}{2}| \to 0$ *in probability as* $n \to \infty$;

(ii). *As* $n \to \infty$, $2p\log\sin\Theta_{\min} + 4\log n - \log\log n$ *converges weakly to the extreme value distribution with the distribution function* $F(y) = 1 - e^{-Ke^{y/2}}$, $y \in \mathbb{R}$ *and* $K = 1/(4\sqrt{2\pi})$. *The conclusion still holds if* $\Theta_{\min}$ *is replaced by* $\Theta_{\max}$.

In this case, both $\Theta_{\min}$ and $\Theta_{\max}$ converge to $\pi/2$ in probability. The above extreme value distribution differs from that in (8) where the dimension $p$ is fixed. This is obviously caused by the fact that $p$ is finite in Theorem 2 and goes to infinity in Theorem 6.

**Corollary 7** *Let* $p = p_n$ *satisfy* $\lim_{n\to\infty} \frac{\log n}{\sqrt{p}} = \alpha \in [0,\infty)$. *Then* $p\cos^2\Theta_{\min} - 4\log n + \log\log n$ *converges weakly to a distribution with the cumulative distribution function* $\exp\{-\frac{1}{4\sqrt{2\pi}}e^{-(y+8\alpha^2)/2}\}$, $y \in \mathbb{R}$. *The conclusion still holds if* $\Theta_{\min}$ *is replaced by* $\Theta_{\max}$.

**Theorem 8 (Extreme Law: Exponential Case)** *Let* $p = p_n$ *satisfy* $\frac{\log n}{p} \to \beta \in (0,\infty)$ *as* $n \to \infty$, *then*

(i). $\Theta_{\min} \to \cos^{-1}\sqrt{1-e^{-4\beta}}$ *and* $\Theta_{\max} \to \pi - \cos^{-1}\sqrt{1-e^{-4\beta}}$ *in probability as* $n \to \infty$;

(ii). *As* $n \to \infty$, $2p\log\sin\Theta_{\min} + 4\log n - \log\log n$ *converges weakly to a distribution with the distribution function*

$$F(y) = 1 - \exp\left\{-K(\beta)e^{(y+8\beta)/2}\right\},\ y \in \mathbb{R},\ \textit{where}\ K(\beta) = \left(\frac{\beta}{8\pi(1-e^{-4\beta})}\right)^{1/2},$$

*and the conclusion still holds if* $\Theta_{\min}$ *is replaced by* $\Theta_{\max}$.

In contrast to Theorem 6, neither $\Theta_{\max}$ nor $\Theta_{\min}$ converges to $\pi/2$ under the case that $(\log n)/p \to \beta \in (0,\infty)$. Instead, they converge to different constants depending on $\beta$.

**Theorem 9 (Extreme Law: Super-Exponential Case)** *Let* $p = p_n$ *satisfy* $\frac{\log n}{p} \to \infty$ *as* $n \to \infty$. *Then,*

(i). $\Theta_{\min} \to 0$ *and* $\Theta_{\max} \to \pi$ *in probability as* $n \to \infty$;

(ii). *As* $n \to \infty$, $2p\log\sin\Theta_{\min} + \frac{4p}{p-1}\log n - \log p$ *converges weakly to the extreme value distribution with the distribution function* $F(y) = 1 - e^{-Ke^{y/2}},\ y \in \mathbb{R}$ *with* $K = 1/(2\sqrt{2\pi})$. *The conclusion still holds if* $\Theta_{\min}$ *is replaced by* $\Theta_{\max}$.

It can be seen from Theorems 6, 8 and 9 that $\Theta_{\max}$ becomes larger when the rate $\beta = \lim(\log n)/p$ increases. They are $\pi/2$, $\pi - \cos^{-1}\sqrt{1-e^{-4\beta}} \in (\pi/2,\pi)$ and $\pi$ when $\beta = 0$, $\beta \in (0,\infty)$ and $\beta = \infty$, respectively.

Set $f(\beta) = \pi - \cos^{-1}\sqrt{1-e^{-4\beta}}$. Then $f(0) = \pi/2$ and $f(+\infty) = \pi$, which corresponds to $\Theta_{\max}$ in (i) of Theorem 6 and (i) of Theorem 9, respectively. So the conclusions in Theorems 6, 8 and 9 are consistent.

Theorem 3 provides the limiting distribution of $\Theta_{\max} + \Theta_{\min} - \pi$ when the dimension $p$ is fixed. It is easy to see from the above theorems that $\Theta_{\max} + \Theta_{\min} - \pi \to 0$ in probability as both $n$ and $p$ go to infinity. Its asymptotic distribution is much more involved and we leave it as future work.

**Remark 10** As mentioned in the introduction, Cai and Jiang (2011, 2012) considered the limiting distribution of the coherence of a random matrix and the coherence is closely related to the minimum angle $\Theta_{\min}$. In the current setting, the coherence $L_{n,p}$ is defined by

$$L_{n,p} = \max_{1\le i<j\le n} |\rho_{ij}|$$

where $\rho_{ij} = \mathbf{X}_i^T\mathbf{X}_j$. The results in Theorems 6, 8 and 9 are new. Their proofs can be essentially reduced to the analysis of $\max_{1\le i<j\le n}\rho_{ij}$. This maximum is analyzed through modifying the proofs of the results for the limiting distribution of the coherence $L_{n,p}$ in Cai and Jiang (2012). The key step in the proofs is the study of the maximum and minimum of *pairwise* i.i.d. random variables $\{\rho_{ij};\ 1 \le i < j \le n\}$ by using the Chen-Stein method. It is noted that $\{\rho_{ij};\ 1 \le i < j \le n\}$ are not i.i.d. random variables (see, for example, p.148 from Muirhead (1982)), the standard techniques to analyze the extreme values of $\{\rho_{ij};\ 1 \le i < j \le n\}$ do not apply.

## 4. Applications to Statistics

The results developed in the last two sections can be applied to test the spherical symmetry (Fang et al., 1990):

$$H_0 : \mathbf{Z} \text{ is spherically symmetric in } \mathbb{R}^p$$

based on an i.i.d. sample $\{\mathbf{Z}_i\}_{i=1}^n$. Under the null hypothesis $H_0$, $\mathbf{Z}/\|\mathbf{Z}\|$ is uniformly distributed on $\mathbb{S}^{p-1}$. It is expected that the minimum angle $\Theta_{\min}$ is stochastically larger under the null hypothesis than that under the alternative hypothesis. Therefore, one should reject the null hypothesis when $\Theta_{\min}$ is too small or formally, reject $H_0$ when

$$n^{2/(p-1)}\Theta_{\min} \le c_\alpha,$$

where the critical value $c_\alpha$, according to Theorem 2, is given by

$$c_\alpha = \left(-K^{-1}\log(1-\alpha)\right)^{1/(p-1)}$$

for the given significance level $\alpha$. This provides the minimum angle test for sphericity or the packing test on sphericity.

We run a simulation study to examine the power of the packing test. The following 6 data generating processes are used:

*Distribution 0*: the components of $\mathbf{X}$ follow independently the standard normal distribution;

*Distribution 1*: the components of $\mathbf{X}$ follow independently the uniform distribution on $[-1,1]$;

*Distribution 2*: the components of $\mathbf{X}$ follow independently the uniform distribution on $[0,1]$;

*Distribution 3*: the components of $\mathbf{X}$ follow the standard normal distribution with correlation 0.5;

*Distribution 4*: the components of $\mathbf{X}$ follow the standard normal distribution with correlation 0.9;

*Distribution 5*: the components of $\mathbf{X}$ follow independently the mixture distribution $2/3\exp(-x)I(x \ge 0) + 1/3\exp(x)I(x \le 0)$.

The results are summarized in Table 1 below. Note that for Distribution 0, the power corresponds to the size of the test, which is slightly below $\alpha = 5\%$.

| Distribution | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| $p=2$ | 4.20 | 5.20 | 20.30 | 5.55 | 10.75 | 5.95 |
| $p=3$ | 4.20 | 6.80 | 37.20 | 8.00 | 30.70 | 8.05 |
| $p=4$ | 4.80 | 7.05 | 64.90 | 11.05 | 76.25 | 11.20 |
| $p=5$ | 4.30 | 7.45 | 90.50 | 18.25 | 99.45 | 11.65 |

Table 1: The power (percent of rejections) of the packing test based on 2000 simulations

The packing test does not examine whether there is a gap in the data on the sphere. An alternative test statistic is $\mu_n$ or its normalized version $\mu_{n,p}$ when $p$ is large, defined respectively by (1) and (2). A natural test statistic is then to use a distance such as the Kolmogrov-Smirnov distance between $\mu_n$

and $h(\theta)$. In this case, one needs to derive further the null distribution of such a test statistic. This is beyond the scope of this paper and we leave it for future work.

Our study also shed lights on the magnitude of spurious correlation. Suppose that we have a response variable $Y$ and its associate covariates $\{X_j\}_{j=1}^p$ (for example, gene expressions). Even when there is no association between the response and the covariate, the maximum sample correlation between $X_j$ and $Y$ based on a random sample of size $n$ will not be zero. It is closely related to the minimum angle $\Theta_{\min}$ (Fan and Lv, 2008). Any correlation below a certain thresholding level can be spurious—the correlation of such a level can occur purely by chance. For example, by Theorem 6(ii), any correlation (in absolute value) below

$$\sqrt{1-n^{-4/p}(\log(n))^{1/p}}$$

can be regarded as the spurious one. Take, for example, $p = 30$ and $n = 50$ as in Figure 4, the spurious correlation can be as large 0.615 in this case.

The spurious correlation also helps understand the bias in calculating the residual $\sigma^2 = \text{var}(\varepsilon)$ in the sparse linear model

$$Y = \mathbf{X}_S^T \beta_S + \varepsilon$$

where $S$ is a subset of variables $\{1, \cdots p\}$. When an extra variable besides $X_S$ is recruited by a variable selection algorithm, that extra variable is recruited to best predict $\varepsilon$ (Fan et al., 2012). Therefore, by the classical formula for the residual variance, $\sigma^2$ is underestimated by a factor of $1 - \cos^2(\Theta_{\min})$. Our asymptotic result gives the order of magnitude of such a bias.

## 5. Discussions

We have established the limiting empirical and extreme laws of the angles between random unit vectors, both for the fixed dimension and growing dimension cases. For fixed $p$, we study the empirical law of angles, the extreme law of angles and the law of the sum of the largest and smallest angles in Theorems 1, 2 and 3. Assuming $p$ is large, we establish the empirical law of random angles in Theorem 4. Given two vectors $\mathbf{u}$ and $\mathbf{v}$, the cosine of their angle is equal to the Pearson correlation coefficient between them. Based on this observation, among the results developed in this paper, the limiting distribution of the minimum angle $\Theta_{\min}$ given in Theorems 6-9 for the setting where both $n$ and $p \to \infty$ is obtained by similar arguments to those in Cai and Jiang (2012) on the coherence of an $n \times p$ random matrix (a detailed discussion is given in Remark 10). See also Jiang (2004), Li and Rosalsky (2006), Zhou (2007), Liu et al. (2008), Li et al. (2009) and Li et al. (2010) for earlier results on the distribution of the coherence which were all established under the assumption that both $n$ and $p \to \infty$.

The study of the random angles $\Theta_{ij}$'s, $\Theta_{\min}$ and $\Theta_{\max}$ is also related to several problems in machine learning as well as some deterministic open problems in physics and mathematics. We briefly discuss some of these connections below.

### 5.1 Connections to Machine Learning

Our studies shed lights on random geometric graphs, which are formed by $n$ random points on the $p$-dimensional unit sphere as vertices with edge connecting between points $\mathbf{X}_i$ and $\mathbf{X}_j$ if $\Theta_{ij} > \delta$ for certain $\delta$ (Penrose, 2003; Devroye et al., 2011). Like testing isotropicity in Section 4, a generalization of our results can be used to detect if there are any implanted cliques in a random

graph, which is a challenging problem in machine learning. It can also be used to describe the distributions of the number of edges and degree of such a random geometric graph. Problems of hypothesis testing on isotropicity of covariance matrices have strong connections with clique numbers of geometric random graphs as demonstrated in the recent manuscript by Castro et al. (2012). This furthers connections of our studies in Section 4 to this machine learning problem.

Principal component analysis (PCA) is one of the most important techniques in high-dimensional data analysis for visualization, feature extraction, and dimension reduction. It has a wide range of applications in statistics and machine learning. A key aspect of the study of PCA in the high-dimensional setting is the understanding of the properties of the principal eigenvectors of the sample covariance matrix. In a recent paper, Shen et al. (2013) showed an interesting asymptotic conical structure in the critical sample eigenvectors under a spike covariance models when the ratio between the dimension and the product of the sample size with the spike size converges to a nonzero constant. They showed that in such a setting the critical sample eigenvectors lie in a right circular cone around the corresponding population eigenvectors. Although these sample eigenvectors converge to the cone, their locations within the cone are random. The behavior of the randomness of the eigenvectors within the cones is related to the behavior of the random angles studied in the present paper. It is of significant interest to rigorously explore these connections. See Shen et al. (2013) for further discussions.

### 5.2 Connections to Some Open Problems in Mathematics and Physics

The results on random angles established in this paper can be potentially used to study a number of open deterministic problems in mathematics and physics.

Let $\mathbf{x}_1, \cdots, \mathbf{x}_n$ be $n$ points on $\mathbb{S}^{p-1}$ and $R = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$. The $\alpha$-energy function is defined by

$$E(R,\alpha) = \begin{cases} \sum_{1\le i<j\le n} \|\mathbf{x}_i - \mathbf{x}_j\|^{\alpha}, & \text{if } \alpha \neq 0; \\ \sum_{1\le i<j\le n} \log \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|}, & \text{if } \alpha = 0, \end{cases}$$

and $E(R,-\infty) = \min_{1\le i<j\le n} \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|}$ where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^p$. These are known as the electron problem ($\alpha = 0$) and the Coulomb potential problem ($\alpha = -1$). See, for example, Kuijlaars and Saff (1998) and Katanforoush and Shahshahani (2003). The goal is to find the extremal $\alpha$-energy

$$\varepsilon(R,\alpha) := \begin{cases} \inf_R E(R,\alpha), & \text{if } \alpha \le 0, \\ \sup_R E(R,\alpha), & \text{if } \alpha > 0, \end{cases}$$

and the extremal configuration $R$ that attains $\varepsilon(R,\alpha)$. In particular, when $\alpha = -1$, the quantity $\varepsilon(R,-1)$ is the minimum of the Coulomb potential

$$\sum_{1\le i<j\le n} \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|}.$$

These open problems, as a function of $\alpha$, are: (i) $\alpha = -\infty$: Tammes problem; (ii) $\alpha = -1$: Thomson problem; (iii) $\alpha = 1$: maximum average distance problem; and (iv) $\alpha = 0$: maximal product of distances between all pairs. Problem (iv) is the 7th of the 17 most challenging mathematics problems in the 21st century according to Smale (2000). See, for example, Kuijlaars and Saff (1998) and Katanforoush and Shahshahani (2003), for further details.

The above problems can also be formulated through randomization. Suppose that $\mathbf{X}_1, \cdots, \mathbf{X}_n$ are i.i.d. uniform random vectors on $\mathbb{S}^{p-1}$. Suppose $R = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ achieves the infinimum or supremum in the definition of $\varepsilon(R, \alpha)$. Since $P(\max_{1 \le i \le n} \|\mathbf{X}_i - \mathbf{x}_i\| < \varepsilon) > 0$ for any $\varepsilon > 0$, it is easy to see that $\varepsilon(R, \alpha) = \text{ess} \cdot \inf(E(R, \alpha))$ for $\alpha \le 0$ and $\varepsilon(R, \alpha) = \text{ess} \cdot \sup(E(R, \alpha))$ for $\alpha > 0$ with $R = \{\mathbf{X}_1, \cdots, \mathbf{X}_n\}$, where $\text{ess} \cdot \inf(Z)$ and $\text{ess} \cdot \sup(Z)$ are the essential infinimum and the essential maximum of random variable $Z$, respectively.

For the Tammes problem ($\alpha = -\infty$), the extremal energy $\varepsilon(R, -\infty)$ can be further studied through the random variable $\Theta_{\max}$. Note that $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = 2(1 - \cos\theta_{ij})$, where $\theta_{ij}$ is the angle between vectors $\overrightarrow{\mathbf{Ox}}_i$ and $\overrightarrow{\mathbf{Ox}}_j$. Then

$$\frac{1}{2E(R, -\infty)^2} = \max_{\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{S}^{p-1}} (1 - \cos\theta_{ij}) = 1 - \cos\tilde{\Theta}_{\max},$$

where $\tilde{\Theta}_{\max} = \max\{\theta_{ij}; 1 \le i < j \le n\}$. Again, let $\mathbf{X}_1, \cdots, \mathbf{X}_n$ be i.i.d. random vectors with the uniform distribution on $\mathbb{S}^{p-1}$. Then, it is not difficult to see

$$\frac{1}{2\varepsilon(R, -\infty)^2} = \sup_R \frac{1}{2E(R, -\infty)^2} = \sup_R (1 - \cos\tilde{\Theta}_{\max}) = 1 - \cos\Delta$$

where $\Delta := \text{ess} \cdot \sup(\Theta_{\max})$ is the essential upper bound of the random variable $\Theta_{\max}$ as defined in (4). Thus,

$$\varepsilon(R, -\infty) = \frac{1}{\sqrt{2(1 - \cos\Delta)}}. \tag{10}$$

The essential upper bound $\Delta$ of the random variable $\Theta_{\max}$ can be approximated by random sampling of $\Theta_{\max}$. So the approach outlined above provides a direct way for using a stochastic method to study these deterministic problems and establishes connections between the random angles and open problems mentioned above. See, for example, Katanforoush and Shahshahani (2003) for further comments on randomization. Recently, Armentano et al. (2011) studied this problem by taking $\mathbf{x}_i$'s to be the roots of a special type of random polynomials. Taking independent and uniform samples $\mathbf{X}_1, \cdots, \mathbf{X}_n$ from the unit sphere $\mathbb{S}^{p-1}$ to get (10) is simpler than using the roots of a random polynomials.

## 6. Proofs

We provide the proofs of the main results in this section.

### 6.1 Technical Results

Recall that $\mathbf{X}_1, \mathbf{X}_2, \cdots$ are random points independently chosen with the uniform distribution on $\mathbb{S}^{p-1}$, the unit sphere in $\mathbb{R}^p$, and $\Theta_{ij}$ is the angle between $\overrightarrow{\mathbf{OX}}_i$ and $\overrightarrow{\mathbf{OX}}_j$ and $\rho_{ij} = \cos\Theta_{ij}$ for any $i \ne j$. Of course, $\Theta_{ij} \in [0, \pi]$ for all $i \ne j$. It is known that the distribution of $(\mathbf{X}_1, \mathbf{X}_2, \cdots)$ is the same as that of

$$\left(\frac{\mathbf{Y}_1}{\|\mathbf{Y}_1\|}, \frac{\mathbf{Y}_2}{\|\mathbf{Y}_2\|}, \cdots\right)$$

where $\{\mathbf{Y}_1, \mathbf{Y}_2, \cdots\}$ are independent $p$-dimensional random vectors with the normal distribution $N_p(\mathbf{0}, \mathbf{I}_p)$, that is, the normal distribution with mean vector $\mathbf{0}$ and the covariance matrix equal to the $p \times p$ identity matrix $\mathbf{I}_p$. Thus,

$$\rho_{ij} = \cos\Theta_{ij} = \frac{\mathbf{Y}_i^T \mathbf{Y}_j}{\|\mathbf{Y}_i\| \cdot \|\mathbf{Y}_i\|}$$

for all $1 \le i < j \le n$. See, for example, the Discussions in Section 5 from Cai and Jiang (2012) for further details. Of course, $\rho_{ii} = 1$ and $|\rho_{ij}| \le 1$ for all $i, j$. Set

$$M_n = \max_{1 \le i < j \le n} \rho_{ij} = \cos\Theta_{\min}. \tag{11}$$

**Lemma 11** *((22) in Lemma 4.2 from Cai and Jiang (2012)) Let $p \ge 2$. Then $\{\rho_{ij}; 1 \le i < j \le n\}$ are pairwise independent and identically distributed with density function*

$$g(\rho) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \cdot (1-\rho^2)^{\frac{p-3}{2}}, \quad |\rho| < 1. \tag{12}$$

Notice $y = \cos x$ is a strictly decreasing function on $[0, \pi]$, hence $\Theta_{ij} = \cos^{-1}\rho_{ij}$. A direct computation shows that Lemma 11 is equivalent to the following lemma.

**Lemma 12** *Let $p \ge 2$. Then,*

*(i) $\{\Theta_{ij}; 1 \le i < j \le n\}$ are pairwise independent and identically distributed with density function*

$$h(\theta) = \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \cdot (\sin\theta)^{p-2}, \quad \theta \in [0, \pi]. \tag{13}$$

*(ii) If "$\Theta_{ij}$" in (i) is replaced by "$\pi - \Theta_{ij}$", the conclusion in (i) still holds.*

Let $I$ be a finite set, and for each $\alpha \in I$, $X_\alpha$ be a Bernoulli random variable with $p_\alpha = P(X_\alpha = 1) = 1 - P(X_\alpha = 0) > 0$. Set $W = \sum_{\alpha \in I} X_\alpha$ and $\lambda = EW = \sum_{\alpha \in I} p_\alpha$. For each $\alpha \in I$, suppose we have chosen $B_\alpha \subset I$ with $\alpha \in B_\alpha$. Define

$$b_1 = \sum_{\alpha \in I} \sum_{\beta \in B_\alpha} p_\alpha p_\beta \ \text{ and } \ b_2 = \sum_{\alpha \in I} \sum_{\alpha \ne \beta \in B_\alpha} P(X_\alpha = 1, X_\beta = 1).$$

**Lemma 13** *(Theorem 1 from Arratia et al. (1989)) For each $\alpha \in I$, assume $X_\alpha$ is independent of $\{X_\beta; \beta \in I - B_\alpha\}$. Then $\left|P(X_\alpha = 0 \text{ for all } \alpha \in I) - e^{-\lambda}\right| \le b_1 + b_2$.*

The following is essentially a special case of Lemma 13.

**Lemma 14** *Let $I$ be an index set and $\{B_\alpha, \alpha \in I\}$ be a set of subsets of $I$, that is, $B_\alpha \subset I$ for each $\alpha \in I$. Let also $\{\eta_\alpha, \alpha \in I\}$ be random variables. For a given $t \in \mathbb{R}$, set $\lambda = \sum_{\alpha \in I} P(\eta_\alpha > t)$. Then*

$$|P(\max_{\alpha \in I} \eta_\alpha \le t) - e^{-\lambda}| \le (1 \wedge \lambda^{-1})(b_1 + b_2 + b_3)$$

*where*

$$b_1 = \sum_{\alpha\in I}\sum_{\beta\in B_\alpha} P(\eta_\alpha > t)P(\eta_\beta > t),\ \ b_2 = \sum_{\alpha\in I}\sum_{\alpha\neq\beta\in B_\alpha} P(\eta_\alpha > t, \eta_\beta > t),$$
$$b_3 = \sum_{\alpha\in I} E|P(\eta_\alpha > t|\sigma(\eta_\beta, \beta\notin B_\alpha)) - P(\eta_\alpha > t)|,$$

*and* $\sigma(\eta_\beta, \beta\notin B_\alpha)$ *is the* $\sigma$*-algebra generated by* $\{\eta_\beta, \beta\notin B_\alpha\}$. *In particular, if* $\eta_\alpha$ *is independent of* $\{\eta_\beta, \beta\notin B_\alpha\}$ *for each* $\alpha$, *then* $b_3 = 0$.

**Lemma 15** *Let* $p = p_n \geq 2$. *Recall* $M_n$ *as in (11). For* $\{t_n \in [0,1]; n \geq 2\}$, *set*

$$h_n = \frac{n^2 p^{1/2}}{\sqrt{2\pi}} \int_{t_n}^1 (1-x^2)^{\frac{p-3}{2}}\, dx.$$

*If* $\lim_{n\to\infty} p_n = \infty$ *and* $\lim_{n\to\infty} h_n = \lambda \in [0,\infty)$, *then* $\lim_{n\to\infty} P(M_n \leq t_n) = e^{-\lambda/2}$.

**Proof**. For brevity of notation, we sometimes write $t = t_n$ if there is no confusion. First, take $I = \{(i,j);\ 1 \leq i < j \leq n\}$. For $u = (i,j) \in I$, set $B_u = \{(k,l) \in I;$ one of $k$ and $l = i$ or $j$, but $(k,l) \neq u\}$, $\eta_u = \rho_{ij}$ and $A_u = A_{ij} = \{\rho_{ij} > t\}$. By the i.i.d. assumption on $\mathbf{X}_1, \cdots, \mathbf{X}_n$ and Lemma 14,

$$|P(M_n \leq t) - e^{-\lambda_n}| \leq b_{1,n} + b_{2,n} \tag{14}$$

where

$$\lambda_n = \frac{n(n-1)}{2} P(A_{12}) \tag{15}$$

and

$$b_{1,n} \leq 2n^3 P(A_{12})^2 \text{ and } b_{2,n} \leq 2n^3 P(A_{12}A_{13}).$$

By Lemma 11, $A_{12}$ and $A_{13}$ are independent events with the same probability. Thus, from (15),

$$b_{1,n} \vee b_{2,n} \leq 2n^3 P(A_{12})^2 \leq \frac{8n\lambda_n^2}{(n-1)^2} \leq \frac{32\lambda_n^2}{n} \tag{16}$$

for all $n \geq 2$. Now we compute $P(A_{12})$. In fact, by Lemma 11 again,

$$P(A_{12}) = \int_t^1 g(x)\, dx \quad = \quad \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \int_t^1 (1-x^2)^{\frac{p-3}{2}}\, dx.$$

Recalling the Stirling formula (see, for example, p.368 from Gamelin (2001) or (37) on p.204 from Ahlfors (1979)):

$$\log\Gamma(z) = z\log z - z - \frac{1}{2}\log z + \log\sqrt{2\pi} + O\left(\frac{1}{x}\right)$$

as $x = \mathrm{Re}\,(z) \to \infty$, it is easy to verify that

$$\frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \sim \sqrt{\frac{p}{2}} \tag{17}$$

as $p \to \infty$. Thus,

$$P(A_{12}) \sim \frac{p^{1/2}}{\sqrt{2\pi}} \int_t^1 (1-x^2)^{\frac{p-3}{2}}\, dx$$

as $n \to \infty$. From (15), we know

$$\lambda_n \sim \frac{p^{1/2} n^2}{2\sqrt{2\pi}} \int_t^1 (1-x^2)^{\frac{p-3}{2}}\, dx = \frac{h_n}{2}$$

as $n \to \infty$. Finally, by (14) and (16), we know

$$\lim_{n\to\infty} P(M_n \le t) = e^{-\lambda/2} \text{ if } \lim_{n\to\infty} h_n = \lambda \in [0,\infty). \qquad \blacksquare$$

### 6.2 Proofs of Main Results in Section 2

**Lemma 16** *Let $\mathbf{X}_1, \mathbf{X}_2, \cdots$ be independent random points with the uniform distribution on the unit sphere in $\mathbb{R}^p$.*

*(i) Let $p$ be fixed and $\mu$ be the probability measure with the density $h(\theta)$ as in (5). Then, with probability one, $\mu_n$ in (1) converges weakly to $\mu$ as $n \to \infty$.*

*(ii) Let $p = p_n$ and $\{\varphi_n(\theta); n \ge 1\}$ be a sequence of functions defined on $[0,\pi]$. If $\varphi_n(\Theta_{12})$ converges weakly to a probability measure $\nu$ as $n \to \infty$, then, with probability one,*

$$\nu_n := \frac{1}{\binom{n}{2}} \sum_{1\le i<j\le n} \delta_{\varphi_n(\Theta_{ij})} \tag{18}$$

*converges weakly to $\nu$ as $n \to \infty$.*

**Proof**. First, we claim that, for any bounded and continuous function $u(x)$ defined on $\mathbb{R}$,

$$\frac{1}{\binom{n}{2}} \sum_{1\le i<j\le n} [u(\varphi_n(\Theta_{ij})) - Eu(\varphi_n(\Theta_{ij}))] \to 0 \ \ a.s. \tag{19}$$

as $n \to \infty$ regardless $p$ is fixed as in (i) or $p = p_n$ as in (ii) in the statement of the lemma. For convenience, write $u_n(\theta) = u(\varphi_n(\theta))$. Then $u_n(\theta)$ is a bounded function with $M := \sup_{\theta\in[0,\pi]} |u_n(\theta)| < \infty$. By the Markov inequality

$$\begin{aligned} & P\Big(\Big|\sum_{1\le i<j\le n} (u_n(\Theta_{ij}) - Eu_n(\Theta_{ij}))\Big| \ge \varepsilon \binom{n}{2}\Big) \\ \le\ & \frac{1}{\binom{n}{2}^2 \varepsilon^2} E\Big|\sum_{1\le i<j\le n} (u_n(\Theta_{ij}) - Eu_n(\Theta_{ij}))\Big|^2 \end{aligned}$$

for any $\varepsilon > 0$. From (i) of Lemma 12, $\{\Theta_{ij}; 1 \le i < j \le n\}$ are pairwise independent with the common distribution, the last expectation is therefore equal to $\binom{n}{2}\mathrm{Var}(u_n(\Theta_{12})) \le \binom{n}{2} M^2$. This says that, for any $\varepsilon > 0$,

$$P\Big(\Big|\sum_{1\le i<j\le n} (u_n(\Theta_{ij}) - Eu_n(\Theta_{ij}))\Big| \ge \varepsilon \binom{n}{2}\Big) = O\Big(\frac{1}{n^2}\Big)$$

as $n \to \infty$. Note that the sum of the right hand side over all $n \geq 2$ is finite. By the Borel-Cantelli lemma, we conclude (19).

(i) Take $\varphi_n(\theta) = \theta$ for $\theta \in \mathbb{R}$ in (19) to get that

$$\frac{1}{\binom{n}{2}} \sum_{1 \leq i < j \leq n} u(\Theta_{ij}) \to Eu(\Theta_{12}) = \int_0^\pi u(\theta) h(\theta)\, d\theta \ \ a.s.$$

as $n \to \infty$, where $u(\theta)$ is any bounded continuous function on $[0, \pi]$ and $h(\theta)$ is as in (5). This leads to that, with probability one, $\mu_n$ in (1) converges weakly to $\mu$ as $n \to \infty$.

(ii) Since $\varphi_n(\Theta_{12})$ converges weakly to $\nu$ as $n \to \infty$, we know that, for any bounded continuous function $u(x)$ defined on $\mathbb{R}$, $Eu(\varphi_n(\Theta_{12})) \to \int_{-\infty}^{\infty} u(x)\, d\nu(x)$ as $n \to \infty$. By (i) of Lemma 12, $Eu(\varphi_n(\Theta_{ij})) = Eu(\varphi_n(\Theta_{12}))$ for all $1 \leq i < j \leq n$. This and (19) yield

$$\frac{1}{\binom{n}{2}} \sum_{1 \leq i < j \leq n} u(\varphi_n(\Theta_{ij})) \to \int_{-\infty}^{\infty} u(x)\, d\nu(x) \ \ a.s.$$

as $n \to \infty$. Reviewing the definition of $\nu_n$ in (18), the above asserts that, with probability one, $\nu_n$ converges weakly to $\nu$ as $n \to \infty$. ■

**Proof of Theorem 1**. This is a direct consequence of (i) of Lemma 16. ■

Recall $\mathbf{X}_1, \cdots, \mathbf{X}_n$ are random points independently chosen with the uniform distribution on $\mathbb{S}^{p-1}$, the unit sphere in $\mathbb{R}^p$, and $\Theta_{ij}$ is the angle between $\overrightarrow{\mathbf{OX}}_i$ and $\overrightarrow{\mathbf{OX}}_j$ and $\rho_{ij} = \cos\Theta_{ij}$ for all $1 \leq i, j \leq n$. Of course, $\rho_{ii} = 1$ and $|\rho_{ij}| \leq 1$ for all $1 \leq i \neq j \leq n$. Review (11) to have

$$M_n = \max_{1 \leq i < j \leq n} \rho_{ij} = \cos\Theta_{\min}.$$

To prove Theorem 2, we need the following result.

**Proposition 17** *Fix $p \geq 2$. Then $n^{4/(p-1)}(1 - M_n)$ converges to the distribution function*

$$F_1(x) = 1 - \exp\{-K_1 x^{(p-1)/2}\}, \ \ x \geq 0,$$

*in distribution as $n \to \infty$, where*

$$K_1 = \frac{2^{(p-5)/2}}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p+1}{2})}. \tag{20}$$

**Proof**. Set $t = t_n = 1 - x n^{-4/(p-1)}$ for $x \geq 0$. Then

$$t \to 1 \ \text{ and } \ t^2 = 1 - \frac{2x}{n^{4/(p-1)}} + O\Big(\frac{1}{n^{8/(p-1)}}\Big) \tag{21}$$

as $n \to \infty$. Notice

$$P(n^{4/(p-1)}(1 - M_n) < x) = P(M_n > t) = 1 - P(M_n \leq t).$$

Thus, to prove the theorem, since $F_1(x)$ is continuous, it is enough to show that

$$P(M_n \leq t) \to e^{-K_1 x^{(p-1)/2}} \tag{22}$$

as $n \to \infty$, where $K_1$ is as in (20).

Now, take $I = \{(i,j);\ 1 \le i < j \le n\}$. For $u = (i,j) \in I$, set $B_u = \{(k,l) \in I;$ one of $k$ and $l =$ $i$ or $j$, but $(k,l) \ne u\}$, $\eta_u = \rho_{ij}$ and $A_u = A_{ij} = \{\rho_{ij} > t\}$. By the i.i.d. assumption on $\mathbf{X}_1, \cdots, \mathbf{X}_n$ and Lemma 14,

$$|P(M_n \le t) - e^{-\lambda_n}| \le b_{1,n} + b_{2,n} \tag{23}$$

where

$$\lambda_n = \frac{n(n-1)}{2} P(A_{12}) \tag{24}$$

and

$$b_{1,n} \le 2n^3 P(A_{12})^2 \text{ and } b_{2,n} \le 2n^3 P(A_{12}A_{13}).$$

By Lemma 11, $A_{12}$ and $A_{13}$ are independent events with the same probability. Thus, from (24),

$$b_{1,n} \vee b_{2,n} \le 2n^3 P(A_{12})^2 \le \frac{8n\lambda_n^2}{(n-1)^2} \le \frac{32\lambda_n^2}{n} \tag{25}$$

for all $n \ge 2$. Now we evaluate $P(A_{12})$. In fact, by Lemma 11 again,

$$P(A_{12}) = \int_t^1 g(x)\,dx \quad = \quad \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \int_t^1 (1-x^2)^{\frac{p-3}{2}}\,dx.$$

Set $m = \frac{p-3}{2} \ge -\frac{1}{2}$. We claim

$$\int_t^1 (1-x^2)^m\,dx \sim \frac{1}{2m+2}(1-t^2)^{m+1} \tag{26}$$

as $n \to \infty$. In fact, set $s = x^2$. Then $x = \sqrt{s}$ and $dx = \frac{1}{2\sqrt{s}} ds$. It follows that

$$\begin{aligned} \int_t^1 (1-x^2)^m\,dx &= \int_{t^2}^1 \frac{1}{2\sqrt{s}}(1-s)^m\,ds \\ &\sim \frac{1}{2}\int_{t^2}^1 (1-s)^m\,ds = \frac{1}{2m+2}(1-t^2)^{m+1} \end{aligned}$$

as $n \to \infty$, where the fact $\lim_{n\to\infty} t = \lim_{n\to\infty} t_n = 1$ stated in (21) is used in the second step to replace $\frac{1}{2\sqrt{s}}$ by $\frac{1}{2}$. So the claim (26) follows.

Now, we know from (24) that

$$\begin{aligned} \lambda_n \sim \frac{n^2}{2\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \int_t^1 (1-x^2)^{\frac{p-3}{2}}\,dx &\sim \frac{n^2}{2\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{(p-1)\Gamma(\frac{p-1}{2})}(1-t^2)^{(p-1)/2} \\ &= \frac{1}{4\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p+1}{2})} \Big(n^{4/(p-1)}(1-t^2)\Big)^{(p-1)/2} \end{aligned}$$

as $n \to \infty$, where (26) is used in the second step and the fact $\Gamma(x+1) = x\Gamma(x)$ is used in the last step. By (21),

$$n^{4/(p-1)}(1-t^2) = 2x + O\Big(\frac{1}{n^{4/(p-1)}}\Big)$$

as $n \to \infty$. Therefore,

$$\lambda_n \to \frac{2^{(p-5)/2}}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p+1}{2})} x^{(p-1)/2} = K_1 x^{(p-1)/2}$$

as $n \to \infty$. Finally, by (23) and (25), we know

$$\lim_{n\to\infty} P(M_n \le t) = e^{-K_1 x^{(p-1)/2}}.$$

This concludes (22). ■

**Proof of Theorem 2**. First, since $M_n = \cos\Theta_{\min}$ by (3), then use the identity $1-\cos h = 2\sin^2\frac{h}{2}$ for all $h \in \mathbb{R}$ to have

$$n^{4/(p-1)}(1-M_n) = 2n^{4/(p-1)} \sin^2 \frac{\Theta_{\min}}{2}. \tag{27}$$

By Proposition 17 and the Slusky lemma, $\sin\frac{\Theta_{\min}}{2} \to 0$ in probability as $n \to \infty$. Noticing $0 \le \Theta_{\min} \le \pi$, we then have $\Theta_{\min} \to 0$ in probability as $n \to \infty$. From (27) and the fact that $\lim_{x\to 0}\frac{\sin x}{x} = 1$ we obtain

$$\frac{n^{4/(p-1)}(1-M_n)}{\frac{1}{2}n^{4/(p-1)}\Theta_{\min}^2} \to 1$$

in probability as $n \to \infty$. By Proposition 17 and the Slusky lemma again, $\frac{1}{2}n^{4/(p-1)}\Theta_{\min}^2$ converges in distribution to $F_1(x)$ as in Proposition 17. Second, for any $x > 0$,

$$\begin{aligned} P(n^{2/(p-1)}\Theta_{\min} \le x) &= P\Big(\frac{1}{2}n^{4/(p-1)}\Theta_{\min}^2 \le \frac{x^2}{2}\Big) \\ &\to 1-\exp\{-K_1(x^2/2)^{(p-1)/2}\} = 1-\exp\{-Kx^{p-1}\} \end{aligned} \tag{28}$$

as $n \to \infty$, where

$$K = 2^{(1-p)/2}K_1 = \frac{1}{4\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p+1}{2})}. \tag{29}$$

Now we prove

$$n^{2/(p-1)}(\pi - \Theta_{\max}) \text{ converges weakly to } F(x) \text{ as } n \to \infty. \tag{30}$$

In fact, recalling the proof of the above and that of Proposition 17, we only use the following properties about $\rho_{ij}$ :

(i) $\{\rho_{ij}; 1 \le i < j \le n\}$ are pairwise independent.

(ii) $\rho_{ij}$ has density function $g(\rho)$ given in (12) for all $1 \le i < j \le n$.

(iii) For each $1 \le i < j \le n$, $\rho_{ij}$ is independent of $\{\rho_{kl}; 1 \le k < l \le n; \{k,l\} \cap \{i,j\} = \emptyset\}$.

By using Lemmas 11 and 12 and the remark between them, we see that the above three properties are equivalent to

(a) $\{\Theta_{ij}; 1 \le i < j \le n\}$ are pairwise independent.

(b) $\Theta_{ij}$ has density function $h(\theta)$ given in (13) for all $1 \le i < j \le n$.

(c) For each $1 \le i < j \le n$, $\Theta_{ij}$ is independent of $\{\Theta_{kl}; 1 \le k < l \le n; \{k,l\} \cap \{i,j\} = \emptyset\}$.

It is easy to see from (ii) of lemma 12 that the above three properties are equivalent to the corresponding (a) , (b) and (c) when "$\Theta_{ij}$" is replaced by "$\pi - \Theta_{ij}$" and "$\Theta_{kl}$" is replaced by "$\pi - \Theta_{kl}$." Also, it is key to observe that $\min\{\pi - \Theta_{ij}; 1 \le i < j \le n\} = \pi - \Theta_{\max}$. We then deduce from (28) that

$$P(n^{2/(p-1)}(\pi - \Theta_{\max}) \le x) \to 1 - \exp\{-Kx^{p-1}\} \tag{31}$$

as $n \to \infty$, where $K$ is as in (29). ■

**Proof of Theorem 3**. We will prove the following:

$$\lim_{n\to\infty} P\big(n^{2/(p-1)}\Theta_{\min} \ge x, n^{2/(p-1)}(\pi - \Theta_{\max}) \ge y\big) = e^{-K(x^{p-1}+y^{p-1})} \tag{32}$$

for any $x \ge 0$ and $y \ge 0$, where $K$ is as in (9). Note that the right hand side in (32) is identical to $P(X \ge x, Y \ge y)$, where $X$ and $Y$ are as in the statement of Theorem 3. If (32) holds, by the fact that $\Theta_{\min}, \Theta_{\max}, X, Y$ are continuous random variables and by Theorem 2 we know that $Q_n := \big((n^{2/(p-1)}\Theta_{\min}, n^{2/(p-1)}(\pi - \Theta_{\max})\big) \in \mathbb{R}^2$ for $n \ge 2$ is a tight sequence. By the standard subsequence argument, we obtain that $Q_n$ converges weakly to the distribution of $(X,Y)$ as $n \to \infty$. Applying the map $h(x,y) = x - y$ with $x, y \in \mathbb{R}$ to the sequence $\{Q_n; n \ge 2\}$ and its limit, the desired conclusion then follows from the continuous mapping theorem on the weak convergence of probability measures.

We now prove (32). Set $t_x = n^{-2/(p-1)}x$ and $t_y = \pi - n^{-2/(p-1)}y$. Without loss of generality, we assume $0 \le t_x < t_y < \infty$ for all $n \ge 2$. Then

$$\begin{aligned} & P\big(n^{2/(p-1)}\Theta_{\min} \ge x, n^{2/(p-1)}(\pi - \Theta_{\max}) \ge y\big) \\ = \; & P(t_x \le \Theta_{ij} \le t_y \text{ for all } 1 \le i < j \le n) \\ = \; & P\big(X_u = 0 \text{ for all } u \in I\big) \end{aligned} \tag{33}$$

where $I := \{(i,j);\ 1 \le i < j \le n\}$ and

$$X_u := \begin{cases} 1, & \text{if } \Theta_u \notin [t_x, t_y]; \\ 0, & \text{if } \Theta_u \in [t_x, t_y]. \end{cases}$$

For $u = (i,j) \in I$, set $B_u = \{(k,l) \in I;$ one of $k$ and $l = i$ or $j$, but $(k,l) \ne u\}$. By the i.i.d. assumption on $\mathbf{X}_1, \cdots, \mathbf{X}_n$ and Lemma 13

$$|P\big(X_u = 0 \text{ for all } u \in I\big) - e^{-\lambda_n}| \le b_{1,n} + b_{2,n} \tag{34}$$

where

$$\lambda_n = \frac{n(n-1)}{2} P(A_{12}) \text{ and } A_{12} = \big\{\Theta_{12} \notin [t_x, t_y]\big\} \tag{35}$$

and

$$b_{1,n} \le 2n^3 P(A_{12})^2 \text{ and } b_{2,n} \le 2n^3 P(A_{12}A_{13}) = 2n^3 P(A_{12})^2 \tag{36}$$

by Lemma 12. Now

$$P(A_{12}) = P(\Theta_{12} < t_x) + P(\Theta_{12} > t_y). \tag{37}$$

By Lemma 12 again,

$$\begin{aligned} P(\Theta_{12} > t_y) &= \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \int_{t_y}^{\pi} (\sin\theta)^{p-2} d\theta \\ &= \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \int_{0}^{n^{-2/(p-1)}y} (\sin\eta)^{p-2} d\eta \end{aligned} \tag{38}$$

by setting $\eta = \pi - \theta$. Now, set $v = \cos\eta$ for $\eta \in [0,\pi]$. Write $(\sin\eta)^{p-2} = -(\sin\eta)^{p-3}(\cos\eta)'$. Then the integral in (38) is equal to

$$\int_{v_y}^{1} (1-v^2)^{(p-3)/2} dv$$

where

$$v_y := \cos(n^{-2/(p-1)}y) = 1 - \frac{y^2}{2n^{4/(p-1)}} + O\Big(\frac{1}{n^{8/(p-1)}}\Big)$$

as $n \to \infty$ by the Taylor expansion. Trivially,

$$v_y^2 = 1 - \frac{y^2}{n^{4/(p-1)}} + O\Big(\frac{1}{n^{8/(p-1)}}\Big)$$

as $n \to \infty$. Thus, by (26),

$$\int_{v_y}^{1} (1-v^2)^{(p-3)/2} dv \sim \frac{1}{p-1}(1-v_y^2)^{(p-1)/2} = \frac{y^{p-1}}{(p-1)n^2}\Big(1 + O\Big(\frac{1}{n^{4/(p-1)}}\Big)\Big)$$

as $n \to \infty$. Combining all the above we conclude that

$$\begin{aligned} P(\Theta_{12} > t_y) &= \frac{\Gamma(\frac{p}{2})}{\sqrt{\pi}(p-1)\Gamma(\frac{p-1}{2})} \frac{y^{p-1}}{n^2}(1+o(1)) \\ &= \frac{\Gamma(\frac{p}{2})}{2\sqrt{\pi}\Gamma(\frac{p+1}{2})} \frac{y^{p-1}}{n^2}(1+o(1)) \end{aligned} \tag{39}$$

as $n \to \infty$. Similar to the part between (38) and (39), we have

$$\begin{aligned} P(\Theta_{12} < t_x) &= \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \int_{0}^{n^{-2/(p-1)}x} (\sin\theta)^{p-2} d\theta \\ &= \frac{\Gamma(\frac{p}{2})}{2\sqrt{\pi}\Gamma(\frac{p+1}{2})} \frac{x^{p-1}}{n^2}(1+o(1)) \end{aligned}$$

as $n \to \infty$. This joint with (39) and (37) implies that

$$P(A_{12}) = \frac{\Gamma(\frac{p}{2})}{2\sqrt{\pi}\Gamma(\frac{p+1}{2})} \frac{x^{p-1}+y^{p-1}}{n^2}(1+o(1))$$

as $n \to \infty$. Recalling (35) and (36), we obtain

$$\lim_{n\to\infty} \lambda_n = K(x^{p-1} + y^{p-1})$$

and $b_{1,n} \vee b_{2,n} = O\Big(\frac{1}{n}\Big)$ as $n \to \infty$, where $K$ is as in (9). These two assertions and (34) yield

$$\lim_{n\to\infty} P\big(X_u = 0 \text{ for all } u \in I\big) = e^{-K(x^{p-1}+y^{p-1})}.$$

Finally, this together with (33) implies (32). ■

### 6.3 Proofs of Main Results in Section 3

**Proof of Theorem 4**. Notice $(p-2)/p \to 1$ as $p \to \infty$, to prove the theorem, it is enough to show that the theorem holds if "$\mu_{n,p}$" is replaced by "$\frac{1}{\binom{n}{2}} \sum_{1\le i<j\le n} \delta_{\sqrt{p}(\frac{\pi}{2}-\Theta_{ij})}$." Thus, without loss of generality, we assume (with a bit of abuse of notation) that

$$\mu_{n,p} = \frac{1}{\binom{n}{2}} \sum_{1\le i<j\le n} \delta_{\sqrt{p}(\frac{\pi}{2}-\Theta_{ij})},\ n \ge 2,\ p \ge 2.$$

Recall $p = p_n$. Set $Y_n := \sqrt{p}(\frac{\pi}{2} - \Theta_{12})$ for $p \ge 2$. We claim that

$$Y_n \text{ converges weakly to } N(0,1) \tag{40}$$

as $n \to \infty$. Assuming this is true, taking $\varphi_n(\theta) = \sqrt{p}(\frac{\pi}{2} - \theta)$ for $\theta \in [0,\pi]$ and $\nu = N(0,1)$ in (ii) of Lemma 16, then, with probability one, $\mu_{n,p}$ converges weakly to $N(0,1)$ as $n \to \infty$.

Now we prove the claim. In fact, noticing $\Theta_{12}$ has density $h(\theta)$ in (13), it is easy to see that $Y_n$ has density function

$$\begin{aligned} h_n(y): &= \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \cdot \Big[\sin\Big(\frac{\pi}{2} - \frac{y}{\sqrt{p}}\Big)\Big]^{p-2} \cdot \Big|-\frac{1}{\sqrt{p}}\Big| \\ &= \frac{1}{\sqrt{p\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \cdot \Big(\cos\frac{y}{\sqrt{p}}\Big)^{p-2} \end{aligned} \tag{41}$$

for any $y \in \mathbb{R}$ as $n$ is sufficiently large since $\lim_{n\to\infty} p_n = \infty$. By (17),

$$\frac{1}{\sqrt{p\pi}} \frac{\Gamma(\frac{p}{2})}{\Gamma(\frac{p-1}{2})} \to \frac{1}{\sqrt{2\pi}} \tag{42}$$

as $n \to \infty$. On the other hand, by the Taylor expansion,

$$\Big(\cos\frac{y}{\sqrt{p}}\Big)^{p-2} = \Big(1 - \frac{y^2}{2p} + O\Big(\frac{1}{p^2}\Big)\Big)^{p-2} \to e^{-y^2/2}$$

as $n \to \infty$. The above together with (41) and (42) yields that

$$\lim_{n\to\infty} h_n(y) \to \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \tag{43}$$

for any $y \in \mathbb{R}$. The assertions in (41) and (42) also imply that $\sup_{y\in\mathbb{R}} |h_n(y)| \le C$ for $n$ sufficiently large, where $C$ is a constant not depending on $n$. This and (43) conclude (40). ■

**Proof of Proposition 5**. By (i) of Lemma 12,

$$P(|\Theta - \frac{\pi}{2}| \ge \varepsilon) = C_p \int_{|\theta-\frac{\pi}{2}|\ge\varepsilon} (\sin\theta)^{p-2} d\theta = C_p \int_{\varepsilon\le|t|\le\pi/2} (\cos t)^{p-2} dt$$

by making transform $t = \theta - \frac{\pi}{2}$, where $C_p := \frac{1}{\sqrt{\pi}}\Gamma(\frac{p}{2})/\Gamma(\frac{p-1}{2})$. The last term above is identical to

$$2C_p \int_{\varepsilon}^{\pi/2} (\cos t)^{p-2} dt \le \pi C_p (\cos\varepsilon)^{p-2}.$$

It is known that $\lim_{x\to+\infty} \Gamma(x+a)/(x^a\Gamma(x)) = 1$, see, for example, Dong, Jiang and Li (2012). Then $\pi C_p \le K\sqrt{p}$ for all $p \ge 2$, where $K$ is a universal constant. The desired conclusion then follows. ■

**Proof of Theorem 6**. Review the proof of Theorem 1 in Cai and Jiang (2012). Replacing $|\rho_{ij}|$, $L_n$ in (2) and Lemma 6.4 from Cai and Jiang (2012) with $\rho_{ij}$, $M_n$ in (11) and Lemma 15 here, respectively. In the places where "$n-2$" or "$n-4$" appear in the proof, change them to "$p-1$" or "$p-3$" accordingly. Keeping the same argument in the proof, we then obtain the following.

(a) $M_n \to 0$ in probability as $n \to \infty$.

(b) Let $T_n = \log(1-M_n^2)$. Then, as $n \to \infty$,

$$pT_n + 4\log n - \log\log n$$

converges weakly to an extreme value distribution with the distribution function $F(y) = 1 - e^{-Ke^{y/2}}$, $y \in \mathbb{R}$ and $K = 1/(2\sqrt{8\pi}) = 1/(4\sqrt{2\pi})$. From (11) we know

$$M_n = \max_{1\le i<j\le n} \rho_{ij} = \cos\Theta_{\min} \text{ and } \Theta_{\min} \in [0,\pi]; \tag{44}$$

$$T_n = \log(1-M_n^2) = 2\log\sin\Theta_{\min}. \tag{45}$$

Then (a) above implies that $\Theta_{\min} \to \pi/2$ in probability as $n \to \infty$, and (b) implies (ii) for $\Theta_{\min}$ in the statement of Theorem 6. Now, observe that

$$\min_{1\le i<j\le n} \{\pi - \Theta_{ij}\} = \pi - \Theta_{\max} \text{ and } \sin(\pi - \Theta_{\max}) = \sin\Theta_{\max}. \tag{46}$$

By the same argument between (30) and (31), we get $\pi - \Theta_{\max} \to \pi/2$ in probability as $n \to \infty$, that is, $\Theta_{\max} \to \pi/2$ in probability as $n \to \infty$. Notice

$$\begin{aligned} &\max_{1\le i<j\le p} \left|\Theta_{ij} - \frac{\pi}{2}\right| \\ \le\ &\left|\Theta_{\max} - \frac{\pi}{2}\right| + \left|\Theta_{\min} - \frac{\pi}{2}\right| \to 0 \end{aligned}$$

in probability as $n \to \infty$. We get (i).

Finally, by the same argument between (30) and (31) again, and by (46) we obtain

$$2p\log\sin\Theta_{\max} + 4\log n - \log\log n$$

converges weakly to $F(y) = 1 - e^{-Ke^{y/2}}$, $y \in \mathbb{R}$ and $K = 1/(4\sqrt{2\pi})$. Thus, (ii) also holds for $\Theta_{\max}$. ■

**Proof of Corollary 7**. Review the proof of Corollary 2.2 from Cai and Jiang (2012). Replacing $L_n$ and Theorem 1 there by $M_n$ and Theorem 6, we get that

$$pM_n^2 - 4\log n + \log\log n$$

converges weakly to the distribution function $\exp\{-\frac{1}{4\sqrt{2\pi}}e^{-(y+8\alpha^2)/2}\}$, $y \in \mathbb{R}$. The desired conclusion follows since $M_n = \cos\Theta_{\min}$. ■

**Proof of Theorem 8**. Review the proof of Theorem 2 in Cai and Jiang (2012). Replacing $|\rho_{ij}|$, $L_n$ in (2) and Lemma 6.4 from Cai and Jiang (2012) with $\rho_{ij}$, $M_n$ in (11) and Lemma 15, respectively. In the places where "$n-2$" and "$n-4$" appear in the proof, change them to "$p-1$" and "$p-3$" accordingly. Keeping the same argument in the proof, we then have the following conclusions.

(i) $M_n \to \sqrt{1-e^{-4\beta}}$ in probability as $n \to \infty$.

(ii) Let $T_n = \log(1-M_n^2)$. Then, as $n \to \infty$,

$$pT_n + 4\log n - \log\log n$$

converges weakly to the distribution function

$$F(y) = 1 - \exp\Big\{-K(\beta)e^{(y+8\beta)/2}\Big\},\ y \in \mathbb{R},$$

where

$$K(\beta) = \frac{1}{2}\Big(\frac{\beta}{2\pi(1-e^{-4\beta})}\Big)^{1/2} = \Big(\frac{\beta}{8\pi(1-e^{-4\beta})}\Big)^{1/2}.$$

From (44) and (45) we obtain

$$\Theta_{\min} \to \cos^{-1}\sqrt{1-e^{-4\beta}} \text{ in probability and} \tag{47}$$

$$2p\log\sin\Theta_{\min} + 4\log n - \log\log n \tag{48}$$

converges weakly to the distribution function

$$F(y) = 1 - \exp\Big\{-K(\beta)e^{(y+8\beta)/2}\Big\},\ y \in \mathbb{R}, \text{ where } K(\beta) = \Big(\frac{\beta}{8\pi(1-e^{-4\beta})}\Big)^{1/2} \tag{49}$$

as $n \to \infty$. Now, reviewing (46) and the argument between (30) and (31), by (47) and (48), we conclude that $\Theta_{\max} \to \pi - \cos^{-1}\sqrt{1-e^{-4\beta}}$ in probability and $2p\log\sin\Theta_{\max} + 4\log n - \log\log n$ converges weakly to the distribution function $F(y)$ as in (49). The proof is completed. ■

**Proof of Theorem 9**. Review the proof of Theorem 3 in Cai and Jiang (2012). Replacing $|\rho_{ij}|$, $L_n$ in (2) and Lemma 6.4 from Cai and Jiang (2012) with $\rho_{ij}$, $M_n$ in (11) and Lemma 15, respectively. In the places where "$n-2$" or "$n-4$" appear in the proof, change them to "$p-1$" or "$p-3$" accordingly. Keeping the same argument in the proof, we get the following results.

i) $M_n \to 1$ in probability as $n \to \infty$.

ii) As $n \to \infty$,

$$pM_n + \frac{4p}{p-1}\log n - \log p$$

converges weakly to the distribution function $F(y) = 1 - e^{-Ke^{y/2}}$, $y \in \mathbb{R}$ with $K = 1/(2\sqrt{2\pi})$. Combining i), ii), (44) and (45), we see that, as $n \to \infty$,

$$\begin{aligned}
&\Theta_{\min} \to 0 \text{ in probability;} \\
&2p\log\sin\Theta_{\min} + \frac{4p}{p-1}\log n - \log p \text{ converges weakly to}
\end{aligned}$$

$F(y) = 1 - e^{-Ke^{y/2}}$, $y \in \mathbb{R}$ with $K = 1/(2\sqrt{2\pi})$. Finally, combining the above two convergence results with (46) and the argument between (30) and (31), we have

$$\begin{aligned}
&\Theta_{\max} \to \pi \text{ in probability;} \\
&2p\log\sin\Theta_{\max} + \frac{4p}{p-1}\log n - \log p \text{ converges weakly to}
\end{aligned}$$

$F(y) = 1 - e^{-Ke^{y/2}}$, $y \in \mathbb{R}$ with $K = 1/(2\sqrt{2\pi})$. ■

## Acknowledgments

The research of Tony Cai was supported in part by NSF FRG Grant DMS-0854973, NSF Grant DMS-1209166, and NIH Grant R01 CA127334. The research of Jianqing Fan was supported in part by NSF grant DMS-1206464 and NIH grants NIH R01-GM072611 and R01GM100474. The research of Tiefeng Jiang was supported in part by NSF FRG Grant DMS-0449365 and NSF Grant DMS-1209166.

## References

Lars V. Ahlfors. *Complex Analysis*. McGraw-Hill, New York, 1979.

Vangalur S. Alagar. The distribution of the distance between random points. *Journal of Applied Probability*, 13(3):558–566, 1976.

Diego Armentano, Carlos Beltrán, and Michael Shub. Minimizing the discrete logarithmic energy on the sphere: The role of random polynomials. *Transactions of the American Mathematical Society*, 363(6):2955–2965, 2011.

Richard Arratia, Larry Goldstein, and Louis Gordon. Two moments suffice for poisson approximations: the chen-stein method. *The Annals of Probability*, 17(1):9–25, 1989.

Peter J. Bickel, Yaacov Ritov, and Alexandre B. Tsybakov. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, 37(4):1705–1732, 2009.

Tony T. Cai and Tiefeng Jiang. Limiting laws of coherence of random matrices with applications to testing covariance structure and construction of compressed sensing matrices. *The Annals of Statistics*, 39(3):1496–1525, 2011.

Tony T. Cai and Tiefeng Jiang. Phase transition in limiting distributions of coherence of high-dimensional random matrices. *Journal of Multivariate Analysis*, 107:24–39, 2012.

Ery Arias Castro, Sébastien Bubeck, and Gábor Lugosi. Detecting positive correlations in a multivariate sample. *arXiv preprint arXiv:1202.5536*, 2012.

Luc Devroye, András György, Gábor Lugosi, and Frederic Udina. High-dimensional random geometric graphs and their clique number. *Electronic Journal of Probability*, 16:2481–2508, 2011.

Persi Diaconis and David Freedman. Asymptotics of graphical projection pursuit. *The Annals of Statistics*, 12(3):793–815, 1984.

Zhishan Dong, Tiefeng Jiang, and Danning Li. Circular law and arc law for truncation of random unitary matrix. *Journal of Mathematical Physics*, 53:013301–14, 2012.

David L Donoho and Xiaoming Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001.

Kai-Tai Fang, Samuel Kotz, and Kai Wang Ng. *Symmetric Multivariate and Related Distributions*. Chapman and Hall Ltd. London., 1990.

Theodore W Gamelin. *Complex Analysis*. Springer New York, 2001.

Ricardo García-Pelayo. Distribution of distance in the spheroid. *Journal of Physics A: Mathematical and General*, 38(16):3475–3482, 2005.

Eric Gautier and Alexandre B Tsybakov. High-dimensional instrumental variables regression and confidence sets. *arXiv preprint arXiv:1105.2454*, 2011.

Peter Hall, JS Marron, and Amnon Neeman. Geometric representation of high dimension, low sample size data. *Journal of the Royal Statistical Society: Series B*, 67(3):427–444, 2005.

John M Hammersley. The distribution of distance in a hypersphere. *The Annals of Mathematical Statistics*, 21(3):447–452, 1950.

Tiefeng Jiang. The asymptotic distributions of the largest entries of sample correlation matrices. *The Annals of Applied Probability*, 14(2):865–880, 2004.

Ali Katanforoush and Mehrdad Shahshahani. Distributing points on the sphere, i. *Experimental Mathematics*, 12(2):199–209, 2003.

Wilfrid S Kendall and Ilya Molchanov. *New Perspectives in Stochastic Geometry*. Oxford University Press, 2010.

Arno Kuijlaars and E Saff. Asymptotics for minimal discrete energy on the sphere. *Transactions of the American Mathematical Society*, 350(2):523–538, 1998.

Michel Ledoux. *The Concentration of Measure Phenomenon*. American Mathematical Society, 2001.

Deli Li and Andrew Rosalsky. Some strong limit theorems for the largest entries of sample correlation matrices. *The Annals of Applied Probability*, 16(1):423–447, 2006.

Deli Li, Weidong Liu, and Andrew Rosalsky. Necessary and sufficient conditions for the asymptotic distribution of the largest entry of a sample correlation matrix. *Probability Theory and Related Fields*, 148(1-2):5–35, 2010.

Deli Li, Yongcheng Qi, and Andrew Rosalsky. On jiang's asymptotic distribution of the largest entry of a sample correlation matrix. *Journal of Multivariate Analysis*, 111:256–270, 2012.

Weidong Liu, Zhengyan Lin, and Qiman Shao. The asymptotic distribution and berry–esseen bound of a new test for independence in high dimension with an application to stochastic optimization. *The Annals of Applied Probability*, 18(6):2337–2366, 2008.

Reginald Douglas Lord. The distribution of distance in a hypersphere. *The Annals of Mathematical Statistics*, 25(4):794–798, 1954.

Robb J Muirhead. *Aspects of Multivariate Statistical Theory*. Wiley. New York., 1982.

Mathew Penrose. *Random Geometric Graphs*. Oxford University Press. Oxford., 2003.

Sidney I Resnick. *Extreme Values, Regular Variation, and Point Processes*. Springer-Verlag, New York., 2007.

Dan Shen, Haipeng Shen, Hongtu Zhu, and JS Marron. Surprising asymptotic conical structure in critical sample eigen-directions. *arXiv preprint arXiv:1303.6171*, 2013.

Steve Smale. Mathematical problems for the next century. *Mathematics: Frontiers and Perspectives (Ed. V. Arnold, M. Atiyah, P. Lax, and B. Mazur)*, pages 271–294, 2000.

Dietrich Stoyan, Wilfrid S Kendall, Joseph Mecke, and DG Kendall. *Stochastic Geometry and its Applications* (2nd ed.), volume 2. Wiley. Chichester W. Sussex and New York., 1995.

Shu-Ju Tu and Ephraim Fischbach. Random distance distribution for spherical objects: general theory and applications to physics. *Journal of Physics A: Mathematical and General*, 35(31): 6557–6570, 2002.

Mike Williams. On a probability distribution function arising in stochastic neutron transport theory. *Journal of Physics A: Mathematical and General*, 34(22):4653–4662, 2001.

Wang Zhou. Asymptotic distribution of the largest off-diagonal entry of correlation matrices. *Transactions of the American Mathematical Society*, 359(11):5345–5363, 2007.

Journal of Machine Learning Research 14 (2013) 1865-1889 Submitted 12/11; Revised 11/12; Published 7/13

# Cluster Analysis: Unsupervised Learning via Supervised Learning with a Non-convex Penalty

**Wei Pan** WEIP@BIOSTAT.UMN.EDU
*Division of Biostatistics*
*University of Minnesota*
*Minneapolis, MN 55455, USA*

**Xiaotong Shen** XSHEN@UMN.EDU
*School of Statistics*
*University of Minnesota*
*Minneapolis, MN 55455, USA*

**Binghui Liu** LIUBH024@GMAIL.COM
*Division of Biostatistics and School of Statistics*
*University of Minnesota*
*Minneapolis, MN 55455, USA*

**Editor:** Francis Bach

## Abstract

Clustering analysis is widely used in many fields. Traditionally clustering is regarded as unsupervised learning for its lack of a class label or a quantitative response variable, which in contrast is present in supervised learning such as classification and regression. Here we formulate clustering as penalized regression with grouping pursuit. In addition to the novel use of a non-convex group penalty and its associated unique operating characteristics in the proposed clustering method, a main advantage of this formulation is its allowing borrowing some well established results in classification and regression, such as model selection criteria to select the number of clusters, a difficult problem in clustering analysis. In particular, we propose using the generalized cross-validation (GCV) based on generalized degrees of freedom (GDF) to select the number of clusters. We use a few simple numerical examples to compare our proposed method with some existing approaches, demonstrating our method's promising performance.

**Keywords:** generalized degrees of freedom, grouping, K-means clustering, Lasso, penalized regression, truncated Lasso penalty (TLP)

## 1. Introduction

Clustering analysis has been widely used in many fields, for example, for microarray gene expression data (Thalamuthu et al., 2006), mainly for exploratory data analysis or class novelty discovery; see Xu and Wunsch (2005) for an extensive review on the methods and applications. In the absence of a class label, clustering analysis is also called unsupervised learning, as opposed to supervised learning that includes classification and regression. Accordingly, approaches to clustering analysis are typically quite different from supervised learning.

In this paper we adopt a novel framework for clustering analysis by viewing it as a regression problem (Pelckmans et al., 2005; Hocking et al., 2011; Lindsten et al., 2011). We explicitly parametrize each multivariate observation, say $x_i$, with its own centroid, say $\mu_i$. Clustering analysis

©2013 Wei Pan, Xiaotong Shen and Binghui Liu.

is formulated to identify a small subset of distinct values of these $\mu_i$. Since we have observation-specific and over-parameterized $\mu_i$'s, a key question is how to estimate these parameters. Taking advantage of the recent advance in penalized regression (Tibshirani et al., 2005; Shen and Huang, 2010), we propose a novel non-convex penalty for grouping pursuit that data-adaptively encourages the equality among some unknown subsets of parameter estimates, thus effectively realizing clustering. We call our proposed method as *penalized regression-based clustering* (**PRclust**).

An advantage of regarding clustering as a regression problem is its unification with regression, which in turn provides the opportunity to apply or modify many established results and techniques, such as model selection criteria, in regression to clustering. In particular, a notoriously difficult model selection problem in clustering analysis is to determine the number of clusters; already numerous methods exist with new ones constantly emerging (Tibshirani et al., 2001; Sugar and James, 2003; Wang, 2010). Here we propose the use of generalized cross-validation (GCV) (Golub et al., 1979) that has been widely used for model selection in regression for its solid theoretical foundation, computational efficiency and good empirical performance. However, GCV requires estimating the degrees of freedom (df) or effective number of parameters. In clustering analysis, due to the data-adaptive nature of model searches in finding clusters, it is unclear what is, or how to estimate df. Here we propose using a general method called generalized degrees of freedom (GDF) that was specifically developed in the context of classification and regression to take into account the complex effects of data-adaptive modeling (Ye, 1998; Shen and Ye, 2002). To our knowledge, GDF is mainly studied in the context of regression. Again by formulating clustering as regression, we can adapt the use of GDF to our current context. Although not the main point of this paper, we will show that GDF-based GCV performed well in our numerical examples.

In spite of many advantages of formulating clustering analysis as a penalized regression problem, there are some challenges in its implementation. In particular, with a desired non-smooth and non-convex penalty function, many existing algorithms for penalized regression are not suitable. We develop a novel and efficient computational algorithm that combines the difference of convex (DC) programming (An and Tao, 1997) and a coordinate-wise descent algorithm (Friedman et al., 2007; Wu and Lange, 2008).

Due to some conceptual similarity between our proposed PRclust and the popular K-means clustering, we use the K-means as a benchmark to assess the performance of PRclust. In particular, we show that in some complex situations, for example, in the presence of non-convex clusters, in which the K-means is not suitable, PRclust might perform much better. Hence, complementary to the K-means, PRclust is a potentially useful clustering tool. In addition, we consider a related procedure based on hard thresholding pair-wise distances between observations, called *HTclust*. Although simpler, due to the lack of shrinkage estimation, HTclust may not perform as well as PRclust. Albeit not the focus here, we also propose GDF-based GCV as a general model selection criterion to determine the number of clusters in the above clustering approaches; a comparison with several existing methods demonstrates the promising performance of GCV.

## 2. Methods

We first present our new method, including its computational algorithm, before comparing it with two related methods. Then we propose a GCV-based method to select the number of clusters.

### 2.1 New Method: Clustering via Penalized Regression

Given data $X = (x_1', ..., x_n')'$ with $x_i = (x_{i1}, x_{i2}, ..., x_{ip})'$, we would like to conduct a cluster analysis; that is, we would like to identify group-memberships of the observations such that the within-group similarity and between-group dissimilarity are as strong as possible. There are different ways of defining a similarity/dissimilarity between two observations, leading to various clustering approaches. Here we consider the situation with continuous attributes $x_{ik}$'s, and define the dissimilarity based on some distance metric, as to be elaborated later. We assume that each data point $x_i$ has its own centroid $\mu_i = (\mu_{i1}, \mu_{i2}, ..., \mu_{ip})'$, which can be its mean or median (or other measure), depending on the application. Our goal is to estimate $\mu_i$'s while acknowledging the possibility that many $\mu_i$'s would be equal if their corresponding $x_i$'s are from the same cluster. Hence, we would like to adopt a fused-Lasso-type or fusion penalty (Tibshirani et al., 2005) to encourage the equality of the centroids. In general, we estimate the parameters $\mu = (\mu_1', ..., \mu_n')'$ through minimizing an objective function

$$\hat{\mu} = \arg\min_{\mu} \frac{1}{2} \sum_{i=1}^{n} L(x_i - \mu_i) + \lambda \sum_{i<j} h(\mu_i - \mu_j),$$

where $L()$ is a loss function, for example, the squared error, $h()$ is a *grouping* or *fusion* penalty, for example, the $L_1$-norm or Lasso penalty (Tibshirani, 1996), and $\lambda$ is a tuning parameter to be selected. Specifically, with a squared error and Lasso penalty, our objective function is

$$\frac{1}{2} \sum_{i=1}^{n} ||x_i - \mu_i||_2^2 + \lambda \sum_{i<j} ||\mu_i - \mu_j||_1,$$

where $||.||_q$ is the $L_q$-norm. The main idea is that, for the purpose of clustering, we would like to strike a balance between minimizing the distance between the observations and their centroids and reducing the number of centroids via grouping some close centroids together. As pointed out by a reviewer, the general idea with a convex $L_q$-norm as the fusion penalty has appeared in the literature (Pelckmans et al., 2005; Hocking et al., 2011; Lindsten et al., 2011); here we propose a novel non-convex penalty.

Since it is well known that the Lasso penalty leads to biased parameter estimates (Fan and Li, 2001; Shen et al., 2012), it is more desirable to consider some non-convex penalties; here we propose a new form of the truncated Lasso penalty (TLP) (Shen et al., 2012). For a scalar parameter $\alpha$ and a given tuning parameter $\tau$, TLP is defined as

$$\text{TLP}(\alpha; \tau) = \min(|\alpha|, \tau),$$

which is the $L_1$-norm (i.e., Lasso) penalty for a small $\alpha \leq \tau$, but imposes no further penalty for a large $\alpha > \tau$. Importantly, $\text{TLP}(\alpha; \tau)/\tau$ tends to the $L_0$-norm of $\alpha$, $L_0(\alpha) = I(\alpha \neq 0)$, as $\tau \to 0^+$.

If two observations, $x_i$ and $x_j$, come from the same cluster, we would have $\mu_i = \mu_j$; that is, all the components of $\mu_i$ are equal to that of $\mu_j$. Hence, to more effectively realize $\mu_i = \mu_j$, we use a group penalty that encourages simultaneous equality between all the components of $\mu_i$ and $\mu_j$ (Yuan and Lin, 2006). Again, to alleviate the bias of the usual convex $L_2$-norm (or more generally, $L_q$-norm for $q > 1$) group penalty, we propose a novel and non-convex group penalty based on TLP, called group TLP or simply gTLP, defined as

$$\text{gTLP}(\mu_i - \mu_j; \tau) = \text{TLP}(||\mu_i - \mu_j||_2; \tau).$$

As to be shown, the group TLP performs much better than the Lasso (and other $L_q$-norms). Note that the group TLP has not been used before.

In this paper, we consider the use of the squared error exclusively, though other loss functions can be used as discussed later. Depending on the use of the penalty, we have two ways to estimate $\mu_i$'s:

$$\hat{\mu} = \arg\min_{\mu} \frac{1}{2}\sum_{i=1}^{n} ||x_i - \mu_i||_2^2 + \lambda \sum_{i<j} ||\mu_i - \mu_j||_1,$$
$$\hat{\mu} = \arg\min_{\mu} \frac{1}{2}\sum_{i=1}^{n} ||x_i - \mu_i||_2^2 + \lambda \sum_{i<j} \mathrm{TLP}(||\mu_i - \mu_j||_2; \tau).$$

Once we have $\hat{\mu}_i$, then the observations with an equal $\hat{\mu}_i$ are assigned to the same cluster.

### 2.2 Computing

The above grouping penalties are not separable in $\mu_i$'s in the sense that they cannot be written as a sum of the terms, each of which is a function of a single $\mu_i$ only. With the above non-separable penalties, the efficient coordinate-wise algorithm may not converge to a stationary point (Friedman et al., 2007; Wu and Lange, 2008). To develop an efficient coordinate-wise algorithm, we reparametrize by introducing some new parameters and then apply the quadratic penalty method (Nocedal and Wright, 2000). Specifically, we define $\theta_{ij} = \mu_i - \mu_j$ for $1 \le i < j \le n$, and then modify the new objective function accordingly as:

$$\begin{aligned} S_L(\mu,\theta) &= \frac{1}{2}\sum_{i=1}^{n} ||x_i - \mu_i||_2^2 + \frac{\lambda_1}{2}\sum_{i<j} ||\mu_i - \mu_j - \theta_{ij}||_2^2 + \lambda_2 \sum_{i<j} ||\theta_{ij}||_1, \\ S(\mu,\theta) &= \frac{1}{2}\sum_{i=1}^{n} ||x_i - \mu_i||_2^2 + \frac{\lambda_1}{2}\sum_{i<j} ||\mu_i - \mu_j - \theta_{ij}||_2^2 + \\ &\quad \lambda_2 \sum_{i<j} \mathrm{TLP}(||\theta_{ij}||_2; \tau). \end{aligned}$$

For $S_L(\mu,\theta)$, the first two terms are quadratic (and thus differentiable and convex) while the third is non-smooth but separable and convex, so the coordinate-wise descent algorithm can be applied and will converge to a global minimum (Tseng, 2001); its updates at iteration $m+1$ are

$$\begin{aligned} \hat{\mu}_i^{(m+1)} &= \frac{x_i + \lambda_1 \sum_{j>i}(\hat{\mu}_j^{(m)} + \hat{\theta}_{ij}^{(m)}) + \lambda_1 \sum_{j<i}(\hat{\mu}_j^{(m+1)} - \hat{\theta}_{ji}^{(m)})}{1 + \lambda_1 (n-1)}, \\ \hat{\theta}_{ij}^{(m+1)} &= \mathrm{ST}(\hat{\mu}_i^{(m+1)} - \hat{\mu}_j^{(m+1)}, \lambda_2/\lambda_1), \end{aligned} \tag{1}$$

where $\mathrm{ST}(\alpha,\lambda) = \mathrm{sign}(\alpha)(|\alpha| - \lambda)_+$ is the soft-thresholding rule, and $(a)_+$ takes the positive part of $a$: it equals to $a$ if $a > 0$, and equals to 0 otherwise. By default any scalar operation on a vector is element-wise.

For $S(\mu,\theta)$, the updating formula for $\mu_i$ remains the same as in (1). On the other hand, to deal with the non-convex TLP on $\theta_{ij}$'s, we apply the difference of convex programming technique. We

decompose $S(\mu,\theta)$ into a difference of two convex functions $S_1(\mu,\theta) - S_2(\theta)$:

$$S_1(\mu,\theta) = \frac{1}{2}\sum_{i=1}^{n}||x_i-\mu_i||_2^2 + \frac{\lambda_1}{2}\sum_{i<j}||\mu_i-\mu_j-\theta_{ij}||_2^2 + \lambda_2\sum_{i<j}||\theta_{ij}||_2,$$
$$S_2(\theta) = \lambda_2\sum_{i<j}(||\theta_{ij}||_2-\tau)_+.$$

We then construct a sequence of upper approximations iteratively by replacing $S_2(\theta)$ at iteration $m+1$ by its piecewise affine minorization

$$S_2^{(m)}(\theta) = S_2(\hat{\theta}^{(m)}) + \lambda_2\sum_{i<j}(||\theta_{ij}||_2 - ||\hat{\theta}_{ij}^{(m)}||_2)I(||\hat{\theta}_{ij}^{(m)}||_2 \geq \tau)$$

at the current estimate $\hat{\theta}^{(m)}$ from iteration $m$, leading to an upper convex approximating function at iteration $m+1$:

$$\begin{aligned} S^{(m+1)}(\mu,\theta) &= \frac{1}{2}\sum_{i=1}^{n}||x_i-\mu_i||_2^2 + \frac{\lambda_1}{2}\sum_{i<j}||\mu_i-\mu_j-\theta_{ij}||_2^2 + \\ &\quad \lambda_2\sum_{i<j}||\theta_{ij}||_2 I(||\hat{\theta}_{ij}^{(m)}||_2<\tau) + \lambda_2\tau\sum_{i<j}I(||\hat{\theta}_{ij}^{(m)}||_2\geq\tau). \end{aligned} \tag{2}$$

Applying the (block) coordinate-wise algorithm for the group Lasso (Yuan and Lin, 2006), we have

$$\hat{\theta}_{ij}^{(m+1)} = \begin{cases} \hat{\mu}_i^{(m+1)} - \hat{\mu}_j^{(m+1)}, & \text{if } ||\hat{\theta}_{ij}^{(m)}||_2 \geq \tau; \\ \left(||\hat{\mu}_i^{(m+1)} - \hat{\mu}_j^{(m+1)}||_2 - \frac{\lambda_2}{\lambda_1}\right)_+ \frac{\hat{\mu}_i^{(m+1)} - \hat{\mu}_j^{(m+1)}}{||\hat{\mu}_i^{(m+1)} - \hat{\mu}_j^{(m+1)}||_2}, & \text{otherwise.} \end{cases} \tag{3}$$

We summarize below our DC algorithm as **Algorithm 1**:
STEP 1. (Initialization) Compute an initial estimate $(\hat{\mu}^{(0)},\hat{\theta}^{(0)})$.
STEP 2. (Iteration) At iteration $m+1$, compute $(\hat{\mu}^{(m+1)},\hat{\theta}^{(m+1)})$ that minimizes (2).
STEP 3. (Stopping rule) Terminate if $S(\hat{\mu}^{(m+1)},\hat{\theta}^{(m+1)}) - S(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) \geq 0$; otherwise go to Step 2 with $m \leftarrow m+1$.

We have the following convergence result; its proof is given in an appendix.

**Theorem 1** *In Algorithm 1, $S(\hat{\mu}^{(m)},\hat{\theta}^{(m)})$ decreases strictly in m until it terminates in finite steps; that is, there exists an $m^\star < \infty$ with*

$$S(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) = S(\hat{\mu}^{(m^\star)},\hat{\theta}^{(m^\star)}) \qquad \textit{for } m \geq m^\star.$$

*Furthermore, $(\hat{\mu}^{(m^\star)},\hat{\theta}^{(m^\star)})$ is a local minimizer of $S(\mu,\theta)$.*

In implementing the Step 2 of Algorithm 1, we can repeatedly apply the coordinate-wise updates (1) and (3) to minimize (2). Since the objective function (2) is a sum of a differentiable and convex function and a convex penalty in $\mu$ and $\theta$ (while $\hat{\theta}^{(m)}$ is a known vector), the coordinate-wise descent algorithm will converge to its minimizer (Tseng, 2001). By Theorem 1, we know that this implementation of Algorithm 1 will converge to a local minimum of $S(\mu,\theta)$. In practice, especially in earlier iterations, one may not want to run the coordinate-wise updates fully until convergence in Step 2 to save computing time.

Note that, due to the use of the quadratic/ridge penalty on $\mu_i - \mu_j - \theta_{ij}$, no matter how large $\lambda_1$ is used, we cannot obtain exactly $\mu_i - \mu_j - \theta_{ij} = 0$, though their difference tends to 0 as $\lambda_1 \to \infty$; on the other hand, the ridge penalty is smooth, and thus facilitates the applicability of the coordinate-wise descent algorithm. To enforce the constraint $\mu_i - \mu_j = \theta_{ij}$ approximately, it is desirable to use a large $\lambda_1$; however, by (1), we see that a large $\lambda_1$ effectively reduces the weight of observation $x_i$'s contributing to estimating $\mu_i$. We fix $\lambda_1 = 1$ throughout, leaving it to section 4 to discuss an alternative algorithm allowing $\lambda_1 \to \infty$.

Due to the use of Lasso or TLP on $\theta_{ij}$'s, we can obtain exactly $\hat{\theta}_{ij} = 0$ for a large $\lambda_2$. We form clusters based on $\hat{\theta}_{ij}$'s: for any two observations $x_i$ and $x_j$, if $\hat{\theta}_{ij} = 0$, they are declared to be in the same cluster. We construct a graph $G$ based on an adjacency matrix $A = (a_{ij})$ with elements $a_{ij} = I(\hat{\theta}_{ij} = 0)$; finding clusters is equivalent to finding connected subcomponents of $G$. It is possible that other more sophisticated graph-based methods can be used to identify clusters, which we leave as a future topic. By default, PRclust is based on the TLP, not Lasso, unless specified otherwise.

### 2.3 Comparison with Two Related Methods

Our proposed method is closely related to the K-means method, which can be formulated as finding the centroids, say $\mu_1$, ..., $\mu_K$, for $K$ clusters, where $K \geq 1$ is a tuning parameter. To find the centroids $\mu = (\mu_1', ..., \mu_K')'$,

$$\hat{\mu} = \arg\min_{\mu} \sum_{i=1}^{n} ||x_i - \mu_{c(i)}||_2^2,$$

where $c(i)$ maps observation $i$ to cluster $c(i) \in \{1, 2, ..., K\}$, one of the $K$ candidate clusters. A typical K-means algorithm starts with some initial estimate of $\mu$, assigns each observation to its nearest centroid/cluster, recalculates each centroid, then repeats the above process until convergence. Depending on the initial estimates, the K-means may converge only to a local minimum, hence multiple starts are often used.

It is clear that both the PRclust and K-means aim to identify centroids by minimizing the total $L_2$ distance between observations and their corresponding centroids, but they approach and formulate the problem differently. PRclust over-parametrizes the centroid for each observation, then via shrinking parameters by grouping pursuit, finds a fewer number of distinct centroids; the final result depends on the specified tuning parameters $\lambda_2$ and $\tau$. In contrast, by specifying $K$, the number of clusters as the only tuning parameter, the K-means starts with some $K$ initial centroids, then assigns each observation to a cluster before updating the centroid estimates. Hence, PRclust differs from the K-means in that PRclust does not explicitly assign an observation to any cluster; clustering is implicitly done after the convergence of the algorithm.

A simple alternative to PRclust is to apply the hard-thresholding rule to pair-wise distances between observations. Suppose that $D = (d_{ij})$ is a pair-wise distance matrix with $d_{ij} = ||x_i - x_j||_2$. For any threshold $d$, we can define an adjacency matrix $A = (a_{ij})$ with $a_{ij} = I(d_{ij} < d)$; as in PRclust, we can define any connected subcomponent based on $A$ as a cluster, resulting in a clustering method called **HTclust**, which is named as a"connected components" algorithm in Ng et al. (2002). As a comparison, PRclust defines its adjacency matrix as $a_{ij} = I(\hat{\theta}_{ij} = 0) = I(||\hat{\mu}_i - \hat{\mu}_j||_2 < d_{0,ij})$ for some small and possibly $(i, j)$-dependent threshold $d_{0,ij} > 0$. HTclust is related to agglomerative hierarchical clustering: the latter also starts with each observation being its own cluster, then sequentially merges two nearest clusters until only one cluster left. Indeed, as shown in an appendix,

HTclust is equivalent to the single-linkage hierarchical clustering, in which the distance between two clusters is defined as the shortest distance between any two observations, one from each of the two clusters. An apparent difference between PRclust and HTclust (and hierarchical clustering) is the lack of shrinkage in parameter estimation in the latter, in contrast to that in the former as shown in (1). In general PRclust behaves differently from HTclust, as to be shown in a few examples.

### 2.4 Selecting Tuning Parameters or Number of Clusters

A bonus with the regression approach to clustering is the potential application of many existing model selection methods for regression or supervised learning to clustering. Here we propose using generalized cross-validation (GCV) that has been used extensively, for example, in selecting the tuning parameter in ridge regression (Golub et al., 1979). GCV can be regarded as an approximation to leave-one-out cross-validation (CV). Hence, GCV provides an approximately unbiased estimate of the prediction error. In our notation,

$$\text{GCV(df)} = \frac{\text{RSS}}{(np - \text{df})^2} = \frac{\sum_{i=1}^{n}\sum_{k=1}^{p}(x_{ik} - \hat{\mu}_{ik})^2}{(np - \text{df})^2},$$

where df is the degrees of freedom used in estimating $\mu_i$'s. For our problem, a naive treatment is to take df = $Kp$, the number of unknown parameters in $\mu_i$'s, which however does not take into account the data-adaptive nature in estimating $\mu_i$'s in clustering analysis. As to be shown, the naive estimate of df, NDF=$Kp$, is in general severely under-biased. A better way is to use the generalized degrees of freedom (GDF) (Ye, 1998). We define GDF as

$$\text{GDF} = \sum_{i=1}^{n}\sum_{k=1}^{p}\frac{1}{\sigma^2}\text{cov}(\hat{\mu}_{ik}(X), x_{ik} - \mu_{ik}) = \sum_{i=1}^{n}\sum_{k=1}^{p}\lim_{\delta\to 0} E_\mu\left[\frac{\hat{\mu}_{ik}(X + \delta e_{ik}) - \hat{\mu}_{ik}(X)}{\delta}\right],$$

where we write $\hat{\mu}_{ik} = \hat{\mu}_{ik}(X)$ to emphasize that the estimate $\hat{\mu}_{ik}$ depends on the data $X$ being used, and $e_{ik}$ is a vector of length $np$ with all elements 0 except a 1 in position $ik$. Accordingly, we can use Monte Carlo simulations to estimate GDF in the following way:

Step 1. For $b = 1, ..., B$, repeat Steps 2-3.

Step 2. Generate $\Delta_b = (\delta_{b,1}, ..., \delta_{b,np})$ with $\delta_{b,i}$ iid $N(0, v)$.

Step 3. Conduct a cluster analysis (in the same way as for the original data $X$) with data $X + \Delta_b$ to yield an estimate $\hat{\mu}(X + \Delta_b)$.

Step 4. For fixed $i$ and $k$, regress $\hat{\mu}_{ik}(X + \Delta_b)$ on $\delta_{b,ik}$ with $b = 1, ...B$; denote the slope estimate as $\hat{h}_{ik}$.

Step 5. Repeat Step 4 for each $i$ and $k$. Then an GDF estimate is GDF = $\sum_{i=1}^{n}\sum_{k=1}^{p}\hat{h}_{ik}$.

We used $B = 100$ in Step 1 throughout. In Step 2, the perturbation size (i.e., standard deviation, SD) $v$ is chosen to be small, typically with $v \in [0.5\sigma, \sigma]$, where a common variance $\sigma^2 = \text{var}(x_{ik})$ is assumed for all attributes. As discussed in Ye (1998) and Shen and Ye (2002), often the GDF estimate is not too sensitive to the choice of $v$. In Step 3, we apply the same clustering algorithm (e.g., PRclust or HTclust) with any fixed tuning parameter values as applied to the original data $X$.

We try with various tuning parameter values, obtaining their corresponding GDFs and thus GCV statistics, then choose the set of the tuning parameters with the minimum GCV statistic.

The above method can be equally applied to the K-means method to select the number of clusters: we just need to apply the K-means with a fixed number of clusters, say $K$, in Step 3, then use the cluster centroid of observation $x_i$ as its estimated mean $\hat{\mu}_i$; other steps remain the same. Again, we try various values of $K$, and choose $\hat{K} = K$ that minimizes the corresponding GCV(GDF) statistic.

As a comparison, we also apply the Jump statistic to select the number of clusters for the K-means (Sugar and James, 2003). For $K$ clusters, a distortion (or average within-cluster sum of squares) is defined to be

$$W_K = \sum_{i=1}^{n} \sum_{k=1}^{p} (x_{ik} - \hat{\mu}_{ik})^2/(np),$$

and the Jump statistic is defined as

$$J_K = 1/W_K^{p/2} - 1/W_{K-1}^{p/2},$$

with $1/W_0^{p/2} = 0$. We choose $\hat{K} = \arg\min_K J_K$.

Wang (2010) proposed a consistent estimator for the number of clusters based on clustering stability. It is based on an intuitive idea: with the correct number of clusters, the clustering results should be most stable. The method requires the use of three subsets of data: two are used to build two predictive models for the same clustering algorithm with the same number of clusters, and then the third is used to estimate the clustering stability by comparing the predictive results of the third subset when applied to the two built predictive models. For a given data set, cross-validation is used to repeatedly splitting the data into three (almost equally sized) subsets. Wang (2010) proposed two CV schemes, called CV with voting and CV with averaging. We will simply call the two methods as CV1 and CV2.

## 3. Numerical Examples

Now we use both simulated data and real data to evaluate the performance of our method and compare it with several other methods.

### 3.1 Simulation Set-ups

We considered five simulation set-ups, covering a variety of scenarios, as described below.

Case I: two convex clusters in two dimensions (Figure 1a). We consider two somewhat overlapping clusters with the same spherical shape, which is ideal for the K-means. Specifically, we have $n = 100$ observations, 50 from a bivariate Normal distribution $N((0,0)', 0.33I)$ while the other 50 from $N((1,1)', 0.33I)$.

Case II: two non-convex clusters in two dimensions (Figure 1b). In contrast to the previous case favoring the K-means, the second simulation set-up was the opposite. There were 2 clusters as two nested circles (distorted with some added noises), each with 100 observations (see the upper-left panel in Figure 3). Specifically, for cluster 1, we had $x_{i1} = -1 + 2(i-1)/99$, $x_{i2} = s_i\sqrt{1 - x_{i1}^2} + \varepsilon_i$, $s_i = -1$ or 1 with an equal probability, $\varepsilon_i$ randomly drawn from $U(-0.1, 0.1)$, for $i = 1, ..., 100$; for cluster 2, similarly we had $x_{i1} = -2 + 4(i-101)/99$, $x_{i2} = s_i\sqrt{4 - x_{i1}^2} + \varepsilon_i$ for $i = 101, ..., 200$. This

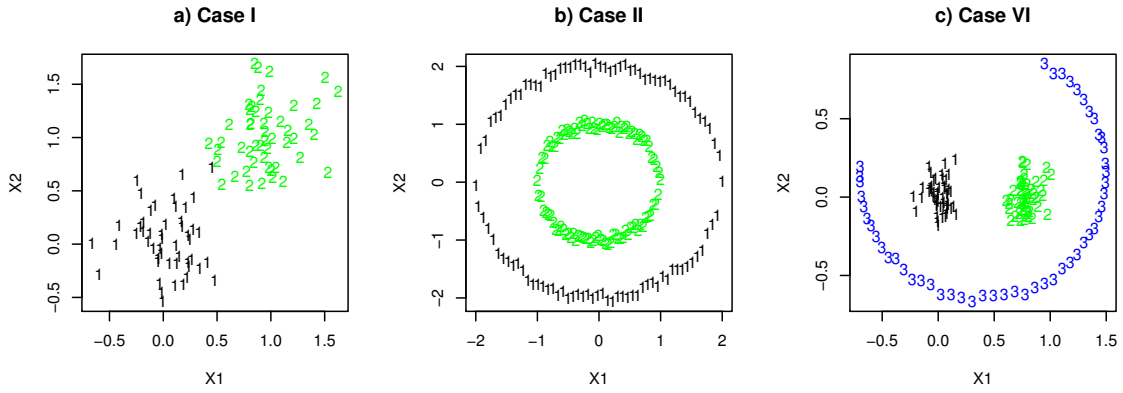



Figure 1: The first simulated data set in a) Case I, b) Case II and c) Case VI.

is similar to the "two-circle" case in Ng et al. (2002), but perhaps more challenging here with larger distances between some points within the same cluster.

Case III: a null case with only a single cluster in 10 dimensions. 200 observations were uniformly distributed over a unit square independently in each of the 10 dimensions. This is scenario (a) in Tibshirani et al. (2001).

Case IV: four clusters in 3 dimensions. The four cluster centers were randomly drawn from $N(0,5I)$ in each simulation; if any of their distance was less than 1.0, then the simulation was abandoned and re-run. In each cluster, 25 or 50 observations were randomly chosen, each drawn from a normal distribution with mean at the cluster center and the identity covariance matrix. This is scenario (c) in Tibshirani et al. (2001).

Case V: two elongated clusters in 3 dimensions. This is similar to scenario (e) in Tibshirani et al. (2001), but with a much shorter distance between the two clusters. Specifically, cluster one was generated as follows: 100 observations were generated be equally spaced along the main diagonal of a three dimensional cube, then independent normal variate with mean 0 and SD=0.1 were added to each coordinate of each of the 100 observations; that is, $x_{ij} = -0.5 + (i-1)/99 + \varepsilon_{ij}$, $\varepsilon_i \sim N(0,0.1)$ for $j = 1$, 2, 3 and $i = 1,...,100$. Cluster 2 was generated in the same way, but with a shift of 2, not 10, making it harder than that used in Tibshirani et al. (2001) in each dimension.

Case VI: three clusters in 2 dimension with two spherically shaped clusters inside 3/4 of a perturbed circle (Figure 1c). This is similar to a case in Ng et al. (2002). Specifically, for cluster 1, we generated $x_{i1} = 1.1\sin(2\pi[30+5(i-1)]/360)$ and $x_{i2} = 0.8\sin(2\pi[30+5(i-1)]/360) + \varepsilon_i$ for $i = 1,...,50$, where $\varepsilon_i$ was randomly drawn from $U(-0.025, 0.025)$; 50 observations were drawn from each of the two bivariate Normal distributions, $N((0,0)', 0.1I)$ and $N((0.8,0)', 0.1I)$.

For each case, we applied the K-mean, HTclust and PRclust to 100 simulated data sets. For the K-means we used 20 random starts for each $K = 1,2,...,20$. For HTclust and PRclust, we did grid-searches for $d$, and $(\tau, \lambda_2)$ respectively. For comparison, we also applied Gaussian mixture-model based clustering as implemented in R package `mclust` (Fraley and Raftery, 2006); for each data set, we fitted each of the 10 models corresponding to 10 different ways of parameterizing the mixture model, for $K = 1,2,...,20$ clusters, and the final model, including the number of clusters, was selected by the Bayesian Information Criterion (BIC).

Due to the conceptual similarity between our proposed PRclust and spectral clustering (Sclust), we also included the spectral clustering algorithm of Ng et al. (2002) as outlined below. First, calculate an affinity matrix $A = (A_{ij})$ with elements $A_{ij} = \exp(-||x_i - x_j||_2^2/\gamma)$ for any two observations $i \neq j$ and $A_{ii} = 0$, where $\gamma$ is a scaling parameter to be determined. Second, calculate a diagonal matrix $D = \text{Diag}(D_{11}, ..., D_{nn})$ with $D_{ii} = \sum_{j=1}^{n} A_{ij}$. Third, calculate $L = D^{-1/2} A D^{-1/2}$. Fourth, for a specified number of clusters $k$, we stack the $k$ top eigen-vectors (corresponding to the $k$ largest eigen-values) of $L$ column-wise to form an $n \times k$ matrix, say $Z_k$; normalize each row of $Z_k$ to have a unit $L_2$-norm. Finally, treating each row of $Z_k$ as an observation, we apply the K-means to form $k$ clusters. There are two tuning parameters $\gamma$ and $k$ that have to be decided in the algorithm. We used the implementation in the R package `kernlab`, which includes a method of Ng et al. (2002) to select $\gamma$ automatically; however, one has to specify $k$. We applied the GCV(GDF) to select $k$ as for the K-means. Unfortunately the function `specc()` in the R package `kernlab` was not numerically stable and sometimes might break down (i.e., exiting with an error message), though it could work in a re-run with a different random seed; the error occurred more frequently with an increasing $k$. Hence we only considered its use in a few cases by restricting $k$ to 1 to 3.

To evaluate the performance of a clustering algorithm, we used the Rand index (Rand, 1971), adjusted Rand index (Hubert and Arabie, 1985) and Jaccard index (Jaccard, 1912), all measuring the agreement between estimated cluster memberships and the truth. Each index is between 0 and 1 with a higher value indicating a higher agreement.

### 3.2 Simulation Results

Case I: For the K-means, we chose the number of clusters using Jump, CV1, CV2 and GCV statistics; for comparison, we also fixed the number of clusters around its true value. The results are shown in Table 1. Both the Jump and GCV with the naive df=$np$ methods tended to select a too large number of clusters. In contrast, the GCV(GDF) performed extremely well: it always chose the correct $K = 2$ clusters. Figure 2 shows how GDF and NDF changed with $K$, the number of clusters in the K-means algorithm, for the first simulated data set. Due to the adaptiveness of the K-means, GDF quickly increased to 150 with $K < 10$ and approached the maximum df=$np = 200$ for $K = 20$. Since GDF was in general much larger than NDF, using GDF penalized more on more complex models (i.e., larger $K$ in the K-means), explaining why GCV(GDF) performed much better than GCV(NDF).

Since the two clusters were formed by observations drawn from two Normal distributions, as expected, the model-based clustering Mclust performed best. In addition, the spectral clustering also worked well.

For PRclust, we searched $\tau \in \{0.1, 0.2, ..., 1\}$ and $\lambda_2 \in \{0.01, 0.05, 0.1, 0.2, 1\}$. PRclust with GCV(GDF) selecting its tuning parameters performed well too: the average number of clusters is close to the truth $K_0 = 2$; the corresponding clustering results had high degrees of agreement with the truth, as evidenced by the high indices. Table 1 also displays the frequencies of the number of clusters selected by GCV(GDF): for the overwhelming majority (98%), either the correct number of cluster $K_0 = 2$ was selected, or a slightly larger $K = 3$ or 4 with very *high* agreement indices was chosen.

For the first (and a typical) simulated data set, we show how PRclust operated with various values of the tuning parameter $\lambda_2$ (while $\lambda_1 = 1$ and $\tau = 0.5$), yielding the solution path for $\hat{\mu}_{i1}$, the first coordinate of $\hat{\mu}_i$ (Figure 3a). Note that, due to the use of a fixed $\lambda_1$, even if all $\hat{\theta}_{ij} = 0$ for

| Case | Method | Selection of $K$ | $\hat{K}$ | Rand | aRand | Jaccard |
|---|---|---|---|---|---|---|
| I | K-means | Jump, $K \in [1,20]$ | 18.11 | 0.563 | 0.119 | 0.119 |
| | | Jump, $K \in [1,10]$ | 2.54 | 0.956 | 0.911 | 0.912 |
| | | CV1 | 2 | 0.984 | 0.967 | 0.968 |
| | | CV2 | 2 | 0.984 | 0.967 | 0.968 |
| | | GCV, df=$Kp$ | 29.75 | 0.536 | 0.064 | 0.064 |
| | | GCV, df=GDF | 2 | 0.984 | 0.967 | 0.968 |
| | | Fixed | 3 | 0.859 | 0.717 | 0.720 |
| | | Fixed | 4 | 0.747 | 0.491 | 0.495 |
| | | Fixed | 5 | 0.704 | 0.405 | 0.408 |
| | Mclust | BIC | 2.01 | 0.983 | 0.966 | 0.967 |
| | Sclust | GCV, df=GDF | 2.05 | 0.973 | 0.946 | 0.953 |
| | | $K \in [1,3]$ | | | | |
| | HTclust | GCV, df=$Kp$ | 60.29 | 0.524 | 0.039 | 0.039 |
| | | GCV, df=GDF | 4.12 | 0.901 | 0.802 | 0.858 |
| | | Fixed | 2 | 0.589 | 0.186 | 0.583 |
| | | Fixed | 3 | 0.711 | 0.426 | 0.690 |
| | | Fixed | 4 | 0.779 | 0.562 | 0.746 |
| | | Fixed | 5 | 0.805 | 0.612 | 0.760 |
| | PRclust | GCV, df=$Kp$ | 87.00 | 0.510 | 0.009 | 0.009 |
| | | GCV, df=GDF | 2.35 | 0.974 | 0.947 | 0.953 |
| | | Subset, freq=1 | 1 | 0.495 | 0.000 | 0.495 |
| | | Subset, freq=72 | 2 | 0.982 | 0.965 | 0.966 |
| | | Subset, freq=19 | 3 | 0.973 | 0.946 | 0.946 |
| | | Subset, freq=7 | 4 | 0.966 | 0.933 | 0.933 |
| | | Subset, freq=1 | 5 | 0.887 | 0.774 | 0.788 |

Table 1: Simulation I results based on 100 simulated data sets with 2 clusters.

a sufficiently large $\lambda_2$, there were still quite some unequal $\hat{\mu}_{i1}$'s, which were all remarkably near their true values 0 or 1. In contrast, with the Lasso penalty, the estimated centroids were always shrunk towards each other, leading to their convergence to the same point at the end and thus much worse performance (Figure 3c). It is also noted that the solution paths with the Lasso penalty were almost linear, compared to the nearly step functions with the gTLP. Figure 3d) shows how HTclust worked. In particular, as pointed out by Ng et al. (2002), HTclust is not robust to outliers: since an "outlier" (lower left corner in Figure 1a) was farthest away from any other observations, it formed its own cluster while all others formed another cluster when the threshold $d$ was chosen to yield two clusters. This example demonstrates different operating characteristics between PRclust and HTclust, offering an explanation of the better performance of PRclust over HTclust.

Case II: Since each cluster was not spherically shaped, and more importantly, the two true cluster centroids completely overlapped with each other, the K-means would not work: it could not distinguish the two clusters. As shown in Table 2, no matter what method was used to choose or fix the number of clusters, the K-means always gave results in low agreement with the truth. The problem with the K-means is its defining a cluster centroid as the mean of the observations assigned

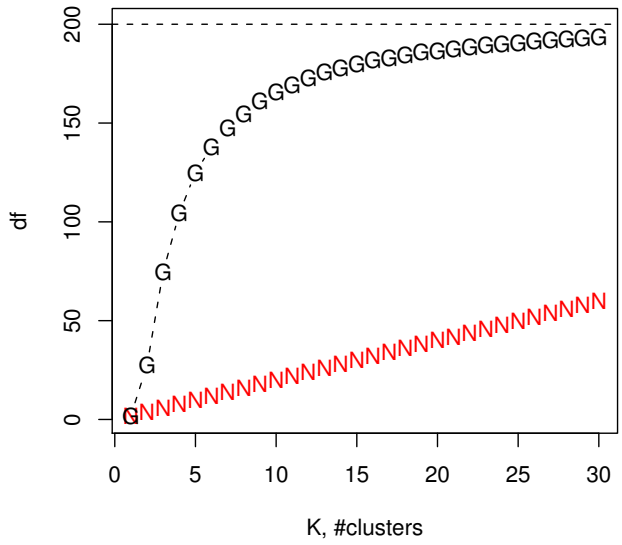



Figure 2: GDF (marked with "G") and NDF (marked with "N") versus the number of clusters, $K$, in the K-means algorithm for the first simulated data set in Case I. The horizontal line gives the maximum df=200.

to the cluster and its assigning a cluster membership of an observation based on its distance to the centroids; since the two clusters share the same centroid in truth, the K-means cannot distinguish the two clusters. Similarly, Mclust did not perform well.

As a comparison, perhaps due to the nature of the local shrinkage in estimating the centroids, PRclust worked much better than the above three methods, as shown in Table 2. Note that, the cluster memberships in PRclust are determined by the estimates of $\theta_{ij} = \mu_i - \mu_j$; due to the use of the ridge penalty with a fixed $\lambda_1 = 1$, we might have $\hat{\theta}_{ij} = 0$ but $\hat{\mu}_i \neq \hat{\mu}_j$.

Since HTclust assigned the cluster-memberships according to the pair-wise distances among the observations, not the nearest distance of an observation to the centroids as done in the K-means, it also performed well.

If the GCV(GDF) was used in Sclust, it would select $\hat{K} = 1$ over $\hat{K} = 2$, even though a specified $\hat{K} = 2$ often led to almost perfect clustering. The reason is that, by symmetry of the two clusters, the two estimated cluster centroids for $\hat{K} = 2$ almost coincided with the estimated centroid of only one cluster, leading to their almost equal RSS (the numerator of the GCV statistics); due to a much larger GDF for $\hat{K} = 2$ than that for $\hat{K} = 1$, the GCV(GDF) statistic for $\hat{K} = 1$ was much smaller than that for $\hat{K} = 2$. Interestingly, an exception happened in four (out of 100) simulations: when Sclust could not correctly distinguish the two true clusters (with low agreement statistics) with $\hat{K} = 2$, it had a smaller GCV(GDF) statistic than that for $\hat{K} = 1$. The results here suggest that, although Sclust may perform well for non-convex clusters with an appropriately chosen $\gamma$ (as selected by the method

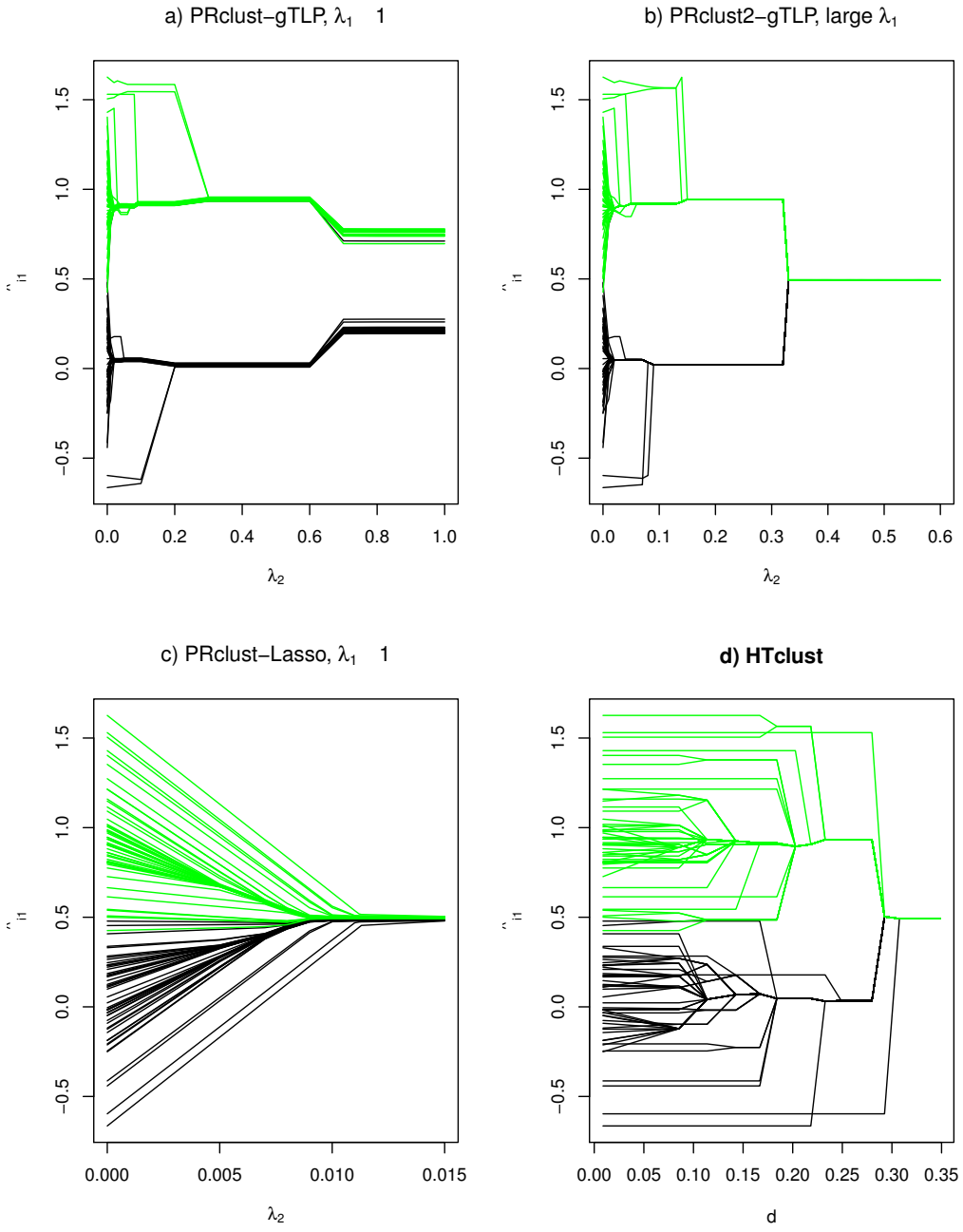



Figure 3: Solution paths of $\hat{\mu}_{i,1}$ for a) PRclust (with gTLP), b) PRclust2, c) PRclust with the Lasso penalty and d) HTclust for the first simulated data set in Case I.

of Ng et al. (2002)), a difficult problem is how to choose the number of clusters; in particular, GCV is not ideal for non-convex clusters.

| Case | Method | Selection of $K$ | $\hat{K}$ | Rand | aRand | Jaccard |
|---|---|---|---|---|---|---|
| II | K-means | Jump, $K \in [1,20]$ | 18.88 | 0.557 | 0.111 | 0.110 |
| | | Jump, $K \in [1,10]$ | 9.99 | 0.597 | 0.191 | 0.195 |
| | | GCV, df=GDF | 2 | 0.498 | -0.005 | 0.329 |
| | | CV1 | 2 | 0.498 | -0.005 | 0.329 |
| | | CV2 | 2 | 0.498 | -0.005 | 0.329 |
| | | Fixed | 3 | 0.498 | -0.006 | 0.261 |
| | | Fixed | 4 | 0.498 | -0.008 | 0.194 |
| | | Fixed | 5 | 0.498 | -0.007 | 0.166 |
| | Mclust | BIC | 16.07 | 0.572 | 0.141 | 0.140 |
| | Sclust | GCV, df=GDF | 1.06 | 0.501 | 0.007 | 0.493 |
| | | $K \in [1,3]$ | | | | |
| | | Subset, freq=95 | 1 | 0.497 | 0.000 | 0.497 |
| | | Subset, freq=4 | 2 | 0.498 | -0.005 | 0.329 |
| | | Subset, freq=1 | 3 | 0.874 | 0.749 | 0.748 |
| | | Fixed | 2 | 0.980 | 0.960 | 0.973 |
| | HTclust | GCV, df=GDF | 3.32 | 0.862 | 0.724 | 0.738 |
| | | Fixed | 2 | 1.000 | 1.000 | 1.000 |
| | | Fixed | 3 | 0.881 | 0.763 | 0.762 |
| | | Fixed | 4 | 0.870 | 0.739 | 0.738 |
| | | Fixed | 5 | 0.866 | 0.732 | 0.731 |
| | PRclust | GCV, df=GDF | 2.93 | 0.895 | 0.791 | 0.790 |
| | | Subset, freq=21 | 2 | 1.000 | 1.000 | 1.000 |
| | | Subset, freq=66 | 3 | 0.880 | 0.759 | 0.759 |
| | | Subset, freq=12 | 4 | 0.810 | 0.620 | 0.619 |
| | | Subset, freq=1 | 5 | 0.746 | 0.491 | 0.490 |

Table 2: Simulation II results based on 100 simulated data sets with 2 clusters.

Cases III-IV: the simulation results are summarized in Table 3. All performed well for the null Case III. Case IV seems to be challenging with partially overlapping spherically shaped clusters of smaller cluster sizes: the number of clusters could be under- or over-selected by various methods. In terms of agreement, overall, as expected, the K-means with GCV(GDF) and Mclust performed best, closely followed by PRclust with GCV(GDF), which performed much better than HTclust.

Cases V-VI: the simulation results are summarized in Table 4. In Case V, all performed perfectly except that the GCV(GDF) over-selected the number of the clusters in the K-means and the two spectral clustering methods. This is interesting since GCV(GDF) seemed to perform well for both HTclust and PRclust. HTclust and PRclust did not yield better clusters than that of the K-means for $K > 2$ clusters, leading to the former two's relatively large GDFs and thus relatively large GCV statistics, while the latter possessed a smaller GDF and GCV, hence GCV(GDF) tended to select a $K > 2$ for the K-means, but not for the other two. Note that the K-means implicitly assumes that all clusters share the same volume and spherical shape, and GCV also implicitly favors such clusters (with smaller within-cluster sum of squares, and thus a smaller GCV statistic). Hence the K-means

| Case | Method | Selection of $K$ | $\hat{K}$ | Rand | aRand | Jaccard |
|---|---|---|---|---|---|---|
| III | K-means | GCV, df=GDF | 1.00 | 1.000 | 1.000 | 1.000 |
| | Mclust | BIC | 1.00 | 1.000 | 1.000 | 1.000 |
| | Sclust | GCV, df=GDF | 1.00 | 1.000 | 1.000 | 1.000 |
| | | $K \in [1,3]$ | | | | |
| | HTclust | GCV, df=GDF | 1.00 | 1.000 | 1.000 | 1.000 |
| | PRclust | GCV, df=GDF | 1.00 | 1.000 | 1.000 | 1.000 |
| IV | K-means | GCV, df=GDF | 3.48 | 0.880 | 0.748 | 0.728 |
| | | CV1 | 3.10 | 0.789 | 0.575 | 0.581 |
| | | CV2 | 4.22 | 0.790 | 0.558 | 0.561 |
| | Mclust | BIC | 3.50 | 0.883 | 0.753 | 0.732 |
| | HTclust | GCV, df=GDF | 6.49 | 0.589 | 0.352 | 0.452 |
| | PRclust | GCV, df=GDF | 4.75 | 0.790 | 0.612 | 0.628 |

Table 3: Simulation Cases III-IV results based on 100 simulated data sets with 1 and 4 clusters, respectively.

| Case | Method | Selection of $K$ | $\hat{K}$ | Rand | aRand | Jaccard |
|---|---|---|---|---|---|---|
| V | K-means | GCV, df=GDF | 7.03 | 0.646 | 0.289 | 0.288 |
| | | CV1 | 2.00 | 1.000 | 1.000 | 1.000 |
| | | CV2 | 2.00 | 1.000 | 1.000 | 1.000 |
| | Mclust | BIC | 2.04 | 0.995 | 0.990 | 0.990 |
| | HTclust | GCV, df=GDF | 2.00 | 1.000 | 1.000 | 1.000 |
| | PRclust | GCV, df=GDF | 2.00 | 1.000 | 1.000 | 1.000 |
| VI | K-means | GCV, df=GDF | 7.95 | 0.902 | 0.761 | 0.704 |
| | | CV1 | 2.00 | 0.722 | 0.444 | 0.497 |
| | | CV2 | 2.00 | 0.722 | 0.444 | 0.497 |
| | Mclust | BIC | 8.04 | 0.906 | 0.769 | 0.714 |
| | HTclust | GCV, df=GDF | 28.02 | 0.872 | 0.678 | 0.611 |
| | PRclust | GCV, df=GDF | 3.08 | 0.997 | 0.993 | 0.993 |

Table 4: Simulation cases V-VI results based on 100 simulated data sets with 2 and 3 clusters, respectively.

divided an elongated cluster into several adjacent spherical clusters, which were then favored by GCV(GDF).

For Case VI, due to the fact of a non-convex cluster, both the K-means with GCV(GDF) and Mclust over-selected the number of clusters, though their agreement statistics were still high. On the other hand, the K-means with CV1 or CV2 and the two spectral clustering methods seemed to under-select the number of clusters, leading to lower agreement statistics. In contrast, PRclust performed much better while HTclust was the worst.

| Method | Selection of $K$ | $\hat{K}$ | Truth: 3 cluster | | | Truth: 2 cluster | | |
|---|---|---|---|---|---|---|---|---|
| | | | Rand | aRand | Jaccard | Rand | aRand | Jaccard |
| K-means | Jump, $K \in [1,30]$ | 27 | 0.710 | 0.154 | 0.122 | 0.490 | 0.069 | 0.077 |
| | Jump, $K \in [1,10]$ | 10 | 0.755 | 0.330 | 0.281 | 0.555 | 0.178 | 0.194 |
| | Jump, $K \in [1,5]$ | 5 | 0.767 | 0.420 | 0.399 | 0.662 | 0.362 | 0.388 |
| | CV1 | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | CV2 | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | GCV, df=GDF | 9 | 0.760 | 0.357 | 0.313 | 0.579 | 0.218 | 0.237 |
| | Fixed | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | Fixed | 3 | 0.832 | 0.620 | 0.594 | 0.777 | 0.570 | 0.597 |
| Mclust | BIC | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | Fixed | 3 | 0.957 | 0.904 | 0.879 | 0.779 | 0.572 | 0.599 |
| Sclust | GCV, df=GDF | 9 | 0.768 | 0.388 | 0.344 | 0.593 | 0.243 | 0.264 |
| | Fixed | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | Fixed | 3 | 0.837 | 0.630 | 0.603 | 0.777 | 0.569 | 0.596 |
| HTclust | GCV, df=GDF | 4 | 0.773 | 0.552 | 0.578 | 0.970 | 0.939 | 0.94 |
| | Fixed | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | Fixed | 3 | 0.772 | 0.558 | 0.562 | 0.996 | 0.991 | 0.992 |
| PRclust | GCV, df=GDF | 3 | 0.777 | 0.564 | 0.589 | 0.982 | 0.965 | 0.968 |
| | Fixed | 2 | 0.776 | 0.568 | 0.595 | 1.000 | 1.000 | 1.000 |
| | Fixed | 3 | 0.775 | 0.562 | 0.599 | 0.987 | 0.974 | 0.977 |

Table 5: Results for Fisher's iris data with 2 or 3 clusters.

### 3.3 Iris Data

We applied the methods to the popular Fisher's iris data. There are 4 measurements on the flower, sepal length, sepal width, petal length and petal width, for each observation. There are 50 observations for each of the three iris subtypes. One subtype is well separated from the other two, but the latter two overlap with each other. For this data set, it is debatable whether there are 2 or 3 clusters; for this reason, for any clustering results, we calculated the agreement indices based on the 3 clusters (each corresponding to each iris subtype), and that based on only 2 clusters by combining the latter two overlapping subtypes into one cluster. Since two observations share an equal value on each variable, there are at most $\hat{K} = 149$ clusters.

We standardized the data such that for each variable we had a sample mean 0 and SD=1. We applied the methods to the standardized data ($p = 4$). We used $v = 0.4$; we tried a few other values of $v$ and obtained similar results for GDF. For the K-means, we tried the number of clusters $K = 1,2,...,30$, each with 20 random starts. For HTclust, we searched 1000 candidate $d$'s according to the empirical distribution of the pair-wise distances among the observations. For PRclust, we tried $\lambda_2 = \in \{0.1, 0.2, ..., 2\}$ and $\tau_2 \in \{1.0, 1.1, ..., 2\}$. The results are shown in Table 5.

For the K-means, in agreement with simulations, the Jump selected perhaps a too large $\hat{K} = 27$, while GCV(GDF) selected $\hat{K} = 9$ perhaps due to the non-spherical shapes of the true clusters (Table 5). Both the K-means with CV1 (or CV2) and Mclust selected $\hat{K} = 2$ and yielded the same clustering results. As for the K-means, GCV(GDF) also selected $\hat{K} = 9$ for Sclust. In comparison, PRclust with

GCV(GDF) yielded $\hat{K} = 3$ clusters with higher agreement indices than those of the K-means, Mclust and Sclust. HTclust selected $\hat{K} = 4$ clusters with the agreement indices less than but close to those of PRclust. We also applied the K-means and Sclust with a fixed $\hat{K} = 2$ or 3, and took the subset of the tuning parameter values yielding 2 or 3 clusters for HTclust and PRclust. It is interesting to note that, with $\hat{K} = 2$, all the methods gave the same results that recovered the two true clusters; however, with $\hat{K} = 3$, the results from PRclust and HTclust were similar, but different from the K-means and Sclust: the K-means and Sclust performed better in terms of the agreement with the 3 true clusters, but less well with the 2 true clusters, than PRclust and HTclust, demonstrating different operating characteristics between the K-means/Sclust and the other two methods. When fixed $\hat{K} = 3$, Mclust gave the best results for $K = 3$, suggesting the advantage of Mclust with overlapping and ellipsoidal clusters.

## 4. Further Modifications and Comparisons

We explore two well-motivated modifications to our new method, which turn out to be less competitive. Then we demonstrate the performance advantages of our new non-convex penalty over several existing convex penalties.

### 4.1 Modifications

In PRclust, so far we have fixed $\lambda_1 = 1$, which cannot guarantee $\hat{\theta}_{ij} = \hat{\mu}_i - \hat{\mu}_j$, even approximately (Figure 3a). As an alternative, following Framework 17.1 of Nocedal and Wright (2000), we start the algorithm at $\lambda_1 = 1$, at convergence we increase the value of $\lambda_1$, for example, by doubling its current value, and re-run the algorithm with the parameter estimates from the previous iteration as its starting values; this process is repeated until the convergence when the parameter estimates barely change. As before, we can use the new estimates $\hat{\theta}_{ij}$'s to form clusters. We call this modified method PRclust2. As shown in Figure 3b), for a sufficiently large $\lambda_2$, we'd have all $\theta_{ij} = 0$, leading to all $\hat{\mu}_{i1}$'s (almost) equal in PRclust2; in contrast, no matter how large $\lambda_2$ was, we had multiple quite distinct $\hat{\mu}_{i1}$'s in PRclust (Figure 3a). We applied PRclust2 to the earlier examples and obtained the following results: when all the clusters were convex, PRclust2 yielded results very similar to those of PRclust; otherwise, their results were different. Table 6 shows some representative results. It is surprising that PRclust performed better than PRclust2 for simulation Case II with two non-convex clusters. A possible explanation lies in their different estimates of $\theta_{ij}$'s, which are used by both PRclust and PRclust2 to perform clustering. PRclust2 yields $\hat{\theta}_{ij} = \hat{\mu}_i - \hat{\mu}_j$ (approximately) while PRclust does not. PRclust2 forms clusters based on the (approximate) equality of $\hat{\mu}_i$'s, while PRclust clusters two observations $i$ and $j$ together if their $\hat{\mu}_i$ and $\hat{\mu}_j$ are close to each other, say, $||\hat{\mu}_i - \hat{\mu}_j||_2 < d_{0,ij}$, where the threshold $d_{0,ij}$ is possibly $(i, j)$-specific. Hence, PRclust2 seems to be more rigid and greedy in forming clusters than PRclust. Alternatively, we can regard PRclust as an early stopped and thus regularized version of PRclust2; it is well known that early stopping is an effective regularization strategy that avoids over-fitting in neural networks and trees (Hastie et al., 2001, p.326).

PRclust forms a cluster based on a connected component of a graph constructed with $\hat{\theta}_{ij}$'s. More generally, one can apply the spectral clustering of Ng et al. (2002) to either $\hat{\mu}_i$'s or $\hat{\theta}_{ij}$'s obtained in PRclust; we call the resulting method PRclust3 and PRclust4 respectively. We propose using the GCV(GDF) to select both the scale parameter in Sclust and the number of clusters. To reduce computational demand, we manually chose a suitable $\gamma$ for Sclust. We applied the methods to the

| Data | Method | Selection of $K$ | $\hat{K}$ | Rand | aRand | Jaccard |
|---|---|---|---|---|---|---|
| Case I | PRclust2 | GCV, df=GDF | 2.28 | 0.980 | 0.959 | 0.960 |
| | PRclust3 | GCV, df=GDF | 2.98 | 0.923 | 0.845 | 0.845 |
| | PRclust4 | GCV, df=GDF | 3.88 | 0.876 | 0.751 | 0.752 |
| Case II | PRclust2 | GCV, df=GDF | 2.00 | 0.498 | -0.005 | 0.329 |
| | PRclust3 | GCV, df=GDF | 2.00 | 0.498 | -0.005 | 0.329 |
| | PRclust4 | GCV, df=GDF | 2.00 | 0.498 | -0.005 | 0.329 |
| Iris | PRclust2 | GCV, df=GDF | 3 | 0.777 | 0.564 | 0.589 |
| | PRclust3 | GCV, df=GDF | 9 | 0.766 | 0.392 | 0.356 |
| | PRclust4 | GCV, df=GDF | 4 | 0.777 | 0.519 | 0.528 |

Table 6: Results for modified PRclust for 100 simulated data sets (2 clusters) or the iris data (3 clusters).

data examples; as shown in Table 6, the two methods did not improve over the original PRclust. As a reviewer suggested, alternatively, we may also apply PRclust, not the K-means, to the eigen-vectors in a modified Sclust; however, it will be challenging to develop computationally more efficient methods to simultaneously choose multiple tuning parameters, that is, $(\gamma, k)$ in Sclust and $(\lambda_2, \tau)$ in PRclust.

### 4.2 Comparison with Some Convex Fusion Penalties

In contrast to our non-convex gTLP penalty, several authors have studied the use of the $L_q$-norm-based convex fusion penalties. Pelckmans et al. (2005) proposed using a fusion penalty based on the $L_q$-norm with the objective function

$$\frac{1}{2}\sum_{i=1}^{n}||x_i-\mu_i||_2^2+\lambda\sum_{i<j}||\mu_i-\mu_j||_q,$$

and proposed an efficient quadratic convex programming-based computing method for $q=1$. Lindsten et al. (2011) recognized the importance of using a group penalty with $q>1$, and applied the Matlab CVX package (Grant and Boyd, 2011) to solve the general convex programming problem for the group Lasso penalty with $q=2$ (Yuan and Lin, 2006). Hocking et al. (2011) exploited the piecewise linearity of the solution paths for $q=1$ or $q=\infty$, and proposed an efficient algorithm for each of $q=1$, 2 and $\infty$ respectively. We call these methods PRclust-$L_q$. Note that PRclust-$L_1$ corresponds to our PRclust-Lasso, for which (and our default PRclust-gTLP) however we have proposed a different computing algorithm, the quadratic penalty method. Importantly, due to the use of the convex penalty, the solution path of PRclust-$L_q$ is quite different from that of PRclust-gTLP. Using the Matlab CVX package, we applied PRclust-$L_q$ with $q\in\{1,2,\infty\}$ to simulation Case I; the results for the first data set are shown in Figure 4. It is clear that the solution path of PRclust-$L_1$ (Figure 4a) was essentially the same as that of PRclust-Lasso (Figure 3c) (while different computing algorithms were applied). More importantly, overall the solution paths of all three PRclust-$L_q$ were similar to each other, sharing the common feature that the estimated centroids were more and more biased towards the overall mean as the penalty parameter $\lambda$ increased. This feature of PRclust-$L_q$ makes it

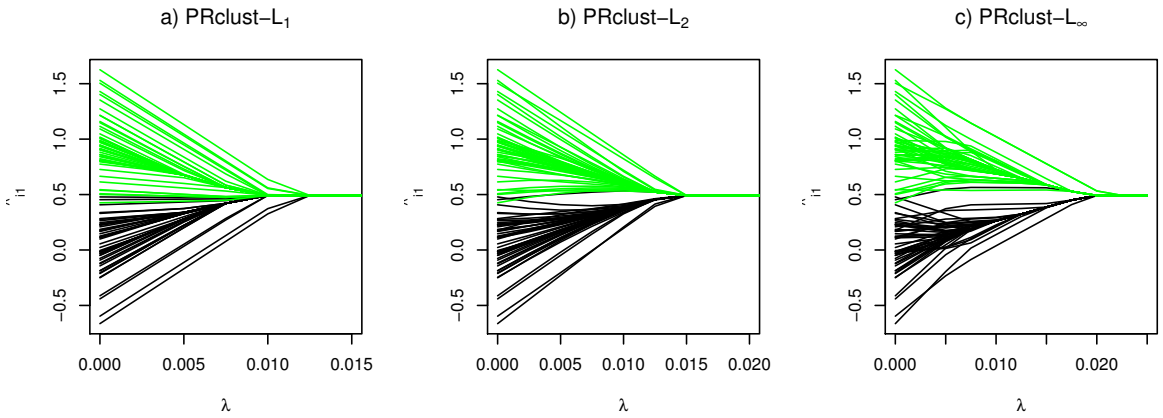



Figure 4: Solution paths of $\hat{\mu}_{i,1}$ for PRclust-$L_q$ with a) $q = 1$, b) $q = 2$ and c) $q = \infty$ for the first simulated data set in Case I.

difficult to correctly select the number of clusters. In fact, both Pelckmans et al. (2005) and Hocking et al. (2011) treated PRclust-$L_q$ as a hierarchical clustering tool; none of the authors discussed the choice of the number of clusters. The issue of an $L_q$-norm penalty in yielding possibly severely biased estimates is well known in penalized regression, which partially motivated the development of non-convex penalties such as TLP (Shen et al., 2012). In the current context, Lindsten et al. (2011) has recognized the issue of the biased centroid estimates in PRclust-$L_q$ and thus proposed a second stage to re-estimate the centroids after a clustering result is obtained. In contrast, with the use of the non-convex gTLP, the above issues are largely avoided as shown in Figure 3ab).

When we applied the GCV(GDF) to select the number of clusters for PRclust-$L_q$ in simulation Case I, as expected, it performed poorly. Hence, for illustration, we considered an ideal (but not practical) alternative. For any $d_0 \geq 0$, similar to hierarchical clustering, we defined an adjacency matrix $A = (a_{ij})$ with $a_{ij} = I(||\hat{\mu}_i - \hat{\mu}_j||_2 \leq d_0)$; any two observations $x_i$ and $x_j$ were assigned to the same cluster if $a_{ij} = 1$. Then for any given $\lambda > 0$ and $d_0 \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 0\}$, we calculated the Rand index for the corresponding PRclust-$L_q$ results and the *true* cluster memberships. We show the results of PRclust-$L_q$ with the values of $(\lambda, d_0)$ achieving the maximum Rand index, giving an upper bound on the performance of PRclust-$L_q$ with any practical criterion to select the number of clusters. As shown in Table 7, a larger value of $q$ seemed to give better ideal performance of PRclust-$L_q$; when compared to PRclust-gTLP (Table 1), none of the three PRclust-$L_q$ methods, even in the ideal case of using the true cluster memberships to select the number of clusters, performed better in selecting the correct number of clusters than PRclust-gTLP with the GCV(GDF) criterion.

## 5. Discussion

The proposed PRclust clustering bears some similarity to the K-means in terms of the objective in minimizing the sum of squared distances between observations and their cluster centroids, however they differ significantly in their specific formulations, algorithms, and importantly, operating characteristics. Consequently, PRclust can perform much better than the K-means in situations un-

| Data | Method | $\hat{K}$ | Rand | aRand | Jaccard |
|---|---|---|---|---|---|
| Case I | PRclust-$L_1$ | 10.42 | 0.860 | 0.719 | 0.723 |
| | PRclust-$L_2$ | 5.56 | 0.949 | 0.899 | 0.900 |
| | PRclust-$L_\infty$ | 3.06 | 0.976 | 0.951 | 0.952 |

Table 7: Results for PRclust-$L_q$ with $\hat{K}$ selected by maximizing the Rand index for 100 simulated data sets (2 clusters) in Case I.

suitable or difficult to the K-means, such as in the presence of non-convex clusters, as demonstrated in our simulation Case II (Table 2). Similarly, Mclust does not perform well for non-convex clusters (Table 2), but may have advantages with overlapping and ellipsoidal clusters as for simulation Case I (Table 1) and the iris data (Table 5). There is also some similarity between PRclust and HTclust (or single-linkage hierarchical clustering). Although much simpler, HTclust does not have any mechanism for shrinkage estimation, and in general did not perform better than PRclust in our examples. Between PRclust and spectral clustering, it seems that they are complementary to each other, though it remains challenging to develop competitive model selection criteria for spectral clustering. For example, our results demonstrated the effectiveness of the method of Ng et al. (2002) in selecting the scale parameter $\gamma$, but the clustering result also critically depended on the specified $k$, the number of clusters, for which the GCV(GDF) might not perform well. Although Zelnik-Manor and Perona (2004) have proposed a model selection criterion to self-tune the two parameters $\gamma$ and $k > 1$, it does not work for $k = 1$; if $k = 1$ is included, the criterion will always select $k = 1$. More generally, model selection is related to kernel learning in spectral clustering (Bach and Jordan, 2006). It is currently an open problem whether the strengths of PRclust and spectral clustering can be combined.

PRclust can be extended in several directions. First, rather than the squared error loss, we can use other loss functions. Corresponding to modifying the K-means to the K-medians, K-midranges or K-modes (Steinley, 2006), we can use an $L_1$, $L_\infty$ and $L_0$ loss function, respectively. Computationally, an efficient coordinate-wise algorithm can be implemented for penalized regression with an $L_1$ loss (Friedman et al., 2007; Wu and Lange, 2008), but it is unclear how to do so for the other two. K-median clustering is closely related to partitioning-around-centroids (PAM) of Kaufman and Rousseeuw (1990), and is more robust to outliers than is the K-means. A modification of PRclust along this direction may retain this advantage. Second, rather than assuming spherically shaped clusters, as implicitly used by the K-means, we can use a general covariance matrix $V$ with a loss function

$$L(x_i - \mu_i) = \frac{1}{2}(x_i - \mu_i)' V^{-1} (x_i - \mu_i),$$

where $V$ is either given or to be estimated. A non-identity $V$ allows a more general model of ellipsoidal clusters. Alternatively, we can also relax the equal cluster volume assumption and use:

$$L(x_i - \mu_i) = \frac{1}{2}(x_i - \mu_i)'(x_i - \mu_i)/\sigma_i^2,$$

where observation-specific variances $\sigma_i^2$'s have to be estimated through grouping pursuit, as for observation-specific means/centroids $\mu_i$'s (Xie et al., 2008). More generally, corresponding to the more general Gaussian mixture model-based clustering (Banfield and Raftery, 1993; McLachlan

and Peel, 2002), we might use

$$L(x_i - \mu_i) = \frac{1}{2}(x_i - \mu_i)' V_i^{-1}(x_i - \mu_i),$$

for a general and observation-specific covariance matrix $V_i$, though it will be challenging to adopt a suitable grouping strategy to estimate $V_i$'s effectively. Among others, it might provide a computationally more efficient algorithm than the EM algorithm commonly adopted in mixture model-based clustering (Dempster et al., 1977). Equally, we may accordingly modify the RSS term in GCV so that it will not overly favor spherically shaped clusters. Third, in our current implementation, after parameter estimation, we construct an adjacency matrix and search connected components in the corresponding graph to form clusters. This is a special and simple approach to more general graph-based clustering (Xu and Wunsch, 2005); other more sophisticated approaches may be borrowed or adapted. We implemented a specific combination of PRclust and spectral clustering along with GCV(GDF) for model selection: we first applied PRclust, then used its output as the input to spectral clustering, but it did not show improvement over PRclust. Other options exist; for example, as suggested by a reviewer, it might be more fruitful to replace the K-means in spectral clustering with PRclust. These problems need to be further investigated. Fourth, in the quadratic penalty method, rather than fixing $\lambda_1 = 1$ or allowing $\lambda_1 \to \infty$, we may want to treat $\lambda_1$ as a tuning parameter; a challenge is to develop computationally more efficient methods (e.g., than data perturbation-based GCV estimation) to select multiple tuning parameters. Alternatively, as a reviewer suggested, we may also apply the alternating direction method of multipliers (ADMM) (Boyd et al., 2011), which is closely related to, but perhaps more general and simpler than the quadratic penalty method. Finally, we have not applied the proposed method to high-dimensional data, for which variable selection is necessary. In principle, we may add a penalty into our objective function for variable selection (Pan and Shen, 2007), which again requires a fast method to select more tuning parameters and is worth future investigation.

Perhaps the most interesting idea of our proposal is the view of regarding clustering analysis as a penalized regression problem, blurring the typical line drawn to distinguish clustering (or unsupervised learning) with regression and classification (i.e., supervised learning). This not only opens a door to using various regularization techniques recently developed in the context of penalized regression, such as novel non-convex penalties and algorithms, but also facilitates the use of other model selection techniques. In particular, we find that our proposed regression-based GCV with GDF is promising for the K-means and PRclust (but perhaps not for spectral clustering) in selecting the number of clusters, a hard and interesting problem in itself; since this is not the main point of this paper, we wish to report more on this topic elsewhere.

## Acknowledgments

The authors thank Junhui Wang for sharing his R code implementing CV1 and CV2 methods, and thank the action editor and reviewers for their helpful and constructive comments. This research was supported by NSF grants DMS-0906616 and DMS-1207771, and NIH grants R01GM081535, R01HL65462 and R01HL105397.

## Appendix A.

We prove Theorem 1 in Section 2.2.

By construction of $S^{(m)}(\mu,\theta)$ and the definition of minimization, for each $m \in N$,

$$\begin{aligned} 0 &\leq S(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) = S^{(m+1)}(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) \leq S^{(m)}(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) \\ &\leq S^{(m)}(\hat{\mu}^{(m)},\hat{\theta}^{(m-1)}) \leq S^{(m)}(\hat{\mu}^{(m-1)},\hat{\theta}^{(m-1)}) = S(\hat{\mu}^{(m-1)},\hat{\theta}^{(m-1)}), \end{aligned}$$

implying that $S(\hat{\mu}^{(m)},\hat{\theta}^{(m)})$ decreases in $m$. Note that $S(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) \geq 0$ for all $m$. Then it converges, and $S(\hat{\mu}^{(m)},\hat{\theta}^{(m)})$ must decreases strictly in $m$ before meeting the stopping rule to terminate. Moreover, by construction, $S^{(m+1)}(\mu,\theta)$ has only a finite number of distinctly different functions in $m$. This implies that $S^{(m+1)}(\hat{\mu}^{(m)},\hat{\theta}^{(m)}) = S(\hat{\mu}^{(m)},\hat{\theta}^{(m)})$ has a finite number of different minimizers across all $m \in N$, hence termination must occur finitely.

To show that $(\hat{\mu}^{(m^\star)},\hat{\theta}^{(m^\star)})$ is a local minimizer of $S(\mu,\theta)$, we check if it satisfies a local optimality of $S(\mu,\theta)$, defined by regular subdifferentials (Rockafellar and Wets, 2003):

$$[1+\lambda_1(n-1)]\mu_i - x_i - \lambda_1 \sum_{j>i}(\mu_j+\theta_{ij}) - \lambda_1 \sum_{j<i}(\mu_j-\theta_{ji}) = 0,\ i = 1,...,n, \tag{4}$$

$$-\lambda_1(\mu_i-\mu_j-\theta_{ij}) + \lambda_2 b_{ij}\frac{\theta_{ij}}{||\theta_{ij}||_2} = 0,\ i,j = 1,...,n\ (i<j), \tag{5}$$

where $b_{ij}$ is the regular subdifferential of $\min(||\theta_{ij}||_2,\tau)$ at $||\theta_{ij}||_2$. Note that $(\hat{\mu}^{(m^\star)},\hat{\theta}^{(m^\star)}) = (\hat{\mu}^{(m^\star-1)},\hat{\theta}^{(m^\star-1)})$ at termination. Then (4) is satisfied with $(\mu,\theta) = (\hat{\mu}^{(m^\star-1)},\hat{\theta}^{(m^\star-1)})$. For (5), we discuss three cases. If $||\hat{\theta}_{ij}^{(m^\star-1)}||_2 > \tau$, then $\hat{\theta}_{ij}^{(m^\star-1)} = \hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}$, implying (5) when $\theta_{ij} = \hat{\theta}_{ij}^{(m^\star)}$ because $b_{ij} = 0$. If $0 < ||\hat{\theta}_{ij}^{(m^\star)}||_2 < \tau$ and $||\hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}||_2 \geq \frac{\lambda_2}{\lambda_1}$, then

$$\hat{\theta}_{ij}^{(m^\star)} = (||\hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}||_2 - \frac{\lambda_2}{\lambda_1})\frac{\hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}}{||\hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}||_2},$$

hence that $||\hat{\theta}_{ij}^{(m^\star)}||_2 = ||\hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}||_2 - \frac{\lambda_2}{\lambda_1}$. Then (4) is met when $\theta_{ij} = \hat{\theta}_{ij}^{(m^\star)}$ because $b_{ij} = 1$. If $0 < ||\hat{\theta}_{ij}^{(m^\star)}||_2 < \tau$ and $||\hat{\mu}_i^{(m^\star)} - \hat{\mu}_j^{(m^\star)}||_2 < \frac{\lambda_2}{\lambda_1}$, then $||\hat{\theta}_{ij}^{(m^\star)}||_2 = 0$, which is contrary to the fact that $0 < ||\hat{\theta}_{ij}^{(m^\star)}||_2 < \tau$. This completes the proof.

## Appendix B.

We prove the equivalence between HTclust and the single-linkage hierarchical clustering (SL-Hclust).

Both the HTclust and SL-Hclust form clusters sequentially and in a finite number of steps; we show that, in each step, the SL-Hclust gives the same clusters as those of HTclust if an appropriate threshold is chosen in the latter. Suppose that the distinct values of $d_{ij} = ||x_i - x_j||_2$ for all $i \neq j$ are ordered from the smallest to the largest as $d_{(1)}, d_{(2)}, ..., d_{(m)}$. At the beginning, each observation forms its own cluster in both the HTclust and SL-Hclust. In the next step, the SL-Hclust combines observations according to whether their distances satisfy $d_{ij} \leq d_{(1)}$ to form clusters; in HTclust, if we use a threshold $d_0 = d_{(1)} + \varepsilon$ with a tiny $\varepsilon > 0$, then it results in the same clusters as those of the SL-Hclust. If the clustering results of the two methods are the same in step $k-1 > 0$ and if we use $d_0 = d_{(k)} + \varepsilon$ in HTclust, then it leads to the same clusters as the SL-Hclust in step $k$. By induction, this completes the proof.

## References

L. An and P. Tao. Solving a class of linearly constrained indefinite quadratic problems by D.C. algorithms. *J. Global Optimization*, 11:253-285, 1997.

F.R. Bach and M.I. Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963-2001, 2006.

J.D. Banfield and A.E. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49:803-821, 1993.

S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1-122, 2011.

A.P. Dempster, N.M. Laird and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *JRSS-B*, 39:1-38, 1977.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and it oracle properties. *JASA*, 96:1348-1360, 2001.

C. Fraley and A.E. Raftery. MCLUST Version 3 for R: Normal Mixture Modeling and Model-Based Clustering. Technical Report no. 504, Department of Statistics, University of Washington. 2006.

J. Friedman, T. Hastie, H. Hofling and R. Tibshirani. Pathwise coordinate optimization. *Ann Appl Statistics*, 2:302-332, 2007.

G.H. Golub, M. Heath and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215-223, 1979.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21, 2011. *http://cvxr.com/cvx*

T. Hastie, R. Tibshirani and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2001.

T. Hocking, A. Joulin, F. Bach and J.-P. Vert. Clusterpath: An Algorithm for Clustering using Convex Fusion Penalties. In L. Getoor and T. Scheffer (Eds.), *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*, p.745-752, 2011.

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193-218, 1985.

P. Jaccard. The distribution of flora in the alpine zone. *New Phytologist.*, 11:37-50, 1912.

L. Kaufman and P.J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, New York, 1990.

F. Lindsten, H. Ohlsson and L. Ljung, L. Clustering using sum-of-norms regularization: With application to particle filter output computation. *2011 IEEE Statistical Signal Processing Workshop (SSP)*, pages 201-204. 2011.

G.J. McLachlan and D. Peel. *Finite Mixture Model.* New York, John Wiley & Sons, Inc. 2002.

A. Ng, M. Jordan and Y. Weiss. On spectral clustering: analysis and an algorithm. *NIPS*, 2002.

J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2000.

W. Pan and X. Shen. Penalized model-based clustering with application to variable selection. *Journal of Machine Learning Research*, 8:1145-1164, 2007.

K. Pelckmans, J. De Brabanter, J.A.K. Suykens and B. De Moor. Convex Clustering Shrinkage. Workshop on Statistics and Optimization of Clustering Workshop (PASCAL), London, U.K., Jul. 2005. Available at ftp://ftp.esat.kuleuven.ac.be/pub/SISTA/kpelckma/ccs_pelckmans2005.pdf.

W.M. Rand. Objective criteria for the evaluation of clustering methods. *JASA*, 66:846-850, 1971.

R.T. Rockafellar and R.J. Wets. *Variational Analysis*. Springer-Verlag, 2003.

X. Shen and J. Ye. Adaptive model selection. *JASA*, 97:210-221, 2002.

X. Shen and H.-C. Huang. Grouping pursuit through a regularization solution surface. *JASA*, 105:727-739, 2010.

X. Shen, W. Pan and Y. Zhu. Likelihood-based selection and sharp parameter estimation. *JASA*, 107:223-232, 2012.

D. Steinley. K-means clustering: a half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, 59:1-34, 2006.

C.A. Sugar and G.M. James. Finding the number of clusters in a data set: An information theoretic approach. *Journal of the American Statistical Association*, 98:750-763, 2003.

A. Thalamuthu, I. Mukhopadhyay, X. Zheng and G.C. Tseng. Evaluation and comparison of gene clustering methods in microarray analysis. *Bioinformatics*, 22:2405-2412, 2006.

R. Tibshirani. Regression shrinkage and selection via the lasso. *JRSS-B*, 58:267-288, 1996.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu and K. Knight. Sparsity and smoothness via the fused lasso. *JRSS-B*, 67:91-108, 2005.

R. Tibshirani, G. Walther and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *JRSS-B*, 63:411-423, 2001.

P. Tseng. Convergence of block coordinate descent method for nondifferentiable maximization. *J. Opt. Theory Appl.*, 109:474-494, 2001.

J. Wang. Consistent selection of the number of clusters via crossvalidation. *Biometrika*, 97:893-904, 2010.

T.T. Wu and K. Lange. Coordinate descent algorithms for Lasso penalized regression. *Ann Appl Statistics*, 2:224-244, 2008.

B. Xie, W. Pan and X. Shen. Penalized model-based clustering with cluster-specific diagonal covariance matrices and grouped variables. *Electronic Journal of Statistics*, 2:168-212, 2008.

R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Trans Neural Networks*, 16:645-678, 2005.

J. Ye. On measuring and correcting the effects of data mining and model selection. *JASA*, 93:120-131, 1998.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *JRSS-B*, 68:49-67, 2006.

L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. *NIPS*, 2004.

Journal of Machine Learning Research 14 (2013) 1891-1945 Submitted 10/12; Revised 2/13; Published 7/13

# Generalized Spike-and-Slab Priors for Bayesian Group Feature Selection Using Expectation Propagation

**Daniel Hernández-Lobato** DANIEL.HERNANDEZ@UAM.ES
*Computer Science Department*
*Universidad Autónoma de Madrid*
*Calle Francisco Tomás y Valiente 11*
*28049 Madrid, Spain*

**José Miguel Hernández-Lobato** JMH233@ENG.CAM.AC.UK
*Department of Engineering*
*University of Cambridge*
*Trumpington Street*
*Cambridge CB2 1PZ, United Kingdom*

**Pierre Dupont** PIERRE.DUPONT@UCLOUVAIN.BE
*Machine Learning Group, ICTEAM*
*Université catholique de Louvain*
*Place Sainte Barbe 2*
*1348, Louvain-la-Neuve, Belgium*

**Editor:** David Blei

## Abstract

We describe a Bayesian method for group feature selection in linear regression problems. The method is based on a generalized version of the standard spike-and-slab prior distribution which is often used for individual feature selection. Exact Bayesian inference under the prior considered is infeasible for typical regression problems. However, approximate inference can be carried out efficiently using Expectation Propagation (EP). A detailed analysis of the generalized spike-and-slab prior shows that it is well suited for regression problems that are sparse at the group level. Furthermore, this prior can be used to introduce prior knowledge about specific groups of features that are *a priori* believed to be more relevant. An experimental evaluation compares the performance of the proposed method with those of group LASSO, Bayesian group LASSO, automatic relevance determination and additional variants used for group feature selection. The results of these experiments show that a model based on the generalized spike-and-slab prior and the EP algorithm has state-of-the-art prediction performance in the problems analyzed. Furthermore, this model is also very useful to carry out sequential experimental design (also known as active learning), where the data instances that are most informative are iteratively included in the training set, reducing the number of instances needed to obtain a particular level of prediction accuracy.

**Keywords:** group feature selection, generalized spike-and-slab priors, expectation propagation, sparse linear model, approximate inference, sequential experimental design, signal reconstruction

## 1. Introduction

Many regression problems of practical interest are characterized by a small number of training instances $n$ and a large number of explanatory variables or features $d$. Examples of these problems

©2013 Daniel Hernández-Lobato, José Miguel Hernández-Lobato and Pierre Dupont.

include the reconstruction of medical images (Seeger et al., 2009), the processing of fMRI data (Gerven et al., 2009), the discovery of gene regulators (Hernández-Lobato et al., 2008), the processing of natural language (Sandler et al., 2009) or the reconstruction of transcription networks (Hernández-Lobato and Dijkstra, 2010). Under these circumstances a simple linear model is typically assumed to describe the observed data. However, when $n \ll d$ the problem of estimating the optimal linear relation is under-determined. In particular, there is an infinite number of values for the model coefficients that explain the data equally well (Johnstone and Titterington, 2009). To address this difficulty and also to alleviate over-fitting, a practical solution that is typically employed is to assume that only a few of all the explanatory variables or features are actually relevant for prediction. The consequence is that the estimation problem is generally regularized by assuming that most of the model coefficients in the optimal solution are exactly equal to zero (Johnstone and Titterington, 2009). That is, the vector of model coefficients is sparse. This means that the features whose associated model coefficients take value equal to zero do not contribute to the decisions made by the model and are hence considered to be irrelevant. To estimate the model coefficients under the sparsity assumption different strategies described in the literature can be used. These include introducing regularization norms or assuming sparse enforcing priors in the estimation process (Tibshirani, 1996; George and McCulloch, 1997; Tipping, 2001; Kappen and Gómez, 2013).

The process of inducing the model coefficients under the sparsity assumption can be facilitated when prior information is available about groups of features that are expected to be jointly relevant or jointly irrelevant for prediction (Huang and Zhang, 2010). That is, when different groups of model coefficients are expected to be jointly equal to or jointly different from zero. Finding this type of information can be difficult in practice. However, such prior information can be deduced from a related task (see, e.g., Section 7.2) or from some additional data (see, e.g., Section 7.3). Furthermore, if this information is available and it is accurate, it can be beneficial to improve the estimates of the model coefficients and to reduce the number of samples required to obtain a good generalization performance. As described by Puig et al. (2011) the class of problems where sparsity at the group level is beneficial include spectrum cartography for cognitive radio (Bazerque et al., 2011), jointly-sparse signal recovery (Wakin et al., 2006), regression with grouped variables (Yuan and Lin, 2006) and source localization (Malioutov et al., 2005). Other classes of problems that can benefit from group sparsity are multi-task feature selection (Hernández-Lobato et al., 2010) or whole genome association mapping (Kim and Xing, 2008). As with the individual sparsity assumption, sparsity at the group level can be introduced in the estimation process of the model coefficients by considering specific regularization norms or by assuming sparse enforcing priors at the group level (Yuan and Lin, 2006; Ji et al., 2009; Vogt and Roth, 2010; Raman et al., 2009; Yen and Yen, 2011). For this purpose, we specifically consider a generalized version of the standard spike-and-slab prior which has been typically employed for individual feature selection (Mitchell and Beauchamp, 1988; Geweke, 1996; George and McCulloch, 1997). Under the assumption that the grouping information is given beforehand, the proposed prior introduces a set of binary latent variables, one for each different group of features. If the latent variable of a particular group is equal to zero, the model coefficients corresponding to that group are set to zero and the features of the group are not used for prediction of the targets. On the other hand, if the latent variable is equal to one, the features of that particular group are used for prediction and the model coefficients are assumed to be generated from a multi-variate Gaussian distribution. When there is no grouping information this prior reduces to the standard spike-and-slab prior. Exact Bayesian inference under the prior considered is infeasible

for typical regression problems. Thus, in practice one has to resort to approximate techniques for Bayesian inference.

The proposed generalized version of the spike-and-slab prior has several practical advantages over other methods for group feature selection. In particular, it is the only prior that puts a positive probability mass on values equal to zero for the model coefficients of each group. Furthermore, introducing the prior fraction of relevant groups for prediction or the expected deviation from zero of the coefficients that are actually different from zero is very easy under this prior. The proposed prior also has a closed form convolution with the Gaussian distribution, which allows the use of efficient algorithms for approximate Bayesian inference like expectation propagation (EP) (Minka, 2001). This is impossible with other existing priors that can also be used for group feature selection like the multi-variate Laplace distribution (Raman et al., 2009) or the multi-variate horseshoe (Carvalho et al., 2009). The proposed prior also provides a direct estimate of the relative importance of each group for prediction, hence identifying easily the most relevant groups. This estimate is simply obtained by computing the posterior probability that the different latent variables of the prior are activated.

The EP algorithm has a total running time under this prior that scales as $O(n^2 d)$, where $n$ is the number of training instances and $d$ is the number of features. This linear dependence on $d$ is very efficient when $n \ll d$, which is the general scenario we assume. The EP algorithm also provides a direct estimate of the posterior covariances of the model coefficients. These are useful for carrying out sequential experimental design in a linear model that incorporates the grouping information using the proposed generalized spike-and-slab prior. Specifically, in sequential experimental design we save on costly experiments by iteratively including in the training set the data instances that are most informative about the model coefficients (Seeger, 2008). Our experiments indicate that EP and the proposed prior are very effective towards this end. Additionally, a detailed analysis of the sparsity properties of the generalized prior shows that it is adequate for group feature selection. In particular, it produces selective shrinkage of the different groups of the model coefficients. Namely, under this prior it is possible to achieve high levels of group sparsity and, at the same time, to avoid shrinking the model coefficients that are truly different from zero. We show that this is not possible with other popular methods for group feature selection. Finally, incorporating prior knowledge about specific groups of features that are more likely to be used for prediction is straight-forward under the generalized spike-and-slab prior. This type of prior information can be very useful to improve the prediction performance.

The performance of a model based on the generalized spike-and-slab prior and the EP algorithm is evaluated on a collection of benchmark regression problems and compared with other methods from the literature that can also be used for group feature selection. The problems investigated include the reconstruction of sparse signals from a reduced set of noisy measurements (Ji et al., 2008), the prediction of user sentiment (Blitzer et al., 2007), and the reconstruction of images of hand-written digits extracted from the MNIST data set (LeCun et al., 1998). The methods we compare with include the group LASSO (Yuan and Lin, 2006; Kim et al., 2006), the Bayesian group LASSO (Raman et al., 2009), a modified version of the horseshoe prior for group feature selection (Carvalho et al., 2009) and the group automatic relevance determination (ARD) principle (Ji et al., 2009). We also include for comparison a model which does not consider the grouping information and a model that uses Markov chain Monte Carlo methods (Gibbs sampling) for approximate inference instead of the EP algorithm. The results of these experiments indicate that the grouping information can significantly improve the performance of the prediction models. Furthermore, the

model based on the proposed prior and the EP algorithm generally performs best in the problems investigated. There is little difference between the performance obtained by using EP or Gibbs sampling for approximate inference. This confirms the accuracy of the posterior approximation computed by EP, while the EP algorithm is orders of magnitude faster than Gibbs sampling. The running time of EP is also better than or similar to the running time of the other methods analyzed in this document. These experiments also show that the proposed prior is very useful to identify the most relevant groups for prediction and that prior knowledge about specific groups that are *a priori* expected to be more relevant for prediction can significantly improve the prediction performance of the resulting model.

The rest of this document is organized as follows. Section 2 describes the generalized spike-and-slab prior considered for group feature selection. Section 3 shows how the EP algorithm can be successfully applied for approximate inference in a linear regression model based on this prior. Section 4 introduces sequential experimental design and shows how it can be efficiently implemented in the model based on the generalized spike-and-slab prior and the EP algorithm. Section 5 gives a summary of other methods that are available in the literature to carry out group feature selection. Section 6 introduces a detailed analysis of the group sparsity properties of the generalized spike-and-slab prior and the other methods that can be used for this purpose. Section 7 shows some experiments comparing the different methods that can be used for group feature selection and, finally, Section 8 presents the conclusions of this work.

## 2. Group Feature Selection Using Spike-and-Slab Priors

In this section we describe the linear regression model which promotes sparsity at the group level using a generalized spike-and-slab prior. Consider some training data in the form of $n$ $d$-dimensional feature vectors summarized in a design matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^{\mathrm{T}}$ and some associated target values $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$ with $y_i \in \mathbb{R}$. A linear predictive rule is assumed for $\mathbf{y}$ given $\mathbf{X}$:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}\,, \tag{1}$$

where $\mathbf{w}$ is a vector of unknown model coefficients and $\boldsymbol{\epsilon}$ is a $n$-dimensional vector of independent additive Gaussian noise with variance $\sigma_0^2$, that is, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_0^2\mathbf{I})$. Given $\mathbf{X}$ and $\mathbf{y}$, the likelihood of $\mathbf{w}$ is defined as

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^{n} p(y_i|\mathbf{x}_i, \mathbf{w}) = \prod_{i=1}^{n} \mathcal{N}(y_i|\mathbf{w}^{\mathrm{T}}\mathbf{x}_i, \sigma_0^2)\,. \tag{2}$$

When $d \gg n$, (2) is not strictly concave and infinitely many values of $\mathbf{w}$ fit the training data equally well with perfect prediction accuracy. These are precisely the type of problems we are interested in. A strong regularization technique that is typically employed in this context is to assume that $\mathbf{w}$ is sparse. We further assume the availability of prior information about groups of components of $\mathbf{w}$ that are expected to be jointly zero or jointly different from zero. This is equivalent to considering specific groups of features that are expected to be jointly relevant or jointly irrelevant for prediction. All these assumptions can be incorporated into the model using a generalized version of the standard *spike-and-slab* prior (Mitchell and Beauchamp, 1988; Geweke, 1996; George and McCulloch, 1997).

Consider a partition of $\mathbf{w}$ into $G$ disjoint groups (in general we do not allow for groups of model coefficients to overlap) such that $\mathbf{w} = (\mathbf{w}_1^{\mathrm{T}}, \ldots, \mathbf{w}_G^{\mathrm{T}})^{\mathrm{T}}$. We introduce a vector $\mathbf{z}$ with $G$ binary latent

variables $\{z_1,\ldots,z_G\}$, where each $z_g$ indicates whether $\mathbf{w}_g$ is zero ($z_g = 0$) or different from zero ($z_g = 1$). When $\mathbf{z}$ is known, the prior for $\mathbf{w}$ is defined as:

$$\begin{aligned}\mathcal{P}(\mathbf{w}|\mathbf{z}) &= \prod_{g=1}^{G} \left[z_g \mathcal{N}(\mathbf{w}_g|\mathbf{0}, v_0\mathbf{I}) + (1-z_g)\delta(\mathbf{w}_g)\right] \\ &= \prod_{j=1}^{d} \left[z_{g(j)} \mathcal{N}(w_j|0, v_0) + (1-z_{g(j)})\delta(w_j)\right], \end{aligned} \tag{3}$$

where $g(j)$ is the index of the group which contains the $j$-th coefficient, $\mathcal{N}(\cdot|\mathbf{0}, v_0\mathbf{I})$ is a Gaussian density, with zero mean and a group specific variance $v_0$ (the slab), and $\delta(\cdot)$ is a point probability mass centered at the origin (the spike). The value of $v_0$ controls the shrinkage of the coefficients that are different from zero. If $v_0$ is large, the coefficients of the groups that are different from zero are barely regularized. Conversely, if $v_0$ is small these coefficients are strongly shrunk towards zero. Finally, the prior for $\mathbf{z}$ is a multivariate Bernoulli distribution:

$$\mathcal{P}(\mathbf{z}) = \text{Bern}(\mathbf{z}|\mathbf{p}_0) = \prod_{g=1}^{G} \left[p_{0,g}^{z_g}(1-p_{0,g})^{(1-z_g)}\right], \tag{4}$$

where $p_{0,g}$ is the prior probability that the coefficients within the $g$-th group are different from zero and $\mathbf{p}_0 = (p_{0,1},\ldots,p_{0,G})^{\mathrm{T}}$. Thus, incorporating prior knowledge about specific groups of features that are more likely to be used for prediction is straight-forward under (4). For this we only have to increase the corresponding components of the vector $\mathbf{p}_0$. When all groups of features are *a priori* believed to be equally relevant for prediction, each $p_{0,g}$ with $g = 1,\ldots,G$ can be set equal to a constant value $p_0$, which indicates the fraction of groups initially expected to be relevant for prediction. Finally, we note that when all groups of features are of size one, (3) reduces to the standard spike-and-slab prior described by George and McCulloch (1997).

### 2.1 Inference, Prediction and Relevant Groups

Given the observed data $\mathbf{X}$ and $\mathbf{y}$, we make inference about the potential values of $\mathbf{w}$ and $\mathbf{z}$ using Bayes' theorem:

$$\mathcal{P}(\mathbf{w},\mathbf{z}|\mathbf{y},\mathbf{X}) = \frac{\mathcal{P}(\mathbf{y}|\mathbf{w},\mathbf{X})\mathcal{P}(\mathbf{w}|\mathbf{z})\mathcal{P}(\mathbf{z})}{\mathcal{P}(\mathbf{y}|\mathbf{X})}, \tag{5}$$

where $\mathcal{P}(\mathbf{y}|\mathbf{X})$ is a normalization constant, known as the model evidence, which is useful for model comparison (Bishop, 2006; MacKay, 2003). This posterior distribution and the likelihood (2) can be combined to compute a predictive distribution for the target $y_{\text{new}} \in \mathbb{R}$ associated to a new observation $\mathbf{x}_{\text{new}}$:

$$\mathcal{P}(y_{\text{new}}|\mathbf{x}_{\text{new}},\mathbf{y},\mathbf{X}) = \sum_{\mathbf{z}} \int \mathcal{P}(y_{\text{new}}|\mathbf{w},\mathbf{x}_{\text{new}})\mathcal{P}(\mathbf{w},\mathbf{z}|\mathbf{y},\mathbf{X})\, d\mathbf{w}. \tag{6}$$

The posterior distribution of $\mathbf{z}$ defines the probability of the features contained in specific relevant groups to be used for prediction:

$$\mathcal{P}(\mathbf{z}|\mathbf{y},\mathbf{X}) = \int \mathcal{P}(\mathbf{w},\mathbf{z}|\mathbf{y},\mathbf{X})\, d\mathbf{w}. \tag{7}$$

Furthermore, one can marginalize (7) over $z_{g'}$, with $g' \neq g$, for a specific latent variable $z_g$ to compute $\mathcal{P}(z_g|\mathbf{y},\mathbf{X})$, that is, the associated posterior probability of using the $g$-th group of features for prediction.

A practical difficulty is that the exact computation of (5), (6) and (7) is intractable for typical learning problems. All these expressions involve a number of summations which grows exponentially with $G$, which is typically of the same order as $d$. Thus, they must be approximated in practice. Approximate Bayesian inference is typically implemented in the literature using Markov chain Monte Carlo techniques (Gibbs sampling, in particular) where one samples from a Markov chain whose stationary distribution coincides with the posterior distribution of the model (Neal, 1993). Unfortunately, these methods are computationally expensive since the Markov chain has to be run for a large number of iterations to get just a few independent samples. As a more efficient alternative we employ here expectation propagation (EP), a method for fast approximate inference with Bayesian models (Minka, 2001). This method is described in the next section.

## 3. Expectation Propagation for Bayesian Group Feature Selection

Expectation propagation (EP) is a deterministic method for carrying out approximate Bayesian inference (Minka, 2001). EP approximates the posterior distribution of the parameters of interest using a simpler parametric distribution $Q$. The form of $Q$ is chosen so that the integrals required to calculate expected values and marginal distributions with respect to $Q$ can be obtained analytically in closed form. EP fixes the parameters of $Q$ to approximately minimize the Kullback-Leibler divergence between the exact posterior and $Q$. As a side effect, EP also provides an estimate of the model evidence, which can be useful to perform model selection.

In many probabilistic models that assume i.i.d. observations the joint probability can be expressed as a product of several factors. In the specific case of a linear regression model, the joint probability of $\mathbf{w}$, $\mathbf{z}$ and $\mathbf{y}$ conditioned to $\mathbf{X}$ can be written as the product of 3 factors:

$$\mathcal{P}(\mathbf{w},\mathbf{z},\mathbf{y}|\mathbf{X}) = \mathcal{P}(\mathbf{y}|\mathbf{w},\mathbf{X})\mathcal{P}(\mathbf{w}|\mathbf{z})\mathcal{P}(\mathbf{z}) = \prod_{i=1}^{3} f_i(\mathbf{w},\mathbf{z})\,, \tag{8}$$

where the first factor corresponds to the likelihood, the second factor corresponds to the prior for $\mathbf{w}$, and the final factor corresponds to the prior for $\mathbf{z}$. Namely,

$$f_1(\mathbf{w},\mathbf{z}) = \mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w},\sigma_0^2\mathbf{I})\,, \tag{9}$$

$$f_2(\mathbf{w},\mathbf{z}) = \prod_{j=1}^{d} z_{g(j)}\mathcal{N}(w_j|0,\sigma_{g(j)}^2) + (1-z_{g(j)})\delta(w_j)\,,$$

$$f_3(\mathbf{w},\mathbf{z}) = \prod_{g=1}^{G} p_{0,g}^{z_g}(1-p_{0,g})^{1-z_g}\,. \tag{10}$$

EP approximates each exact factor $f_i$ by a simpler factor $\tilde{f}_i$ such that

$$\prod_{i=1}^{3} f_i(\mathbf{w},\mathbf{z}) \approx \prod_{i=1}^{3} \tilde{f}_i(\mathbf{w},\mathbf{z})\,.$$

All approximate factors $\tilde{f}_i$ are constrained to belong to the same family of exponential distributions, but they do not have to integrate to one. Once normalized with respect to $\mathbf{w}$, and $\mathbf{z}$, (8) becomes the

exact posterior distribution (5). Similarly, the normalized product of the $\tilde{f}_i$ becomes an approximation to the posterior:

$$Q(\mathbf{w},\mathbf{z}) = \frac{1}{Z}\prod_i \tilde{f}_i(\mathbf{w},\mathbf{z}), \tag{11}$$

where $Z$ is the normalization constant which approximates $\mathcal{P}(\mathbf{y}|\mathbf{X})$. The exponential family of distributions is closed under the product operation. Therefore, $Q$ has the same simple exponential form as the $\tilde{f}_i$ and $Z$ can be readily computed. In practice, the form of $Q$ is selected first, and the $\tilde{f}_i$ are then constrained to have the same form as $Q$. For each approximate factor $\tilde{f}_i$, one considers $Q^{\setminus i} \propto Q/\tilde{f}_i$. EP iteratively updates each $\tilde{f}_i$ one by one while minimizing the Kullback-Leibler (KL) divergence between $f_i Q^{\setminus i}$ and $\tilde{f}_i Q^{\setminus i}$. The KL-divergence minimized by EP includes a correction term so that it can be applied to un-normalized distributions (Zhu and Rohwer, 1995). More precisely, each EP update step minimizes

$$\mathrm{KL}\left(f_i Q^{\setminus i} || \tilde{f}_i Q^{\setminus i}\right) = \sum_{\mathbf{z}} \int \left[ f_i Q^{\setminus i} \log \frac{f_i Q^{\setminus i}}{\tilde{f}_i Q^{\setminus i}} + \tilde{f}_i Q^{\setminus i} - f_i Q^{\setminus i} \right] d\mathbf{w}. \tag{12}$$

with respect to $\tilde{f}_i$. Note that we have omitted in (12) the dependencies of $\tilde{f}_i$, $f_i$ and $Q^{\setminus i}$ on the parameters $\mathbf{w}$ and $\mathbf{z}$ to improve the readability. Specifically, EP involves the following steps:

1. Initialize all $\tilde{f}_i$ and $Q$ to be uniform (non-informative).
2. Repeat until $Q$ converges:
   (a) Select an $\tilde{f}_i$ to refine and compute $Q^{\setminus i} \propto Q/\tilde{f}_i$.
   (b) Update $\tilde{f}_i$ to minimize $\mathrm{KL}\left(f_i Q^{\setminus i} || \tilde{f}_i Q^{\setminus i}\right)$.
   (c) Update the approximation $Q^{\mathrm{new}} \propto \tilde{f}_i Q^{\setminus i}$.
3. Evaluate $Z \approx \mathcal{P}(\mathbf{y}|\mathbf{X})$ as the integral of the product of all the approximate factors.

The optimization problem in step 2-(b) is convex with a single global optimum. The solution to this problem is found by matching the expected values of the sufficient statistics under $f_i Q^{\setminus i}$ and $\tilde{f}_i Q^{\setminus i}$ (Bishop, 2006). EP is not guaranteed to converge globally but extensive empirical evidence shows that most of the times it converges to a fixed point (Minka, 2001). Non-convergence can be prevented by *damping* the EP updates (Minka and Lafferty, 2002). Damping is a standard procedure and consists in setting

$$\tilde{f}_i = [\tilde{f}_i^{\mathrm{new}}]^{\xi} [\tilde{f}_i^{\mathrm{old}}]^{1-\xi} \tag{13}$$

in step 2-(b), where $\tilde{f}_i^{\mathrm{new}}$ is the updated factor and $\tilde{f}_i^{\mathrm{old}}$ is the factor before the update. $\xi \in [0,1]$ is a parameter which controls the amount of damping. When $\xi = 1$, the standard EP update operation is recovered. When $\xi = 0$, no update of the approximate factors occurs. In our experiments we set $\xi = 0.9$ and progressively decay its value at each iteration of EP by 1%. Such a strategy offers good practical results and EP appears to always converge to a stationary solution. Finally, when compared to other approximate inference methods, such as Monte Carlo sampling or variational inference, EP has shown good overall performances (Minka, 2001). EP is also the preferred method for approximate inference in linear models with the standard spike-and-slab prior (Hernández-Lobato, 2010).

### 3.1 The Posterior Approximation

We approximate the posterior (5) using a parametric distribution that belongs to the exponential family

$$Q(\mathbf{w},\mathbf{z}) = \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{V}) \prod_{g=1}^{G} \text{Bern}(z_g|\sigma(p_g)) \,, \tag{14}$$

where $\mathcal{N}(\cdot|\mathbf{m},\mathbf{V})$ denotes the probability density of a multivariate Gaussian with mean vector $\mathbf{m}$ and covariance matrix $\mathbf{V}$, and $\text{Bern}(\cdot|\sigma(p_g))$ denotes the probability mass function of a Bernoulli distribution with success probability $\sigma(p_g)$ and $\sigma(\cdot)$ is the sigmoid function:

$$\sigma(x) = \frac{1}{1+\exp\{-x\}} \,.$$

In (14), $\mathbf{p} = (p_1,\ldots,p_G)^{\text{T}}$, $\mathbf{m}$ and $\mathbf{V}$ are free parameters that have to be estimated by EP. The particular choice of (14) makes all the required computations tractable and offers good experimental results detailed in Section 7. The logistic function eliminates numerical under-flow or over-flow errors and simplifies the EP update operations, in a similar way as when EP is applied to a linear regression model with the standard spike-and-slab prior (Hernández-Lobato, 2010).

The approximate factors $\tilde{f}_i$ must have the same functional form as (14) but need not be normalized. Furthermore, the exact factor $f_1$ corresponding to the likelihood (2), only depends on $\mathbf{w}$. The Bernoulli part of $\tilde{f}_1$ can thus be removed:

$$\tilde{f}_1(\mathbf{w}) = \tilde{s}_1 \exp\left\{ -\frac{1}{2} (\mathbf{w}-\tilde{\mathbf{m}}_1)^{\text{T}} \tilde{\mathbf{V}}_1^{-1} (\mathbf{w}-\tilde{\mathbf{m}}_1)^{\text{T}} \right\} , \tag{15}$$

where $\tilde{\mathbf{m}}_1$, $\tilde{\mathbf{V}}_1$ and $\tilde{s}_1$ are free parameters to be estimated by EP. The second and third approximate factor $\tilde{f}_2$ and $\tilde{f}_3$ also have a special form because the corresponding exact factors $f_2$ and $f_3$ factorize with respect to each component of $\mathbf{w}$ and $\mathbf{z}$, respectively. Furthermore, $f_3$ is independent of $\mathbf{w}$ and its corresponding Gaussian part can be ignored. These approximate factors are hence defined as:

$$\tilde{f}_2(\mathbf{w},\mathbf{z}) = \tilde{s}_2 \left[ \prod_{j=1}^{d} \exp\left\{ -\frac{1}{2\tilde{v}_{2,j}} (w_j - \tilde{m}_{2,j})^2 \right\} \text{Bern}(z_{g(j)}|\sigma(\tilde{p}_{2,j})) \right] , \tag{16}$$

$$\tilde{f}_3(\mathbf{z}) = \tilde{s}_3 \left[ \prod_{g=1}^{G} \text{Bern}(z_g|\sigma(\tilde{p}_{3,g})) \right] , \tag{17}$$

where $\tilde{s}_2$, $\tilde{s}_3$, $\tilde{\mathbf{m}}_2 = (\tilde{m}_{2,1},\ldots,\tilde{m}_{2,d})^{\text{T}}$, $\tilde{\mathbf{v}}_2 = (\tilde{v}_{2,1},\ldots,\tilde{v}_{2,d})^{\text{T}}$, $\tilde{\mathbf{p}}_2 = (\tilde{p}_{2,1},\ldots,\tilde{p}_{2,d})^{\text{T}}$ and $\tilde{\mathbf{p}}_3 = (\tilde{p}_{3,1},\ldots,\tilde{p}_{3,G})^{\text{T}}$ are free parameters to be estimated by EP. The constants $\tilde{s}_1$, $\tilde{s}_2$ and $\tilde{s}_3$ are introduced to make sure that $\tilde{f}_i Q^{\setminus i}$ and the corresponding $f_i Q^{\setminus i}$ integrate up to the same value.

Once the different parameters of $\tilde{f}_1$, $\tilde{f}_2$ and $\tilde{f}_3$ have been estimated, the corresponding parameters $\mathbf{m}$, $\mathbf{V}$ and $\mathbf{p}$ of $Q$ can be easily computed by using (11) and the closure property of the exponential family under the product operation. Namely, the product of two un-normalized Gaussian distributions is another un-normalized Gaussian distribution. Similarly, the product of two Bernoulli distributions is another Bernoulli distribution. The specific details of the product rules for Gaussian

and Bernoulli distributions are described in the Appendix of Hernández-Lobato (2009). The parameters of $Q$ given each $\tilde{f}_i$ are obtained from applying those rules:

$$\mathbf{V} = \left(\tilde{\mathbf{V}}_1^{-1} + \mathbf{\Lambda}^{-1}\right)^{-1}, \tag{18}$$

$$\mathbf{m} = \mathbf{V}\left(\tilde{\mathbf{V}}_1^{-1}\tilde{\mathbf{m}}_1 + \mathbf{\Lambda}^{-1}\tilde{\mathbf{m}}_2\right), \tag{19}$$

$$p_g = \sum_{g(j)=g} \tilde{p}_{2,j} + \tilde{p}_{3,g}, \quad \text{for } g = 1,\ldots,G, \tag{20}$$

where $\mathbf{\Lambda} = \text{diag}(\tilde{v}_{2,1},\ldots,\tilde{v}_{2,d})$ and $\text{diag}(a_1,\ldots,a_d)$ denotes a diagonal matrix with elements $a_1,\ldots,a_d$ in the diagonal. At first glance, the computation of $\mathbf{V}$ according to (18) requires the inversion of a $d \times d$ matrix, with $d$ equal to the dimensionality of the data. However, if $\tilde{\mathbf{V}}_1^{-1}$ is known and the regression problem satisfies $d \gg n$ (as for the regression problems we consider), the Woodbury formula provides a faster alternative with a computational cost of $O(n^2 d)$, where $n$ is the number of observed samples (the specific details are given in the next section).

### 3.2 EP Update Operations

This section details how to update each approximate factor $\tilde{f}_1$, $\tilde{f}_2$ and $\tilde{f}_3$ according to the steps 2-(a), 2-(b) and 2-(c) of the EP algorithm. These operations involve minimizing the KL divergence between $f_i Q^{\setminus i}$ and $\tilde{f}_i Q^{\setminus i}$ with respect to $\tilde{f}_i$. This problem is convex with a single global minimum which is found by setting $\tilde{f}_i$ so that the expected values of the sufficient statistics under $\tilde{f}_i Q^{\setminus i}$ and $f_i Q^{\setminus i}$ match after normalization (Bishop, 2006). To simplify the notation, we only present here the operations for $\xi = 1$, that is, when there is no damping in the EP updates. Incorporating the effect of damping in these operations is straight-forward and is omitted. In particular, the natural parameters of each approximate factor become a convex combination of the natural parameters before and after each update operation, as derived from (13). However, as shown in this section, only the approximate factor $\tilde{f}_2$ needs to be updated. The optimal parameters of $\tilde{f}_1$ and $\tilde{f}_3$ can be computed exactly and these factors need not be updated by EP. Finally, we note that the parameters $\tilde{s}_1$, $\tilde{s}_2$ and $\tilde{s}_3$ of $\tilde{f}_1$, $\tilde{f}_2$ and $\tilde{f}_3$ are only needed to compute the approximation of the marginal likelihood in step 3 of the EP algorithm. Their computation can thus be delayed until EP converges as described below.

We now describe how to compute the parameters $\tilde{\mathbf{m}}_1$ and $\tilde{\mathbf{V}}_1$ of the first approximate factor $\tilde{f}_1$, which is fairly simple. In particular, we note that the corresponding exact factor $f_1$ has a Gaussian form with respect to $\mathbf{w}$, namely the likelihood of the observed data, as described in (9). Furthermore, the form of $\tilde{f}_1$ in (15) is also Gaussian. This means that the factor $f_1$ can be approximated exactly by EP, independently of the values of the other approximate factors $\tilde{f}_2$ and $\tilde{f}_3$. We only have to set $\tilde{f}_1 = f_1$, since both $\tilde{f}_1$ and $f_1$ have the same form. Consequently $\tilde{f}_1$ needs not be re-estimated by EP through the steps 2-(a) to 2-(c) but set equal to $f_1$ at the beginning of the EP algorithm and kept constant afterwards. The parameters of $\tilde{f}_1$ are:

$$\tilde{\mathbf{V}}_1^{-1} = \frac{1}{\sigma_0^2}\mathbf{X}^{\mathrm{T}}\mathbf{X}, \qquad \tilde{\mathbf{V}}_1^{-1}\tilde{\mathbf{m}}_1 = \frac{1}{\sigma_0^2}\mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{21}$$

Note that $\tilde{\mathbf{m}}_1$ is not uniquely defined in (21) whenever $d > n$. More precisely, if $d > n$ then $\tilde{\mathbf{V}}_1^{-1}$ is not full rank, $\tilde{\mathbf{V}}_1$ does not exist and the likelihood (2) is not strictly concave. Therefore, when $d > n$, there are infinitely many vectors $\tilde{\mathbf{m}}_1$ that can be used as potential solutions and must satisfy (21). For this reason, it is better in practice to define $\tilde{f}_1$ in terms of its natural parameters $\tilde{\mathbf{V}}_1^{-1}$ and

$\tilde{\mathbf{V}}_1^{-1}\tilde{\mathbf{m}}_1$. Namely,

$$\tilde{f}_1(\mathbf{w}) = \tilde{s}_1 \exp\left\{ -\frac{1}{2}\mathbf{w}^{\mathrm{T}}\tilde{\mathbf{V}}_1^{-1}\mathbf{w} + \mathbf{w}^{\mathrm{T}}\tilde{\mathbf{V}}_1^{-1}\tilde{\mathbf{m}}_1 \right\},$$

where the constant terms can be included in $\tilde{s}_1$.

The optimal parameters $\tilde{\mathbf{p}}_3$ of the third approximate factor $\tilde{f}_3$ can be found following the same reasoning. Specifically, the exact factor $f_3$ has the same form as the approximate factor $\tilde{f}_3$, that is, a product of $G$ Bernoulli distributions, one for each component of $\mathbf{z}$. See (10) and (17). Thus $\tilde{f}_3$ is set equal to $f_3$ and not iteratively re-estimated through the EP algorithm. The corresponding parameters of $\tilde{f}_3$ are:

$$\tilde{p}_{3,g} = \sigma^{-1}(p_{0,g}), \quad \text{for } g = 1,\ldots,G, \tag{22}$$

where $\sigma^{-1}(\cdot)$ is the logit function, that is, the inverse of the sigmoid function. Namely,

$$\sigma^{-1}(x) = \log\frac{x}{1-x}.$$

In (22) $p_{0,g}$ is the prior probability of using the $g$-th group of features for prediction.

Updating the approximate factor $\tilde{f}_2$ is somewhat more complex and requires implementing all steps 2-(a) to 2-(c) of the EP algorithm. To simplify the computation, each of the $d$ components of $\tilde{f}_2$ that appear in (16) is updated in parallel, as suggested by Gerven et al. (2009). For this, we factorize $\tilde{f}_2$ as follows:

$$\tilde{f}_2(\mathbf{w},\mathbf{z}) = \prod_{j=1}^{d} \tilde{s}_{2,j}\tilde{f}_{2,j}(w_j, z_{g(j)}), \tag{23}$$

where

$$\tilde{s}_2 = \prod_{j=1}^{d}\tilde{s}_{2,j}, \qquad \tilde{f}_{2,j}(w_j, z_{g(j)}) = \exp\left\{ -\frac{1}{2\tilde{v}_{2,j}}(w_j - \tilde{m}_{2,j})^2 \right\} \mathrm{Bern}(z_{g(j)}|\sigma(\tilde{p}_{2,j})).$$

Note that in (23) each component $\tilde{f}_{2,j}$ is the product of a univariate Gaussian distribution and a Bernoulli distribution. Thus, we only need the marginal distributions of $Q$ for each component of $\mathbf{w}$ and $\mathbf{z}$ to obtain the corresponding update operations. The marginal distribution of $Q$ for each component of $\mathbf{z}$, $z_g$, with $g = 1,\ldots,G$, is a Bernoulli distribution with probability parameter $p_g$, where $p_g$ is defined in (20). Finding the means $\mathbf{m}$ and the variances $\mathrm{diag}(\mathbf{V})$ of each marginal distribution of $Q$ for each component of $\mathbf{w}$ is more difficult. In principle, we could use (18) and (19) for computing $\mathrm{diag}(\mathbf{V})$ and $\mathbf{m}$, respectively. However, such a computation would require inverting a $d \times d$ matrix, where $d$ is the number of dimensions. The Woodbury formula offers an efficient alternative when $d \gg n$, with a computational cost in $O(n^2 d)$:

$$\mathbf{V} = \mathbf{\Lambda} - \mathbf{\Lambda}\mathbf{X}^{\mathrm{T}}\left[\mathbf{I}\sigma_0^2 + \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{\mathrm{T}}\right]^{-1}\mathbf{X}\mathbf{\Lambda}, \tag{24}$$

where $\mathbf{\Lambda}$ is the diagonal matrix defined in (18) and $\mathbf{I}\sigma_0^2 + \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{\mathrm{T}}$ is a $n \times n$ matrix. Given this representation for $\mathbf{V}$, the value of the mean parameter $\mathbf{m}$ of $Q$ can be computed using (19) and (21) in $O(n^2 d)$ steps. Namely,

$$\mathbf{m} = \mathbf{\Lambda}\boldsymbol{\eta} - \mathbf{\Lambda}\mathbf{X}^{\mathrm{T}}\left[\mathbf{I}\sigma_0^2 + \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{\mathrm{T}}\right]^{-1}\mathbf{X}\mathbf{\Lambda}\boldsymbol{\eta}, \tag{25}$$

where $\boldsymbol{\eta} = \mathbf{X}^T\mathbf{y}/\sigma_0^2 + \boldsymbol{\Lambda}^{-1}\tilde{\mathbf{m}}_2$ is a $d$-dimensional column vector. The $d \times d$ components of $\mathbf{V}$ need not be evaluated but only the diagonal of this matrix for the variances of the marginals. Consequently the cost of computing $\text{diag}(\mathbf{V}) = (V_{11}, \ldots, V_{dd})^{\text{T}}$ and $\mathbf{m}$ is in $O(n^2 d)$, which is linear in the dimensionality $d$ of the regression problem. The use of the Woodbury formula may lead to numerical instabilities, which however have never been observed for the regression problems considered in Section 7.

Given $\text{diag}(\mathbf{V})$ and $\mathbf{m}$, we compute for each $\tilde{f}_{2,j}$ the corresponding parameters of the marginal distributions of $w_j$ and $z_{g(j)}$ under $Q^{\setminus 2,j} \propto Q/\tilde{f}_{2,j}$. These parameters are obtained using the rules for the quotient between Gaussian distributions and the quotient between Bernoulli distributions. These rules are described in the Appendix of Hernández-Lobato (2009). Consider $m_j^{\setminus 2,j}$ and $v_j^{\setminus 2,j}$ to be respectively the mean and the variance of the marginal distribution of $w_j$ under $Q^{\setminus 2,j}$. Similarly, let $\sigma(p_g^{\setminus 2,j})$, with $g = g(j)$, be the probability of using the $g$-th group for prediction under $Q^{\setminus 2,j}$. These parameters are obtained from $\text{diag}(\mathbf{V})$, $\mathbf{m}$ and the parameters of $\tilde{f}_{2,j}$ as follows:

$$
\begin{aligned}
v_j^{\setminus 2,j} &= \left(V_{jj}^{-1} - \tilde{v}_{2,j}^{-1}\right)^{-1}, \\
m_j^{\setminus 2,j} &= v_j^{\setminus 2,j}\left(V_{jj}^{-1} m_j - \tilde{v}_{2,j}^{-1}\tilde{m}_{2,j}\right), \\
p_g^{\setminus 2,j} &= p_g - \tilde{p}_{2,j},
\end{aligned}
\tag{26}
$$

where $V_{jj}$ and $m_j$ are respectively the variance and the mean of the marginal distribution of $w_j$ under $Q$.

Once $Q^{\setminus 2,j}$ has been computed, we find the corresponding approximate factor $\tilde{f}_{2,j}$ which minimizes the KL divergence between $f_{2,j}Q^{\setminus 2,j}$ and $\tilde{f}_{2,j}Q^{\setminus 2,j}$, where $f_{2,j}$ is obtained from a factorization of $f_2$ equivalent to the one described for $\tilde{f}_2$ in (23). Specifically, $f_{2,j}(w_j, z_{g(j)}) = z_{g(j)}\mathcal{N}(w_j|0,\sigma_{g(j)}^2) + (1 - z_{g(j)})\delta(w_j)$. Given $Q^{\setminus 2,j}$ the optimal parameters of $\tilde{f}_{2,j}$ are:

$$
\begin{aligned}
\tilde{v}_{2,j}^{\text{new}} &= \frac{1}{a_j^2 - b_j} - v_j^{\setminus 2,j}, \\
\tilde{m}_{2,j}^{\text{new}} &= m_j^{\setminus 2,j} + \frac{a_j}{a_j^2 - b_j}, \\
\tilde{p}_{2,j}^{\text{new}} &= \log \mathcal{N}(0|m_j^{\setminus 2,j}, v_j^{\setminus 2,j} + v_0) - \log \mathcal{N}(0|m_j^{\setminus 2,j}, v_j^{\setminus 2,j}),
\end{aligned}
$$

where $v_0$ is the marginal variance of the slab and $a_j$ and $b_j$ are constants defined as:

$$
\begin{aligned}
a_j &= \sigma\left(\tilde{p}_{2,j}^{\text{new}} + p_g^{\setminus 2,j}\right)\frac{m_j^{\setminus 2,j}}{v_j^{\setminus 2,j} + v_0} + \sigma\left(-\tilde{p}_{2,j}^{\text{new}} - p_g^{\setminus 2,j}\right)\frac{m_j^{\setminus 2,j}}{v_j^{\setminus 2,j}}, \\
b_j &= \sigma\left(\tilde{p}_{2,j}^{\text{new}} + p_g^{\setminus 2,j}\right)\frac{(m_j^{\setminus 2,j})^2 - v_j^{\setminus 2,j} - v_0}{(v_j^{\setminus 2,j} + v_0)^2} + \sigma\left(-\tilde{p}_{2,j}^{\text{new}} - p_g^{\setminus 2,j}\right)\frac{(m_j^{\setminus 2,j})^2 - v_j^{\setminus 2,j}}{(v_j^{\setminus 2,j})^2}.
\end{aligned}
$$

These update operations require the value of $v_j^{\setminus 2,j}$ to be positive. In some rare situations a negative value is found for $v_j^{\setminus 2,j}$ after removing $\tilde{f}_{2,j}$ from the posterior approximation $Q$ to compute $Q^{\setminus 2,j}$. In such a rare occurrence, the corresponding update of $\tilde{f}_{2,j}$ is not performed. Furthermore, a negative value for $\tilde{v}_{2,j}^{\text{new}}$ may be observed when computing the optimal parameters for $\tilde{f}_{2,j}$. Negative

variances are common in many EP implementations (Minka, 2001; Minka and Lafferty, 2002). In this case, the approximate factors are not un-normalized density functions, but correction factors that compensate for errors in the first approximate factor $\tilde{f}_1$. It has been observed in the literature that negative variances can lead to erratic behavior in EP and to longer convergence times (Seeger, 2008; Hernández-Lobato, 2010). To avoid these problems, we minimize the KL between $f_{2,j}Q^{\backslash 2,j}$ and $\tilde{f}_{2,j}Q^{\backslash 2,j}$ with the constraint of $\tilde{v}_{2,j}^{\text{new}}$ being positive. In this case, whenever the optimal $\tilde{v}_{2,j}^{\text{new}}$ is negative, we simply set $\tilde{v}_{2,j}^{\text{new}} = v_\infty$, where $v_\infty$ is a large positive constant. The update of the other parameters of $\tilde{f}_{2,j}$ ($\tilde{m}_{2,j}$ and $\tilde{p}_{2,j}$) is kept unchanged. This approach, already used in the linear regression model with standard spike-and-slab priors described by Hernández-Lobato (2010), offered improved convergence results.

Once all the approximate factors $\tilde{f}_{2,j}$, with $j = 1,\ldots,d$, have been updated in parallel, $Q$ needs to be recomputed as the product of all the approximate factors. This corresponds to step 2-(c) of the EP algorithm. For this, we can use (18), (19) and (20). However, we have already described how to obtain the means $\mathbf{m}$ and the variances $\text{diag}(\mathbf{V})$ of the marginals of $Q$ for $\mathbf{w}$ to update each approximate factor $\tilde{f}_{2,j}$. Thus, in practice, one only has to use (20) to recompute the parameter $\mathbf{p}$ of $Q$, which is needed in (26). There is no need to recompute the complete covariance matrix $\mathbf{V}$ of the Gaussian part of $Q$ since only the diagonal of this matrix is strictly needed for the EP updates. In summary, the total cost of the EP algorithm under the assumption of a constant number of iterations until convergence is in $O(n^2 d)$, where $n$ is the number of training instances and $d$ is the number of features.

### 3.3 Approximation of the Model Evidence

The Bayesian approach to model selection specifies that the model with the largest evidence should be preferred (Bishop, 2006; MacKay, 2003). The model evidence is defined in (5) as $\mathcal{P}(\mathbf{y}|\mathbf{X})$, that is, the normalization constant used to compute the posterior distribution from the joint distribution of the model parameters and the data. It can also be described as the probability that the targets $\mathbf{y}$ are generated from the design matrix $\mathbf{X}$ using a linear model whose coefficient vector $\mathbf{w}$ is randomly sampled from the assumed prior distribution. The model evidence naturally achieves a balance between penalizing model complexity and rewarding models that provide a good fit to the training data (Bishop, 2006). However, a practical difficulty in computing $\mathcal{P}(\mathbf{y}|\mathbf{X})$ is its expensive computational cost. Specifically, the exact evaluation of $\mathcal{P}(\mathbf{y}|\mathbf{X})$ is infeasible for large $G$ since it involves a sum of $2^G$ terms. These are the $2^G$ different configurations for the vector of latent variables $\mathbf{z}$. Nevertheless, if needed, EP can be used to efficiently compute an approximation once it has converged, as described in step 3 of the algorithm:

$$\mathcal{P}(\mathbf{y}|\mathbf{X}) \approx \int \sum_{\mathbf{z}} \tilde{f}_1(\mathbf{w},\mathbf{z})\tilde{f}_2(\mathbf{w},\mathbf{z})\tilde{f}_3(\mathbf{w},\mathbf{z})d\mathbf{w}\,. \tag{27}$$

Since all the approximate factors $\tilde{f}_1$, $\tilde{f}_2$ and $\tilde{f}_3$ have simple exponential forms, (27) can be readily evaluated. We only have to use the formulas for the product of Gaussian and Bernoulli distributions. These formulas are described in the Appendix of Hernández-Lobato (2009). The evaluation of (27) also requires the computation of the parameters $\tilde{s}_1$, $\tilde{s}_2$ and $\tilde{s}_3$ of each approximate factor. These parameters are estimated once EP has converged and their specific values are fixed to guarantee that

$\tilde{f}_i Q^{\setminus i}$ and the corresponding $f_i Q^{\setminus i}$ integrate up to the same value:

$$\begin{aligned}
\log(\tilde{s}_1) &= -\frac{n}{2}\log(2\pi\sigma_0^2) - \frac{1}{2\sigma_0^2}\mathbf{y}^{\mathrm{T}}\mathbf{y}\,, \\
\log(\tilde{s}_{2,j}) &= \log(\kappa_j) - \log\left(\sigma(\tilde{p}_{2,j})\sigma(p_{g(j)}^{\setminus 2,j}) + \sigma(-\tilde{p}_{2,j})\sigma(-p_{g(j)}^{\setminus 2,j})\right) - \frac{1}{2}\log\left(2\pi\tilde{v}_{2,j}\right) \\
&\quad - \log\left(\mathcal{N}(0|m_j^{\setminus 2,j} - \tilde{m}_{2,j}, \tilde{v}_{2,j} + V_{jj}^{\setminus 2,j})\right)\,, \quad \text{for } j = 1,\ldots,d \\
\log(\tilde{s}_3) &= 0\,,
\end{aligned}$$

were $\kappa_j = \sigma(p_{g(j)}^{\setminus 2,j})\mathcal{N}(0|m_j^{\setminus 2,j}, v_0 + V_{jj}^{\setminus 2,j}) + \sigma(-p_{g(j)}^{\setminus 2,j})\mathcal{N}(0|m_j^{\setminus 2,j}, V_{jj}^{\setminus 2,j})$ and $\tilde{s}_2 = \prod_{j=1}^d \tilde{s}_{2,j}$ since we have further factorized $\tilde{f}_2$ as the product of $d$ factors $\tilde{f}_{2,j}$ with $j = 1,\ldots,d$. See (23) for further details.

Given these values, we can approximate the logarithm of the model evidence by

$$\begin{aligned}
\log \mathcal{P}(\mathbf{y}|\mathbf{X}) \approx \sum_{i=1}^{3}\log\tilde{s}_i + \sum_{g=1}^{G}\log\left(\sigma(\tilde{p}_{3,g})\prod_{g(j)=g}\sigma(\tilde{p}_{2,j}) + \sigma(-\tilde{p}_{3,g})\prod_{g(j)=g}\sigma(-\tilde{p}_{2,j})\right) \\
- \frac{1}{2}\tilde{\mathbf{m}}_2^{\mathrm{T}}\boldsymbol{\Lambda}^{-1}\tilde{\mathbf{m}}_2 + \frac{d}{2}\log(2\pi) + \frac{1}{2}\log(|\mathbf{V}|) + \frac{1}{2}\boldsymbol{\upsilon}^{\mathrm{T}}\mathbf{V}\boldsymbol{\upsilon}\,, \qquad (28)
\end{aligned}$$

where $\mathbf{V}$ is the covariance matrix of the Gaussian part of $Q$ and $\boldsymbol{\upsilon}$ is a $d$-dimensional vector defined as $\boldsymbol{\upsilon} = 1/\sigma_0^2\mathbf{X}^{\mathrm{T}}\mathbf{y} + \boldsymbol{\Lambda}^{-1}\tilde{\mathbf{m}}_2$. In practice it is better to work with the logarithm of the approximation to $\mathcal{P}(\mathbf{y}|\mathbf{X})$ to avoid numerical over-flow and under-flow errors. Furthermore, the computation of $|\mathbf{V}|$ can be efficiently implemented when $d \gg n$ using the Sylvester's determinant formula and $\boldsymbol{\upsilon}^{\mathrm{T}}\mathbf{V}\boldsymbol{\upsilon}$ can be evaluated in $O(n^2 d)$ steps using the representation for $\mathbf{V}$ given in (24).

We note that one should use the model evidence with care to perform model selection. In particular, if the assumptions made about the form of the model are not accurate enough, the results obtained by Bayesian model comparison can be misleading, as indicated by Bishop (2006). This is precisely the case of the experiments reported in Section 7.2, where we have observed that the approximation to the model evidence (28) provides sub-optimal decisions to choose the model hyper-parameters $p_0$ and $v_0$. Thus, in a practical application it is wise to keep aside an independent validation set to evaluate the overall performance of the final system.

### 3.4 Prediction and Identification of Relevant Groups

Once EP has converged and $Q$ has been estimated, we can use this approximation for making predictions. In particular, we only have to substitute the posterior approximation in (6) to obtain an approximate predictive distribution for the target $y_{\text{new}}$ associated to a new instance $\mathbf{x}_{\text{new}}$:

$$\begin{aligned}
\mathcal{P}(y_{\text{new}}|\mathbf{x}_{\text{new}}, \mathbf{y}, \mathbf{X}) &\approx \sum_{\mathbf{z}}\int \mathcal{P}(y_{\text{new}}|\mathbf{w}, \mathbf{x}_{\text{new}})\,Q(\mathbf{w}, \mathbf{z})\,d\mathbf{w} \\
&= \mathcal{N}(y_{\text{new}}|\mathbf{m}^{\mathrm{T}}\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{new}}^{\mathrm{T}}\mathbf{V}\mathbf{x}_{\text{new}} + \sigma_0^2)\,, \qquad (29)
\end{aligned}$$

where $\mathbf{m}$ and $\mathbf{V}$ are the mean vector and the covariance matrix of the posterior approximation $Q$ for $\mathbf{w}$. Both $\mathbf{x}_{\text{new}}^{\mathrm{T}}\mathbf{V}\mathbf{x}_{\text{new}}$ and $\mathbf{m}^{\mathrm{T}}\mathbf{x}_{\text{new}}$ can be efficiently computed in $O(n^2 d)$ steps using the representations given in (24) and (25), respectively. Finally, if one is only interested in the expected value of $y_{\text{new}}$ and not in the uncertainty of the prediction, the computation of $\mathbf{x}_{\text{new}}^{\mathrm{T}}\mathbf{V}\mathbf{x}_{\text{new}}$ can be omitted.

The posterior approximation $Q$ is also very useful to identify the groups of features that are more relevant for prediction after substituting the exact posterior by $Q$ in (7). Namely,

$$\mathcal{P}(\mathbf{z}|\mathbf{y},\mathbf{X}) \approx \int Q(\mathbf{w},\mathbf{z})\, d\mathbf{w} = \prod_{g=1}^{G} \text{Bern}(z_g|\sigma(p_g))\,,$$

where $\sigma(p_g)$ approximates the posterior probability of using the $g$-th group of features for prediction. Thus, we can use the parameters $p_1,\ldots,p_G$ of $Q$ to identify the groups which are more likely used for prediction. More precisely, we should observe a bi-separation of the different groups in two sets according to these parameters. The first set would contain features which are unlikely to be used for prediction. By contrast, the second set would contain features which are used for prediction with high posterior probability.

## 4. Sequential Experimental Design

Sequential experimental design (also known in the literature as optimal design or active learning) deals with the problem of saving on expensive experiments to obtain the highest level of information about the different latent variables or parameters of the assumed model (Seeger, 2008; Chaloner and Verdinelli, 1995; Fedorov, 1972). In our particular scenario, sequential experimental design tries to answer the following problem. Consider several candidate points $\mathbf{x}_{\text{new}}$ that are available for inclusion into the training set of the model. At which of these points should the corresponding target value $y_{\text{new}}$ be sampled to obtain as much new information as possible about the unknown $\mathbf{w}$? Assuming that $\mathbf{x}_{\text{new}}$ and $y_{\text{new}}$ are both known, Seeger (2008) and MacKay (1991) describe some natural scores that can be used to answer this question: for example, the decrease in the posterior uncertainty or the gain in information from the current posterior distribution $\mathcal{P}(\mathbf{w}|\mathbf{y},\mathbf{X})$ to the updated posterior distribution $\mathcal{P}'(\mathbf{w}|\mathbf{y},\mathbf{X})$ that is obtained once the new data instance is introduced in the training set. Information gain can be measured in terms of the KL divergence between these two distributions. Namely, we could aim to maximize $\text{KL}(\mathcal{P}'||\mathcal{P})$. The target value $y_{\text{new}}$ is however often not available for the candidate points $\mathbf{x}_{\text{new}}$. A natural alternative is obtained by marginalizing this score over the expected distribution of $y_{\text{new}}$ given the assumed model. Such a score is defined as $\mathbb{E}\{\text{KL}(\mathcal{P}'||\mathcal{P})\}$, where the expectation goes over $\mathcal{P}(y_{\text{new}}|\mathbf{x}_{\text{new}},\mathbf{y},\mathbf{X})$, that is, the predictive distribution for $y_{\text{new}}$ given the current model. Another potential score to be used for this purpose is the expected decrease in the entropy of the posterior distribution once the new instance has been included in the training set. That is, $\mathbb{E}\{H[\mathcal{P}] - H[\mathcal{P}']\}$, where $H[\mathcal{P}]$ and $H[\mathcal{P}']$ respectively denote the entropy of the posterior distribution before and after the inclusion of the candidate point $\mathbf{x}_{\text{new}}$ into the training set. MacKay (1991) actually shows that both scores are equivalent and lead to the selection of the same instance $\mathbf{x}_{\text{new}}$. Consequently we focus on this last score in the rest of this section. Such score has also been used by Ji and Carin (2007) and by Seeger (2008) to perform sequential experiment design using a sparse linear model and by Lawrence et al. (2003) in the context of sparse Gaussian processes.

As discussed in Section 2.1, the computation of the exact posterior distribution $\mathcal{P}(\mathbf{w}|\mathbf{y},\mathbf{X})$ is intractable in practice. Thus, we have to resort to the EP posterior approximation $Q$ for the estimation of the entropy of the posterior distribution. More precisely, we replace the entropy of the exact posterior $H[\mathcal{P}]$ by the entropy of the EP approximation $H[Q]$. The score to maximize is now defined as $\mathbb{E}\{H[Q] - H[Q']\}$, where $H[Q]$ and $H[Q']$ respectively denote the entropy of $Q$ before and after the inclusion of the candidate point $\mathbf{x}_{\text{new}}$ into the training set.

Assume we would like to score a new instance $\mathbf{x}_{\text{new}}$. Further consider that $Q$ has been marginalized over the latent variables $\mathbf{z}$, which means that only the Gaussian part of $Q$ remains. Under these assumptions, the logarithm of the entropy of $Q$ is:

$$\log H[Q] = -\frac{1}{2}\log\left(\left|\frac{1}{\sigma_0^2}\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{\Lambda}^{-1}\right|\right) + C \tag{30}$$

where $C$ summarizes some constants that are independent of $\mathbf{x}_{\text{new}}$, and $\mathbf{\Lambda}$ is the diagonal matrix defined in (20). Once $\mathbf{x}_{\text{new}}$ has been included in the training set, the logarithm of the entropy of the updated posterior approximation $Q'$ is:

$$\log H[Q'] = \log H[Q] - \frac{1}{2}\log\left(1 + \frac{1}{\sigma_0^2}\mathbf{x}_{\text{new}}^{\mathrm{T}}\mathbf{V}\mathbf{x}_{\text{new}}\right) + C\,. \tag{31}$$

where we have used the Sylvester's determinant theorem and $\mathbf{V}$ is the covariance matrix of the Gaussian part of $Q$, that is, the posterior approximation before the update. Furthermore, in (31) we have made the assumption that the parameters $\tilde{\mathbf{v}}_2$ of the approximate factor $\tilde{f}_2$ in $Q$ (the diagonal entries of the matrix $\mathbf{\Lambda}$) are constant when the candidate point $\mathbf{x}_{\text{new}}$ is included in the training set. Of course, this assumption need not be satisfied in practice and EP has to be run to find the updated parameters $\tilde{\mathbf{v}}_2$ of $\tilde{f}_2$ in $Q'$. However, running the EP algorithm each time a candidate point has to be scored would be very expensive. A simple alternative, used in (31), keeps the approximate factor $\tilde{f}_2$ constant and lets $\tilde{f}_1$ vary in $Q'$. In other words, for the purpose of scoring new candidate points we treat the model as purely linear-Gaussian. Doing so lets us compute the score very efficiently and many candidate points can be evaluated. The same approximation has been used by Seeger (2008) in a linear regression model with Laplace priors. Note that (30) does not depend on the target $y_{\text{new}}$ associated to $\mathbf{x}_{\text{new}}$. This means that $H[Q]$ is independent of $y_{\text{new}}$ and can be taken out from the expectation with respect to $\mathcal{P}(y_{\text{new}}|\mathbf{x}_{\text{new}}, \mathbf{y}, \mathbf{X})$. The same applies to $H[Q']$ under the assumptions described. Thus, for the purpose of scoring candidate points, we can ignore the expectation over $\mathcal{P}(y_{\text{new}}|\mathbf{x}_{\text{new}}, \mathbf{y}, \mathbf{X})$.

Since $H[Q]$ is constant for all candidate points to be scored, expression (31) indicates that the candidate point $\mathbf{x}_{\text{new}}$ to be included in the model is the one that maximizes $\mathbf{x}_{\text{new}}^{\mathrm{T}}\mathbf{V}\mathbf{x}_{\text{new}}$. This is precisely the term which specifies the uncertainty in the prediction of the target value $y_{\text{new}}$ associated to the instance $\mathbf{x}_{\text{new}}$ under the current model. See (29) for further details. In other words, those points for which the model is the most unsure about their target value are preferred. These points are expected to be the most informative. However, in practice, the candidate points can also be unknown. In such a situation, the optimal candidate point $\mathbf{x}_{\text{new}}$ to include in the model is a vector parallel to the eigenvector of the covariance matrix $\mathbf{V}$ with the largest associated eigenvalue. Such vector can be efficiently found, for example, using the power method. The total computational cost of the power method under the assumption of a constant number of iterations until convergence is $O(n^2 d)$ when $d \gg n$. For guaranteeing such computational cost, the algorithm must be efficiently implemented using the representation for $\mathbf{V}$ given in (24). Once $\mathbf{x}_{\text{new}}$ has been found, we can carry out the required experiments to measure the associated target value $y_{\text{new}}$. This is precisely the procedure which is followed in adaptive compressed sensing experiments, where a sparse signal (the model coefficients $\mathbf{w}$) is reconstructed from a small number of sequentially designed measurements (Seeger, 2008; Ji and Carin, 2007). The model described here also includes prior knowledge about groups of components of the sparse signal which are expected to be jointly equal to zero or jointly different from zero. Section 7 shows that the inclusion of this prior knowledge in the inductive procedure leads to improved reconstruction errors.

## 5. Related Methods for Group Feature Selection

In this section we review other methods that are available in the literature to perform feature selection at the group level. Some of these techniques involve a complete Bayesian approach, similar to the one described in this document, but they use Markov chain Monte Carlo sampling techniques instead of EP to approximate the posterior distribution of $\mathbf{w}$ given the observed data (Bishop, 2006; MacKay, 2003). Other techniques do not necessarily proceed this way and specify instead a particular objective function which is optimized. This function includes an error loss and a set of constraints to enforce sparsity at the group level. Finally, other techniques follow a type-II maximum likelihood approach (Bishop, 2006) in which the model evidence is optimized with respect to some hyper-parameters to enforce sparsity at the group level.

### 5.1 The Group LASSO

We start by reviewing the group LASSO (Yuan and Lin, 2006; Kim et al., 2006), which is probably the most popular method employed for group feature selection. This method is a natural extension of the LASSO (Tibshirani, 1996) and consists in estimating a linear predictor by minimizing a squared loss error function evaluated on the observed data, under a series of constraints which enforce sparsity at the group level. Unlike the approach described in this document, this method does not provide a posterior probability distribution for $\mathbf{w}$, but a point estimate. Specifically, the estimator for $\mathbf{w}$ in the group LASSO is:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \quad \mathcal{L}(\mathbf{w}) \qquad\qquad \text{s.t.} \quad \sum_{g=1}^{G} s(d_g)||\mathbf{w}_g|| \leq k\,, \tag{32}$$

where $\mathcal{L}$ is a convex loss function evaluated on the training data, for example, the squared loss defined as $\sum_{i=1}^{n}(y_i - \mathbf{w}^{\mathrm{T}}\mathbf{x}_i)^2$; $\mathbf{w}_g$ is a vector that contains the components of $\mathbf{w}$ within the $g$-th group; $d_g$ is the dimension of the vector $\mathbf{w}_g$; $s(\cdot)$ is a scaling function used to account for groups of different sizes; $||\mathbf{w}_g||$ is a norm of the vector $\mathbf{w}_g$; and $k$ is a positive regularization parameter. Besides the squared loss, other authors have also considered a logistic regression loss to address classification problems (Meier et al., 2008). The norm that penalizes each vector $\mathbf{w}_g$ is typically the $\ell_2$-norm, although the $\ell_\infty$-norm has also been considered by Vogt and Roth (2010). The function $s(\cdot)$ is often set to be the square root (Meier et al., 2008). The group LASSO has been shown to be asymptotically consistent under certain conditions (Bach, 2008; Meier et al., 2008). However, when $d \gg n$, the minimizer of (32) may not be unique (Roth and Fischer, 2008; Vogt and Roth, 2010). The level of sparsity in the group LASSO is determined by the regularization parameter $k$. The smaller the value of $k$ the sparser the solution at the group level and vice-versa. The optimal value of $k$ is specific to the problem under consideration. Typically, it is fixed by minimizing an independent estimate of the generalization error obtained by cross-validation. An optimal solution to (32) can be obtained using the efficient algorithm described by Roth and Fischer (2008). Finally, there exists an equivalent formulation of the group LASSO where the optimization problem is un-constrained, but the loss function is penalized by the sum of the $\ell_2$-norms of the group components of $\mathbf{w}$ (Yuan and Lin, 2006). Namely, (32) is equivalent to

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \quad \mathcal{L}(\mathbf{w}) + \gamma \sum_{g=1}^{G} s(d_g)||\mathbf{w}_g||\,. \tag{33}$$

for some $\gamma > 0$, which plays an opposite role to $k$ in (32). In particular, the larger the value of $\gamma$, the sparser the solution is at the group level.

The group LASSO has proven to be useful in many domain applications (Kim et al., 2006; Meier et al., 2008; Roth and Fischer, 2008). Nevertheless, this method suffers from the problem of finding meaningful variance and covariance estimates for the regression coefficients $\mathbf{w}$, as described by Raman et al. (2009). These estimates could be easily obtained by using the formulation given in (33) to compute the Hessian at the optimal solution. This is, for example, the approach followed by the Laplace approximation to perform approximate Bayesian inference (MacKay, 2003). Unfortunately, the objective function in (33) is not differentiable at the optimum as a consequence of the regularization term which enforces sparsity at the group level. This means that the Hessian is un-defined at the optimum and the variance and covariance estimates of $\mathbf{w}$ cannot be computed in practice. In the case of the model described in this document these estimates can be very useful, as reported in Section 7, to perform sequential experimental design. Finally, another drawback of the group LASSO is that, as such, it does not allow to favor the selection of specific groups of features that are *a priori* believed to be more relevant. In the model described in this document, the inclusion of this type of prior knowledge is very easy by specifying different values for $p_{0,g}$, with $g = 1, \ldots, G$, in the prior distribution (4) for $\mathbf{z}$.

### 5.2 The Bayesian Group LASSO

The Bayesian group LASSO is proposed in Raman et al. (2009) as a full Bayesian treatment of the group LASSO to overcome the problem of covariance estimation just described. From a probabilistic perspective the group LASSO can be understood as a standard linear regression model with Gaussian noise and a product of multi-variate Laplacian priors over the regression coefficients. In particular, the target values $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$ are assumed to be generated according to $y_i = \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + \varepsilon_i$, with $\varepsilon_i \sim \mathcal{N}(0, \sigma_0^2)$, which implies a Gaussian likelihood equivalent to the one described in (2). Assume $d_g$ is the dimension of the $g$-th group of features. Consider now for each group of coefficients $\mathbf{w}_g$, with $g = 1, \ldots, G$, a multivariate and spherical $d_g$-dimensional Multi-Laplace prior, which can be expressed as a hierarchical normal-gamma model. Namely,

$$\begin{aligned}\mathcal{P}(\mathbf{w}_g) &= \int \mathcal{N}(\mathbf{w}_g|\mathbf{0}, \lambda_g^2\mathbf{I})\mathrm{Gamma}\left(\lambda_g^2 \middle| \frac{d_g+1}{2}, \frac{2}{d_g\gamma^2}\right) d\lambda_g^2 \\ &\propto \left(d_g\gamma^2\right)^{\frac{d_g}{2}} \exp\left\{-\gamma\sqrt{d_g}||\mathbf{w}_g||_2\right\},\end{aligned}$$

where $\gamma$ is a parameter which determines the degree of group sparsity, $||\cdot||_2$ represents the $\ell_2$-norm, $\lambda_g^2$ can be seen as some latent parameter and $\mathrm{Gamma}(\cdot|a,b)$ denotes a gamma distribution with shape and scale parameters $a$ and $b$, respectively. The complete prior for $\mathbf{w}$ is hence defined as $\mathcal{P}(\mathbf{w}) = \prod_{g=1}^{G} \mathcal{P}(\mathbf{w}_g)$. Consider now the posterior distribution of $\mathbf{w}$ under this likelihood and this prior distribution $\mathcal{P}(\mathbf{w}|\mathbf{y}) \propto \mathcal{P}(\mathbf{y}|\mathbf{X}, \mathbf{w}) \prod_{g=1}^{G} \mathcal{P}(\mathbf{w}_g)$. If we set $\sigma_0^2 = 1/2$, the group LASSO, as defined in (33), is obtained by maximizing the logarithm of $\mathcal{P}(\mathbf{w}|\mathbf{y})$ with respect to $\mathbf{w}$.

Instead of considering only a single point estimate of $\mathbf{w}$, the Bayesian group LASSO considers the complete posterior distribution for $\mathbf{w}$ given the observed data, under the model just described. Unfortunately, the exact computation of this distribution is intractable and closed form expressions to describe it cannot be obtained. This means that in practice one has to use approximate inference techniques. Raman et al. (2009) have proposed to use Markov chain Monte Carlo methods for this

purpose. In particular, a Gibbs sampling approach which iteratively generates samples from the conditional distributions of each coefficient $w_j$ and each latent parameter $\lambda_g^2$ from the normal-gamma prior. In this document we consider a more efficient Gibbs sampling algorithm for the Bayesian group LASSO where the model coefficients are marginalized out and we directly sample from the unconditional distribution of the latent parameters $\lambda_g^2$, with $g = 1, \ldots, G$. Once these samples have been generated, we then sample from the conditional distribution of $\mathbf{w}$ given the latent parameters. Appendix A describes the details of this algorithm. Finally, even though it successfully provides covariance estimates, the Bayesian group LASSO does not consider favoring the selection of specific groups of features.

### 5.3 The Group Horseshoe

The group horseshoe introduced in this section is a natural extension of the robust horseshoe prior initially proposed to address sparse supervised learning problems (Carvalho et al., 2009). A model incorporating the horseshoe prior can be described by assuming Gaussian noise around the target values $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$, as in the Bayesian model considered here and as in the Bayesian group LASSO. The likelihood for $\mathbf{y}$ given $\mathbf{w}$, $\mathcal{P}(\mathbf{y}|\mathbf{X}, \mathbf{w})$, is equivalent to (2). Under the horseshoe prior each component of $\mathbf{w}$, $w_j$, is assumed to be conditionally independent with a density which can be represented as a scale mixture of normals:

$$\mathcal{P}(w_j|\lambda_j, \tau) = \int \mathcal{N}(w_j|0, \lambda_j^2\tau^2)\mathcal{C}^+(\lambda_j|0,1)d\lambda_j\,, \tag{34}$$

where $\mathcal{C}^+(\cdot|0,1)$ is a half-Cauchy distribution with location and scale parameters equal to 0 and 1, respectively; $\lambda_j^2$ is a latent parameter; and $\tau$ is a shrinkage parameter which determines the level of sparsity: the smaller the value of $\tau$, the sparser the prior. We note that (34) describes a hierarchical normal-half-Cauchy model. This prior has two interesting properties which make it useful for induction under the sparsity assumption for $\mathbf{w}$. First, Cauchy-like tails allow for large values of $w_j$. Second, it has an infinitely tall spike at the origin which favors values of $w_j$ close to zero. A detailed analysis of this prior and several benchmark experiments which consider different regression problems illustrate its advantages with respect to other approaches for sparse learning (Carvalho et al., 2009).

The prior described in (34) can be easily generalized to address sparsity at the group level. For this, we only have to assume the same latent parameter $\lambda_j$ for several components of $\mathbf{w}$. Specifically, we consider for each group of coefficients $\mathbf{w}_g$, with $g = 1, \ldots, G$, a multivariate and spherical $d_g$-dimensional prior, which can be expressed as a hierarchical normal-half-Cauchy model, as in (34):

$$\mathcal{P}(\mathbf{w}_g|\lambda_g, \tau) = \int \mathcal{N}(\mathbf{w}_g|\mathbf{0}, \lambda_g^2\tau^2\mathbf{I})\mathcal{C}^+(\lambda_g|0,1)d\lambda_g\,, \tag{35}$$

where $\lambda_g$ is a latent parameter specific to each group and $\tau$ is a shrinkage parameter. The resulting prior has similar properties to the one-dimensional prior described in (34). That is, Cauchy-like tails to allow for large values of each component of $\mathbf{w}_g$ and an infinitely tall spike at the origin which favors values where all the components of $\mathbf{w}_g$ are close to zero. See Section 6 for further details on this prior.

The posterior distribution of $\mathbf{w}$ under the assumed Gaussian likelihood and the prior distribution introduced in (35) is given by $\mathcal{P}(\mathbf{w}|\mathbf{y}) \propto \mathcal{P}(\mathbf{y}|\mathbf{X}, \mathbf{w}) \prod_{g=1}^{G} \mathcal{P}(\mathbf{w}_g)$. As in the Bayesian group LASSO,

the exact computation of this distribution is intractable and, in practice, we have to resort to approximate techniques. Initially, one can think about using EP for this task. However, the specific application of the EP algorithm to this model is challenging. Specifically, the prior distribution suggested in (35) can not be evaluated exactly since it is not possible to evaluate the corresponding integral in closed form. Furthermore, this prior does not have a closed form convolution with the Gaussian distribution. This makes all the computations required by EP very difficult. Consequently we use a simpler alternative in this document. Namely, a Gibbs sampling technique similar to the one used for the Bayesian group LASSO. Appendix A explains the details of this algorithm.

### 5.4 Automatic Relevance Determination for Groups of Features

Another technique which can be used to perform feature selection at the group level is Automatic Relevance Determination (ARD) (Tipping, 2001; Ji et al., 2009). Like the other models described so far in this section, ARD also assumes a Gaussian likelihood for $\mathbf{y}$ given $\mathbf{w}$. Then, a zero mean factorizing Gaussian prior is fixed for $\mathbf{w}$. In the simplest formulation of ARD, this prior distribution has a different hyper-parameter $\alpha_j$ for each dimension of the problem (Li et al., 2002). Namely, $\mathcal{P}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{A}^{-1})$, where $\mathbf{A}$ is a diagonal matrix with entries $A_{jj} = \alpha_j$, for $j = 1, \ldots, d$. Thus, each hyper-parameter $\alpha_j$, is the inverse of the prior variance for the corresponding component $w_j$ of $\mathbf{w}$. Since both the likelihood and the prior are Gaussian under this formulation, the model evidence can be evaluated exactly. This evidence can be maximized component-wise with respect to the hyper-parameters $\alpha_j$, with $j = 1, \ldots, d$, using the fast algorithm suggested by Tipping and Faul (2003). Specifically, these authors provide a closed form solution for the optimal $\alpha_j$ while the other hyper-parameters are kept fixed. This means that one only has to iteratively optimize the model evidence with respect to each $\alpha_j$ until convergence. In such case, one typically finds that most of these hyper-parameters are driven to infinity during the optimization process. Consequently the posterior distribution of the coefficients of $\mathbf{w}$ corresponding to these hyper-parameters is set to a delta function centered at zero. Thus, this procedure can be used to induce $\mathbf{w}$ under the sparsity assumption.

Following the ARD principle sparsity at the group level can be easily obtained by considering a different hyper-parameter for each group of coefficients. Specifically, Ji et al. (2009) consider the following Gaussian prior distribution for $\mathbf{w}$:

$$\mathcal{P}(\mathbf{w}) = \prod_{g=1}^{G} \mathcal{P}(\mathbf{w}_g) = \prod_{g=1}^{G} \mathcal{N}(\mathbf{w}_g|\mathbf{0}, \alpha_g^{-1}\mathbf{I}),$$

where $\alpha_g$ is the inverse of the prior variances for each component of $\mathbf{w}_g$, that is, the vector of model coefficients within the $g$-th group. Given this prior distribution and a Gaussian likelihood such as the one described in (2), the corresponding model evidence is $\mathcal{P}(\mathbf{y}|\mathbf{X}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{C})$, where $\mathbf{C}$ is a $n \times n$ matrix defined as $\mathbf{C} = \sigma_0^2\mathbf{I} + \mathbf{X}\mathbf{A}^{-1}\mathbf{X}^{\mathrm{T}}$, and $\mathbf{A}$ is a diagonal matrix with components $A_{jj} = \alpha_g$, if the $j$-th feature belongs to the $g$-th group, for $j = 1, \ldots, d$. This value, $\mathcal{P}(\mathbf{y}|\mathbf{X})$, can be easily optimized iteratively with respect to each $\alpha_g$ using an algorithm similar to the one described by Tipping and Faul (2003). In this case one typically finds that most of the $\alpha_g$'s tend to infinity, enforcing the posterior for the corresponding $\mathbf{w}_g$ to be a spike at the origin. Unfortunately the optimization process is more difficult and there is no closed form solution for the optimal $\alpha_g$ while the other hyper-parameters are kept fixed. Ji et al. (2009) provide a closed form approximate solution which is shown to perform well in practice but we consider here the exact maximization of the

model evidence. Appendix B describes an algorithm which can be used for this task. The model evidence need not be convex with respect to the different hyper-parameters. Hence, the solution obtained depends on the starting configuration of the optimization algorithm. One typically starts from all hyper-parameters being equal to infinity and iteratively optimizes each hyper-parameter until convergence.

The group ARD formulation (more precisely the type-II maximum likelihood principle followed by group ARD) can be seen as a Bayesian approach where the posterior distribution of each hyper-parameter $\alpha_g$, with $j = 1, \ldots, d$, is approximated by a delta function centered at the peak of the exact posterior under the assumption of a flat prior for each $\alpha_j$ (Bishop, 2006). Thus, the actual prior for $\mathbf{w}_g$ is a hierarchical mixture where the hyper-prior for $\alpha_g$ is set to be flat (actually flat in log-scale) (Tipping, 2001). If we marginalize out $\alpha_g$, the actual prior can be shown to become the improper prior:

$$\mathcal{P}(\mathbf{w}_g) \propto \int \mathcal{N}(\mathbf{w}_g|\mathbf{0}, \alpha_g^{-1}\mathbf{I}) d\alpha_g/\alpha_g \propto 1/||\mathbf{w}_g||_2^{d_g} .$$

where $1/\alpha_g$ is the flat hyper-prior for $\alpha_g$ in log-scale and $d_g$ is the dimension of $\mathbf{w}_g$. This improper prior favors solutions with all the components of $\mathbf{w}_g$ set equal to zero since it has an infinitely tall spike at the origin. It also promotes solutions with coefficient values far from zero since it has heavy tails. Thus, it enjoys similar properties to those of the group horseshoe.

The posterior distribution of $\mathbf{w}$ under the group ARD model is Gaussian since both the likelihood and the prior are Gaussian. Thus, sequential experimental design can be carried out very easily in this model using techniques similar to those described in Section 4. Finally, note that the group ARD lacks a hyper-parameter to specify the desired level of sparsity at the group level. The uniform prior assumed for $\alpha_g$ can be considered to be optimal when there is no information about the level of sparsity associated to the learning problem. Nevertheless, this prior can be sub-optimal when such information is available beforehand or when it can be estimated from the data, for example, by cross-validation.

### 5.5 Other Related Methods

Instead of the EP algorithm, it is also possible to use Markov chain Monte Carlo techniques to approximate the posterior distribution of the Bayesian model introduced in Section 2. For this, we only have to interpret the prior described in (3) for each group of model coefficients $\mathbf{w}_g$ as a mixture of two multivariate Gaussians. A first multivariate Gaussian with zero variance for the different components of $\mathbf{w}_g$ (the spike) and a second multivariate Gaussian with $v_0$ variance. These two variances are equivalent to the latent parameters $\lambda_g$ described in Sections 5.2 and 5.3 for the Bayesian group LASSO and the group horseshoe. Thus, we can rely on a Gibbs sampling algorithm very similar to the one described in those sections. Appendix A further details this algorithm. It is inspired from other works for approximate inference in a Bayesian model based on the standard spike-and-slab prior (George and McCulloch, 1997; Lee et al., 2003). Gibbs sampling has also been used by Scheipl et al. (2012) to carry out posterior inference on additive regression models using a generalized prior similar to the one described in this document. Section 7 shows however that EP provides equal performance at a much smaller computational cost. In particular, EP is hundreds of times faster than Gibbs sampling, which is too slow to carry out sequential experimental design.

Variational Bayes (VB) (Attias, 2000; Jaakkola, 2001) is a common alternative for approximate inference in Bayesian models. VB consists in fitting a parametric distribution $Q$ with the aim to ap-

proximate the posterior distribution $\mathcal{P}$. For this, the Kullback-Leibler divergence between $Q$ and $\mathcal{P}$, $\mathrm{KL}(Q||\mathcal{P})$, is minimized. Such an approach differs from EP where the approximate minimization of $\mathrm{KL}(\mathcal{P}||Q)$ takes place. As a result, one can expect VB to be less accurate than EP for the model described in Section 2. Specifically, in sparse linear regression models optimal predictive performance in terms of mean square error is expected to be given by the mean of the posterior distribution. In general, EP produces a global fit to the posterior distribution while VB only approximates the posterior locally around one of its modes. This is illustrated in Bishop (2006). Posterior distributions generated by spike-and-slab priors are often multi-modal and hence the global fit produced by EP is expected to be better at approximating the posterior mean than the local approximation generated by VB. Furthermore, several works in the literature also report a preference of EP over VB in terms of accuracy of the posterior approximation (Minka, 2001; Nickisch and Rasmussen, 2008).

A different generalization of the standard spike-and-slab prior has been proposed in a multi-task setting by Hernández-Lobato et al. (2010). They describe a Bayesian model for the selection of features that are relevant for prediction across $L$ classification tasks. These tasks share the same $d$ features, although the feature values might be different across tasks. The relevant features are identified by using standard spike-and-slab priors on the coefficients of each task, where these priors share the same binary latent variables $\mathbf{z} = (z_1, \ldots, z_d)^{\mathrm{T}}$. These latent variables indicate whether the corresponding features are used for classification in all the tasks or in none of them. This prior cannot be used to tackle the group feature selection problem considered here. In particular, it does not allow to introduce prior knowledge on groups of features that are believed to be jointly relevant or irrelevant for the same task. By contrast, the prior considered in this work can be used to address the multi-task feature selection problem by reformulating it as a single-task learning problem on an extended space:

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \mathbf{X}_L \end{pmatrix}, \qquad \mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_L \end{pmatrix}, \qquad \mathbf{w} = \begin{pmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_L \end{pmatrix},$$

where $\mathbf{X}_l$, $\mathbf{y}_l$ and $\mathbf{w}_l$, with $1 \leq l \leq L$, respectively denote the training instances, the targets and the model coefficients for the $l$-th learning task. The equivalence holds provided that there is a group per feature and that the $g$-th group contains the $L$ coefficients associated with the $g$-th feature, for $g = 1, \ldots, d$.

The generalized spike-and-slab prior described in Section 2 can be seen as a degenerate case of the network-based prior proposed by Hernández-Lobato et al. (2011). These authors suggest a modification of the standard spike-and-slab prior by introducing prior dependencies among the components of $\mathbf{z} = (z_1, \ldots, z_d)^{\mathrm{T}}$, that is, the vector of latent variables which indicate whether to use or not each feature for prediction. These dependencies are determined by a network of features. Whenever two features are connected in this network, the two corresponding latent variables are positively correlated. The amount of correlation is specified by a positive parameter $b$. When the network of features contains $G$ connected components, each component being composed of the features of the same group, and $b$ tends to $+\infty$, the network-based prior is equivalent to the prior considered in this work. However, the network-based prior suggested by Hernández-Lobato et al. (2011) does not provide a direct estimate of the relevance of each group. It only computes posterior probabilities for individual features. Furthermore, the approximate inference mechanism used does not take into account correlations among the different components of $\mathbf{w}$. Specifically, the approximation considered for the posterior of $\mathbf{w}$ factorizes among the different components

of this vector. In the model described in this document these correlations are proven to be very useful to perform sequential experimental design, that is, to determine which instance to include in the training set to obtain the most information about $\mathbf{w}$ (see Section 7). Additionally, the model considered by Hernández-Lobato et al. (2011) can only address binary classification problems and was not designed to address regression problems, which is the focus of the present document.

The work of Yen and Yen (2011) describes an alternative generalization of the spike-and-slab prior which considers both sparsity at the group and the feature level. Specifically, these authors introduce two sets of latent variables. A first set is used to describe whether or not each group of variables is used for prediction, and a second set is used to describe whether or not each feature within a group is used for prediction. Thus, the prior considered in this work can be seen as a particular case of the prior considered by these authors where there is no sparsity at the feature level, but only at the group level. To infer the model coefficients from the data Yen and Yen (2011) do not follow a complete Bayesian approach, and instead find the *maximum a posteriori* (MAP) solution using a blockwise coordinate ascent algorithm. Finding the MAP solution in such model is arguably controversial. In particular, the posterior distribution includes delta functions which take infinite values at some positions and make the objective unbounded. Furthermore, this task involves solving a combinatorial problem which is NP-Hard. To address this difficulty Yen and Yen (2011) propose to use a majorization-minimization technique to simplify the computations needed. Finally, even though they can approximate the MAP solution, their approach does not provide an estimate of the correlations among the different components of $\mathbf{w}$ which are required to carry out sequential experimental design, as described in Section 4.

## 6. Analysis of Group Sparsity

In this section we study the properties of the generalized spike-and-slab prior to favor solutions that are sparse at the group level. The alternative priors described in Section 5, which can also be used for this purpose, are also analyzed in detail for the sake of comparison. The analysis introduced is based on the work of Carvalho et al. (2009) about the sparsity properties of the standard horseshoe prior and shows interesting insights about the regularization process enforced by each prior distribution and the potential benefits and drawbacks for group feature selection.

Consider the vector $\mathbf{w}_g$ summarizing the $d_g$ model coefficients corresponding to the features contained in the $g$-th group. The different priors for $\mathbf{w}_g$ analyzed in this section are displayed in Table 1 alongside with their associated hyper-parameters. In this table $\mathcal{N}(\cdot|\boldsymbol{\mu},\boldsymbol{\Sigma})$ denotes a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$; $\mathcal{C}^{+}(\cdot|a,b)$ denotes a half-Cauchy distribution with location and scale parameters $a$ and $b$, respectively; $\delta(\cdot)$ denotes a point probability mass evaluated at the origin; and $||\cdot||_2$ denotes the $\ell_2$-norm. We do not include the prior for the group LASSO since it is identical to the one in the Bayesian group LASSO, as described in Section 5.2. Figure 1 shows the different priors displayed in Table 1 for some values of their hyper-parameters and for a group of size two, that is, $d_g = 2$. In this figure an arrow denotes a point probability mass at the origin. Note that most priors have an infinitely tall spike at the origin to favor solutions with all the model coefficients near or equal to zero. The only exception is the prior corresponding to the group LASSO, which has a sharp peak at the origin instead. From these priors, only the spike-and-slab is able to put a positive probability mass at the origin. This probability is specified by the hyper-parameter $p_{0,g}$. By contrast, in the group horseshoe, the group ARD and the Bayesian group LASSO the probability of observing $\mathbf{w}_g$ at the origin is zero. This means that one will never observe

actual zeros in the samples from these priors. On the other hand, an appealing property of the group ARD and the group horseshoe is that they allow for values of $\mathbf{w}_g$ located far from the origin since they have heavy tails. This is not the case of the spike-and-slab. However, the hyper-parameter that models the variance of the slab, $v_0$, can be made arbitrary large to account for coefficients significantly different from zero without changing the desired level of sparsity, specified by $p_{0,g}$. The group ARD, the spike-and-slab and the group horseshoe are hence expected to be effective for inducing sparsity at the group level. Specifically, for some value of the prior hyper-parameters, they will either strongly drive the values of the model coefficients towards the origin (as a consequence of the spike), or they will leave them barely unchanged (as a consequence of the heavy tails or the large variance of the slab). The prior corresponding to the Bayesian group LASSO has neither heavy tails nor an infinitely tall spike at the origin. Thus, this prior is expected to perform worse in this task. Finally, the ARD prior is not fully adequate to our problem as it does not include a hyper-parameter to set the desired level of group sparsity, which can be strongly problem dependent.

| **Prior for $\mathbf{w}_g$** | **Density** | **Hyper-parameters** |
|---|---|---|
| Generalized spike-and-slab | $p_{0,g}\mathcal{N}(\mathbf{w}_g\|\mathbf{0}, v_0\mathbf{I}) + (1-p_{0,g})\delta(\mathbf{w}_g)$ | $v_0$, $p_{0,g}$ |
| Group horseshoe | $\int \mathcal{N}(\mathbf{w}_g\|\mathbf{0}, \lambda_g^2\tau^2\mathbf{I})\mathcal{C}^+(\lambda_g\|0,1)d\lambda_g$ | $\tau$ |
| Bayesian group LASSO | $\propto \left(d_g\gamma^2\right)^{\frac{d_g}{2}} \exp\left\{-\gamma\sqrt{d_g}\|\|\mathbf{w}_g\|\|_2\right\}$ | $\gamma$ |
| Group ARD | $\propto 1/\|\|\mathbf{w}_g\|\|_2^{d_g}$ | - |

Table 1: Description of the different priors for enforcing sparsity at the group level.

### 6.1 Shrinkage Interpretation of the Prior Distributions

The different prior distributions for $\mathbf{w}_g$ that are displayed in Table 1 can be understood as a scale mixture of multivariate Gaussian distributions. More precisely, these priors are equivalent to a zero-mean multivariate Gaussian with a random covariance matrix $\lambda_g^2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. The prior distribution for $\lambda_g^2$ determines the resulting family of prior distributions for $\mathbf{w}_g$. Thus, under this representation, we have to marginalize out $\lambda_g^2$ to evaluate the actual prior probability density for $\mathbf{w}_g$. In particular,

$$\mathcal{P}(\mathbf{w}_g) = \int \mathcal{N}(\mathbf{w}_g|\mathbf{0}, \lambda_g^2\mathbf{I})\mathcal{P}(\lambda_g^2)d\lambda_g^2\,,$$

where $\mathcal{P}(\lambda_g^2)$ denotes the specific prior distribution for $\lambda_g^2$. The shrinkage properties of each of the prior distribution displayed in Table 1 can be analyzed by looking at the corresponding assumed prior density for $\lambda_g^2$. This density is displayed in Table 2 for each different prior for $\mathbf{w}_g$.

For simplicity, we focus in this section on a toy regression problem which can be analyzed in detail. This problem gives interesting insights about the shrinkage properties of each prior distribution for $\mathbf{w}_g$. In particular, we assume that there is a single group of $d_g$ model coefficients, that is, $\mathbf{w} = \mathbf{w}_g$. Furthermore, we assume that there are $n = d_g$ observations $\mathbf{y}^{\mathrm{T}} = (y_1, \ldots, y_n)$, one for each model coefficient, which are generated according to the rule described in (1) for $\sigma_0^2 = 1$. We also assume that the design matrix $\mathbf{X}$ is equal to the identity matrix $\mathbf{I}$. Under these settings, the optimal value for $\mathbf{w}_g$ is $\mathbf{y}$. Moreover, the expected posterior value for $\mathbf{w}_g$ can be computed exactly given $\lambda_g^2$.

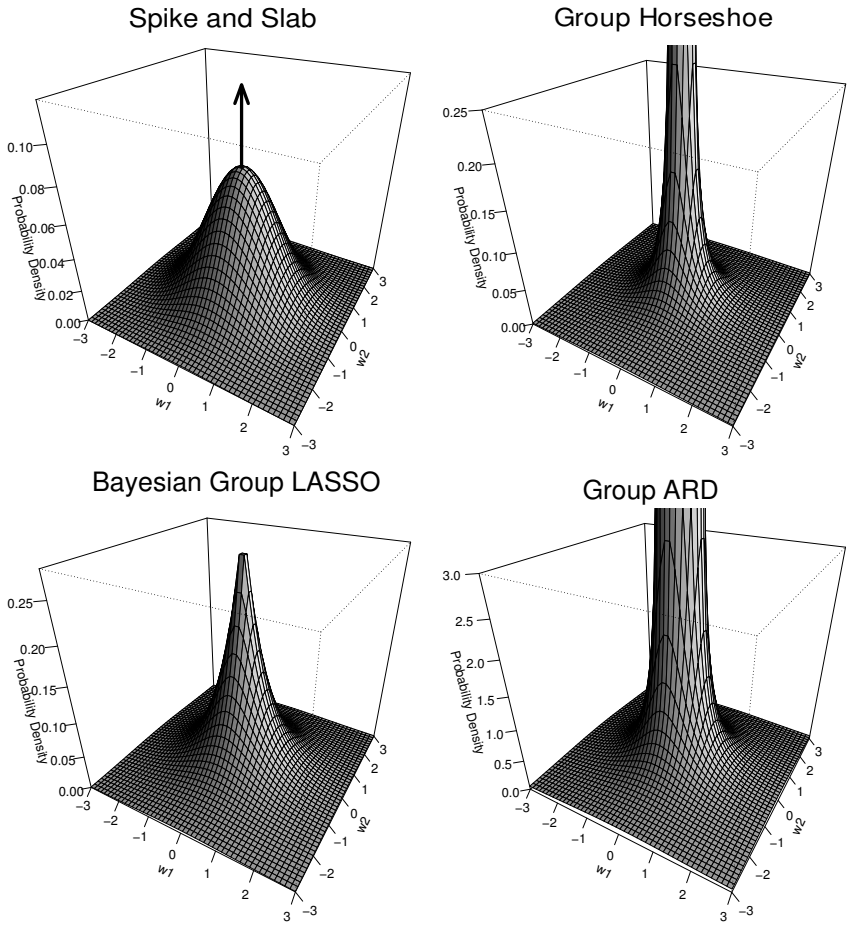



Figure 1: Plots of the different priors displayed in Table 1 for favoring solutions which are sparse at the group level. Results are displayed for $\mathbf{w}_g = (w_1, w_2)^{\mathrm{T}}$, that is, a group of size two, and for some particular values of the hyper-parameters of each different prior distribution. The arrow indicates a point probability mass at the origin. All priors except the prior for the group LASSO and the Bayesian group LASSO have an infinitely tall spike at the origin. The different hyper-parameters are set as follows: $p_{0,g} = 0.5$, $v_0 = 1$, $\tau = 1$ and $\gamma = 1$.

Namely,

$$\mathbb{E}[\mathbf{w}_g|\lambda_g^2] = \frac{\lambda_g^2}{1+\lambda_g^2}\mathbf{y} + \frac{1}{1+\lambda_g^2}\mathbf{0} = \frac{\lambda_g^2}{1+\lambda_g^2}\mathbf{y}, \tag{36}$$

where $\kappa = 1/(1+\lambda_g^2)$, with $\kappa \in [0,1]$, is a random shrinkage coefficient which can be understood as the amount of weight that the posterior mean places at the origin once the targets $\mathbf{y}$ are observed (Carvalho et al., 2009). If $\kappa = 1$ the posterior mean is completely shrunk towards the origin. If $\kappa = 0$, the posterior mean is not regularized at all. Since $\kappa$ is a random variable, it is possible to plot its prior density to analyze the shrinkage properties of each prior. This density is fully specified

| **Prior for $\mathbf{w}_g$** | **Prior density for $\lambda_g^2$** |
|---|---|
| Generalized spike-and-slab | $p_{0,g}\delta(\lambda_g^2 - v_0) + (1 - p_{0,g})\delta(\lambda_g^2)$ |
| Group horseshoe | $\mathcal{C}^+\left(\sqrt{\lambda_g^2}|0,\tau\right) 1/\left(2\sqrt{\lambda_g^2}\right)$ |
| Bayesian group LASSO | $\text{Gamma}\left(\lambda_g^2 \mid \frac{d_g+1}{2}, \frac{2}{d_g\gamma^2}\right)$ |
| Group ARD | $\propto 1/\lambda_g^2$ |

Table 2: Description of the different priors assumed for $\lambda_g^2$.

by the prior distribution for $\lambda_g^2$ which can be any of the ones displayed in Table 2, depending on the actual prior for $\mathbf{w}_g$. In an ideal situation, $\mathcal{P}(\kappa)$, that is, the prior distribution for $\kappa$, should favor the bi-separation of the model coefficients that is characteristic of sparse models at the group level. Specifically, while most groups of model coefficients take values close to zero, a few of them take values significantly different from zero. Thus, $\mathcal{P}(\kappa)$ should be large for values of $\kappa$ near one, to favor the shrinkage of un-important groups of model coefficients. Similarly, $\mathcal{P}(\kappa)$ should be large for values of $\kappa$ near zero, to barely shrink those groups of model coefficients which are important for prediction.

Figure 2 displays for each different prior distribution for $\mathbf{w}_g$, the corresponding prior distribution for the shrinkage coefficient $\kappa$, $\mathcal{P}(\kappa)$. The plots are displayed for a single group of size two. However, similar results are obtained for groups of larger sizes. The prior distributions are obtained from the densities displayed in Table 2 by performing a change of variable since $\kappa = 1/(1+\lambda_g^2)$. For each prior distribution, the corresponding hyper-parameters are selected so that the distance between the 10% and 90% percentiles of the resulting marginal distribution of each component of $\mathbf{w}_g$ is equal to 0.7, 3.5 and 17.5, respectively. These values correspond to *high*, *medium*, and *low* sparsity at the group level. The exception is the prior for $\kappa$ corresponding to group ARD, which does not have any hyper-parameter to specify the desired level of group sparsity. In this figure the arrows denote a point probability mass and the length of the arrow is proportional to the corresponding probability mass.

Figure 2 shows that the prior corresponding to the Bayesian group LASSO is not able to simultaneously produce large densities for values of $\kappa$ close to zero and one. Furthermore, the probability density for $\kappa = 1$ is always equal to zero. This is an unexpected result which questions the capacity of this prior to provide solutions that are sparse at the group level in a selective manner. In particular, under this prior it is not possible to achieve high levels of sparsity (this corresponds to high density values of $\kappa$ near one) while not shrinking the model coefficients that are different from zero towards the origin (this corresponds to high density of $\kappa$ near the origin). The next section illustrates that these issues still appear when the MAP solution is used and zeros are produced in the solution set, as in the group LASSO.

The other priors, that is, the spike-and-slab, the group horseshoe and the group ARD, do not suffer from the limitations described for the Bayesian group LASSO. These priors produce densities that are peaked at $\kappa = 1$ and at values of $\kappa$ near the origin. Furthermore, in the case of the spike-and-slab prior, one actually obtains a positive probability at $\kappa = 1$. Thus, the posterior distribution of the coefficients corresponding to non-predictive features will concentrate near the origin under these priors. On the other hand, both the group horseshoe and the group ARD priors are characterized by heavy tails. This can be observed in Figure 2 by the fact that they simultaneously

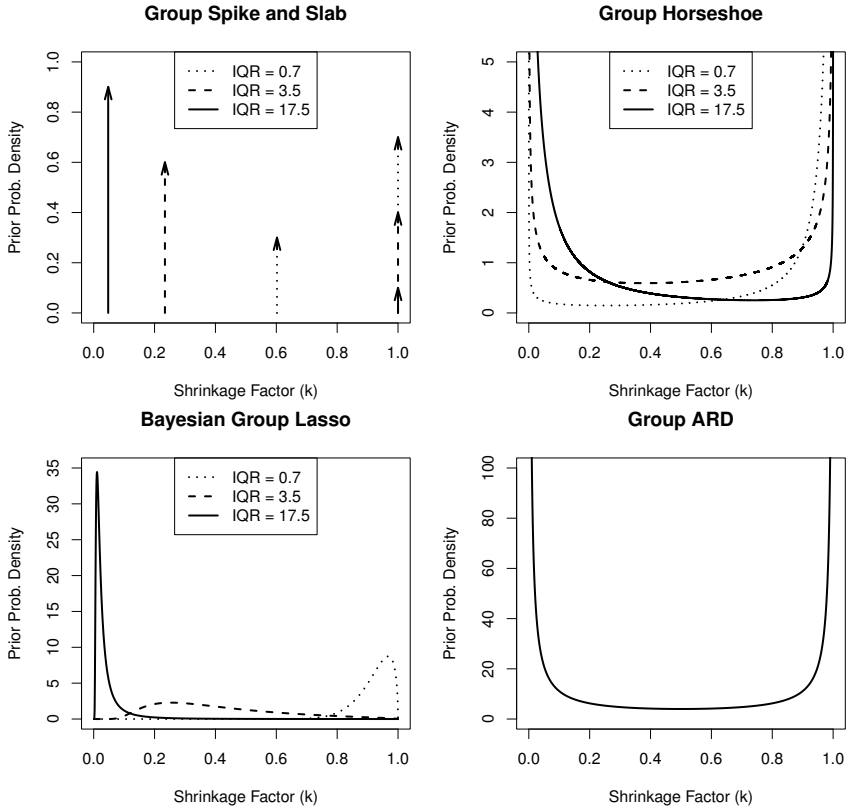



Figure 2: Prior distribution for $\kappa$ associated to the different priors for $\mathbf{w}_g$ displayed in Table 1. The plots are displayed for a single group of size two, and for some particular values of the hyper-parameters of each prior distribution for $\mathbf{w}_g$ that give the same inter-quantile range (IQR) for each individual component of this vector. A single curve is plotted for the group ARD prior since it does not have any hyper-parameter. An arrow denotes a point probability mass.

give high probabilities to values of $\kappa$ near the origin. The consequence is that these priors will barely regularize important groups of coefficients that are strictly needed for prediction. The spike-and-slab prior does not have heavy tails. However, a similar effect can be obtained by specifying large values for $v_0$, the parameter that controls the variance of the slab. Therefore, these three prior distributions are expected to selectively shrink the posterior mean, which is the ideal situation for regression problems which are sparse at the group level.

### 6.2 Regularization Properties of the Prior Distributions

We continue the analysis of the toy regression problem described in Section 6.1. Specifically, we study the behavior of the posterior mean, $\mathbb{E}[\mathbf{w}_g]$, under the different priors for $\mathbf{w}_g$ when the targets $\mathbf{y}$ are similar to or very different from the prior mean, that is, a vector with all the components equal to zero. It is possible to show, by marginalizing (36) over the posterior distribution for $\lambda_g^2$, that in this toy problem $\mathbb{E}[\mathbf{w}_g]$ is a vector parallel to $\mathbf{y}$. More precisely, these two vectors only

differ in their $\ell_2$-norms. In particular, the $\ell_2$-norm of $\mathbb{E}[\mathbf{w}_g]$ is always smaller or equal than the $\ell_2$-norm of $\mathbf{y}$. The same applies when the MAP estimate is considered instead of the posterior mean, as in the group LASSO. Thus, we can analyze the regularization properties of each prior distribution by comparing the $\ell_2$-norm of the targets $||\mathbf{y}||_2$ with the corresponding value of the $\ell_2$-norm of the posterior mean $||\mathbb{E}[\mathbf{w}_g]||_2$. Figure 3 shows a comparison between these two norms for each different prior distribution for $\mathbf{w}_g$ and for different values of the prior hyper-parameters. The identity function is also displayed to indicate where the two norms are equal. We include plots both for the group LASSO and the Bayesian group LASSO to illustrate the differences between the posterior mean $\mathbb{E}[\mathbf{w}_g]$ and the MAP estimate $\hat{\mathbf{w}}_g$. Similarly, since the spike-and-slab prior has two hyper-parameters, $p_{0,g}$ and $v_0$, we display two plots for this prior. In the first plot $p_{0,g}$ varies and $v_0$ is kept constant. Conversely, in the second plot $v_0$ varies and $p_{0,g}$ is kept constant. Finally, we provide a single curve for the group ARD prior since it does not have any hyper-parameter.

Figure 3 shows that for small values of $||\mathbf{y}||_2$, the group LASSO actually drives the model coefficients to zero. In particular, under this model it is possible to show that if $||\mathbf{y}||_2 \leq \sqrt{d_g}\gamma/2$ then $||\hat{\mathbf{w}}_g||_2$, the $\ell_2$-norm of the MAP estimate of $\mathbf{w}$, is equal to zero. When $||\mathbf{y}||_2 > \sqrt{d_g}\gamma/2$, the $\ell_2$-norm of $\hat{\mathbf{w}}_g$ must satisfy $||\hat{\mathbf{w}}_g||_2 = ||\mathbf{y}||_2 - \sqrt{d_g}\gamma/2$. Thus, if the targets $\mathbf{y}$ are significantly different from the mean of the prior, the differences between the two norms in the group LASSO are actually constant and proportional to the value of the hyper-parameter $\gamma$, which controls the level of sparsity. The consequence is that for high levels of group sparsity, as specified by $\gamma$, the group LASSO regularizes the coefficients that are different from zero and introduces a significant bias in their estimation. Specifically, under this model it is not possible to simultaneously consider large values for the model coefficients and high levels of group sparsity. When the posterior mean is considered instead of the MAP estimate, as in the Bayesian group LASSO, the observed behavior is very similar for large values of $||\mathbf{y}||_2$. Nevertheless, in this case, small values of $||\mathbf{y}||_2$ no longer produce zeros in the estimation for $\mathbf{w}$, but a stronger regularization effect which forces $\mathbb{E}[\mathbf{w}_g]$ to be closer to the origin. These two methods, that is, the group LASSO and the Bayesian group LASSO, are hence unable to shrink the model coefficients in a selective manner and are expected to lead to an impaired prediction performance in problems that are actually sparse at the group level.

The group ARD also drives the model coefficients towards zero for small values of $||\mathbf{y}||_2$. Specifically, if $||\mathbf{y}||_2 \leq \sqrt{d_g}$ then the optimal parameter $\lambda_g^2$ which maximizes the model evidence is equal to zero, and the posterior estimate of $\mathbf{w}$, $\mathbb{E}[\mathbf{w}_g]$, is placed at the origin. When $||\mathbf{y}||_2 > \sqrt{d_g}$, the optimal value for $\lambda_g^2$ is equal to $||\mathbf{y}||_2^2/d_g - 1$ and hence, from (36), $||\mathbb{E}[\mathbf{w}_g]||_2 = ||\mathbf{y}||_2 - d_g/||\mathbf{y}||_2$. Thus, the group ARD also introduces a bias in the estimation of the model coefficients when the targets are significantly different from the origin. Nevertheless, this bias is equal to $d_g/||\mathbf{y}||_2$ and tends to zero when $||\mathbf{y}||_2$ approaches infinity. This is a consequence of the heavy tails of the prior for $\mathbf{w}_g$ which barely regularizes the model coefficients when these are significantly different from zero and strictly required for prediction. A similar behavior is observed for the group horseshoe. Namely, when $||\mathbf{y}||_2$ approaches infinity both norms tend to coincide and the prior distribution barely regularizes the model coefficients. By contrast, for small values of $||\mathbf{y}||_2$, the model coefficients are strongly regularized in an amount that depends on $\tau$. The smaller its value, the stronger the regularization effect. Note that this parameter has very little effect on the regularization of the model coefficients when $||\mathbf{y}||_2$ is large. This is a very interesting property. From this, we conclude that these two prior distributions, the group ARD and the group horseshoe, are expected to be useful to provide the bi-separation of the model coefficients that is characteristic of sparse models at the group level.

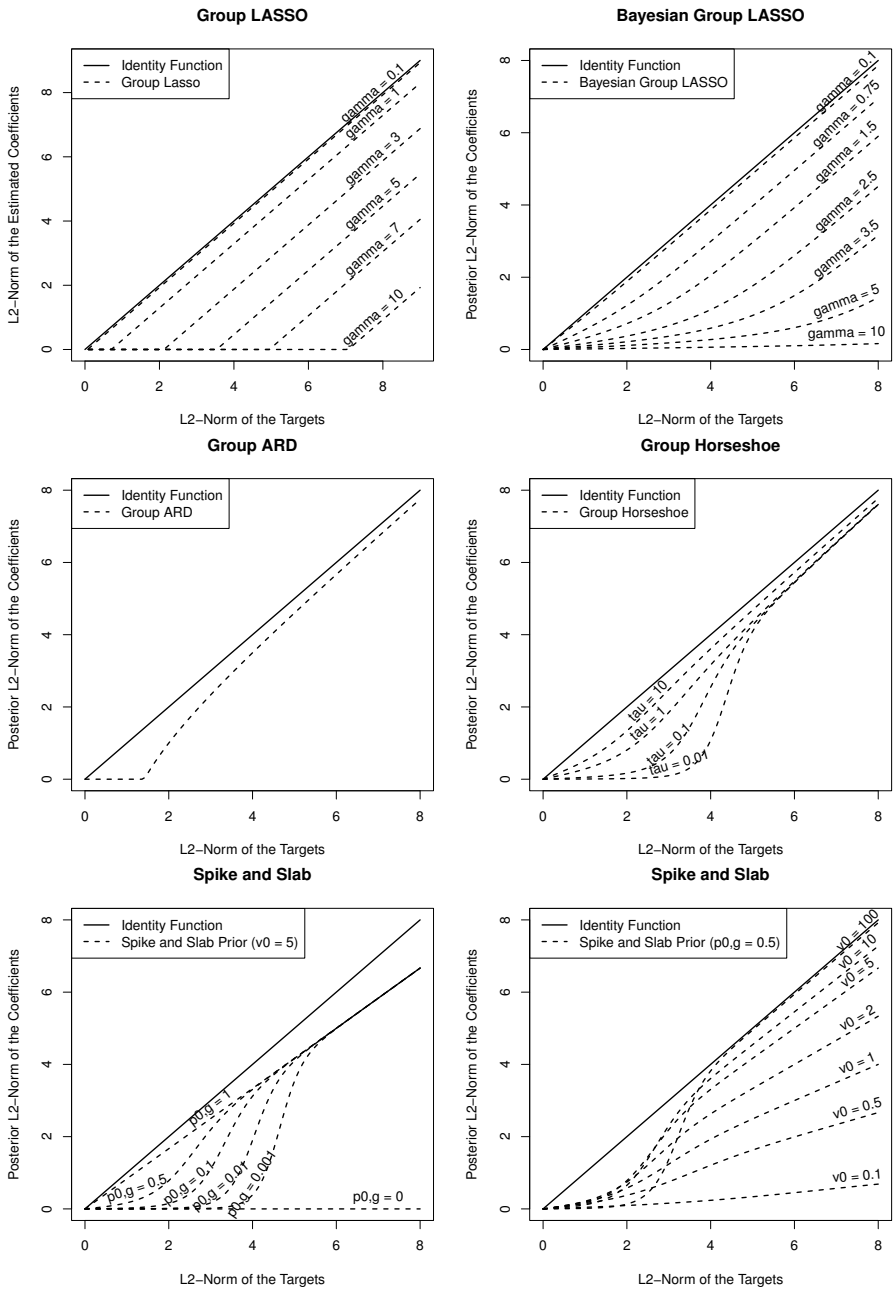



Figure 3: Plots of the $\ell_2$-norm of the posterior expected value of the model coefficients, as a function of the $\ell_2$-norm of the observed targets $\mathbf{y}$, for each different prior distribution for $\mathbf{w}_g$, and for different values of the hyper-parameters of each prior. We report results for both the Bayesian group LASSO and for the group LASSO, that is, the MAP estimate in the Bayesian group LASSO. The diagonal solid line represents where both norms are equal.

In the spike-and-slab prior, modifying $p_{0,g}$, that is, the parameter that determines the prior probability that the model coefficients are equal to zero, has an impact in the posterior mean when the optimal values of the model coefficients are close to zero. In particular, reducing this hyper-parameter produces a stronger regularization effect for small values of $||\mathbf{y}||_2$ without affecting the regularization of the model coefficients when these are significantly different from zero, or equivalently, when $||\mathbf{y}||_2$ is large. An exception is observed when $p_{0,g}$ is set equal to zero. In this case, the posterior mean is placed at the origin. The modification of $v_0$, that is, the parameter that determines the variance of the slab, has little impact when $||\mathbf{y}||_2$ is close to zero. By contrast, this parameter fully specifies the regularization of the model coefficients that are strictly important for prediction. Specifically, for large $||\mathbf{y}||_2$ it is possible to show that the posterior distribution for $\lambda_g^2$ tends to $\delta(\lambda_g^2 - v_0)$ and consequently, from (36), that $\mathbb{E}[\mathbf{w}_g] \approx \mathbf{y} - \mathbf{y}/(1+v_0)$. This indicates that the spike-and-slab prior introduces a positive bias in the estimation of the model coefficients, when these are far from the origin. This bias is proportional to the optimal value of each coefficient and it is a consequence of the absence of heavy tails in this prior distribution to explain for large values of the model coefficients. Nevertheless, we note that it is possible to reduce the estimation bias simply by increasing $v_0$, as illustrated by Figure 3. In summary, the spike-and-slab is also expected to perform well in problems that are sparse at the group level. In particular, this prior distribution is also able to model the bi-separation of the model coefficients that is characteristic of this type of problems.

## 7. Experiments

In this section, the performance of the model based on the EP algorithm and the generalized spike-and-slab prior is evaluated in several regression problems from different domains of application, using both simulated and real-world data. The problems analyzed include the reconstruction of sparse signals from a reduced number of noisy measurements (Huang and Zhang, 2010; Ji et al., 2008), the prediction of user sentiment from customer-written reviews of kitchen appliances and books (Pang et al., 2002; Blitzer et al., 2007) and the reconstruction of images of hand-written digits extracted from the MNIST data set (LeCun et al., 1998). The data sets of these problems have similar characteristics. That is, a large number of attributes and a rather small number of training instances, that is, $d \gg n$. Similarly, on each data set only a reduced number of features is expected to be useful for prediction. These are precisely the characteristics of the regression problems where the sparsity assumption is expected to perform well and to be useful for induction. In these experiments, the prior information about groups of features that are expected to be jointly relevant or jointly irrelevant for prediction is assumed to be given or it is estimated from additional data.

We refer to the regression model that assumes generalized spike-and-slab priors and uses EP for approximate inference as GSS-EP. This model is compared in this section with the related methods for group feature selection described in Section 5. Namely, the group LASSO (G-LASSO), the Bayesian group LASSO (BG-LASSO), the group horseshoe (G-HS), the group ARD formulation (G-ARD), a model that also assumes generalized spike-and-slab priors but uses Markov chain Monte Carlo sampling for approximate inference (GSS-MCMC) and finally, a regression model that also uses EP for approximate inference but only considers the standard spike-and-slab prior for induction (SS-EP). This last model is described in detail by Hernández-Lobato (2010) and is a particular case of GSS-EP which does not consider the grouping information in the induction process,

that is, all groups are of size one. SS-EP is included in the comparison to evaluate the benefit of considering sparsity at the group level instead of only at the feature level. Furthermore, comparing results with respect to SS-EP is also supported by the good performance obtained by such method in regression problems that are sparse at the feature level (Hernández-Lobato, 2010). Similarly, GSS-MCMC is included in the comparison to evaluate the performance of the EP approximation of the posterior distribution. In particular, Markov chain Monte Carlo methods do not suffer from any approximation bias, unlike deterministic techniques, such as the EP algorithm.

In our experiments, we report the training time of each method being evaluated. The different training algorithms have been implemented in R (R Development Core Team, 2011), and care has been taken to make them as efficient as possible.[1] Specifically, the implementation of G-LASSO is based on the fast algorithm described by Roth and Fischer (2008) and Appendix A gives all the details about the implementation of BG-LASSO, G-HS and GSS-MCMC. In these three methods, the posterior distribution of the model is approximated by generating 10,000 Gibbs samples after a burn-in period of 1,000 samples. This number of samples seems to be adequate and experiments indicate that no significant improvements are obtained by increasing the number of samples generated. All the details of the implementation of G-ARD are given in Appendix B. Finally, in GSS-EP, GSS-MCMC, SS-EP, G-ARD, BG-LASSO and G-HS the estimate of the model coefficients, $\hat{\mathbf{w}}$, is given by the approximate posterior mean. In G-LASSO $\hat{\mathbf{w}}$ is given by the MAP estimate.

### 7.1 Reconstruction of Sparse Signals

A first batch of experiments is carried out to illustrate the potential applications of the generalized spike-and-slab prior in the field of compressive sensing (Donoho, 2006; Candes and Wakin, 2008). The objective in compressive sensing is to reconstruct a sparse signal, generally codified in the model coefficients $\mathbf{w} = (w_1, \ldots, w_d)^{\mathrm{T}}$, from a limited set of linear measurements $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$, with $n \ll d$. The measurements $\mathbf{y}$ are obtained after projecting the signal $\mathbf{w}$ onto an $n \times d$ measurement matrix $\mathbf{X}$, that is, $\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} = (\varepsilon_1, \ldots, \varepsilon_n)^{\mathrm{T}} \sim \mathcal{N}(\mathbf{0}, \sigma_0^2 \mathbf{I})$ is a Gaussian noise. If $\mathbf{w}$ is sparse, it is possible to reconstruct this vector accurately from $\mathbf{y}$ and $\mathbf{X}$ using fewer measurements than the number of degrees of freedom of the signal, which is the limit required to guarantee the reconstruction of arbitrary signals. When $\mathbf{w}$ is not sparse, it can still be estimated using less than $d$ samples provided that it is compressible in some orthogonal basis $\mathbf{B}$, for example, a wavelet basis, such that $\tilde{\mathbf{w}} = \mathbf{B}^{\mathrm{T}}\mathbf{w}$ is sparse or nearly sparse. In this case, the measurement process is performed after projecting the signal onto the columns of $\mathbf{B}$, that is, $\mathbf{y} = \mathbf{X}\mathbf{B}^{\mathrm{T}}\mathbf{w} + \boldsymbol{\epsilon} = \mathbf{X}\tilde{\mathbf{w}} + \boldsymbol{\epsilon}$. Once an estimate of $\tilde{\mathbf{w}}$ is obtained from $\mathbf{y}$ and $\mathbf{X}$, we can approximate $\mathbf{w}$ using $\mathbf{w} = \mathbf{B}\tilde{\mathbf{w}}$.

We evaluate the different methods in a problem similar to the standard benchmark problems used in the field of signal reconstruction (Ji et al., 2008). More precisely, we generate 100 random sparse signals to be reconstructed from noisy measurements where each signal has $d = 512$ random components that are codified using a particular group sparsity pattern. Specifically, the components of each signal (i.e., the model coefficients) are iteratively assigned to $G = 128$ different groups of the same size that contain 4 components. From the 128 groups of components, only 4 randomly chosen groups contain components that are actually different from zero. The values of these components are uniformly chosen in the interval $[-1, 1]$. The resulting signal is stored in the vector $\mathbf{w}_0$ of model coefficients. This vector contains the sparse signal to be reconstructed. Given a particular signal $\mathbf{w}_0$ we then generate a reduced amount of measurements using a design matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^{\mathrm{T}}$

1. The R source code for GSS-EP is available at http://arantxa.ii.uam.es/%7edhernan/GSS-EP/.

whose rows are sampled uniformly in the hyper-sphere of radius $\sqrt{d}$. For the reconstruction of each signal a total of $n = 64$ noisy measurements $\mathbf{y} = \{y_1, \ldots, y_n\}$ are used. These are generated as $y_i = \mathbf{w}_0^{\mathrm{T}} \mathbf{x}_i + \varepsilon_i$, for $i = 1, \ldots, n$, where $\varepsilon_i$ follows a standard Gaussian distribution.

Given $\mathbf{X}$ and $\mathbf{y}$ we induce $\mathbf{w}_0$ using the different methods evaluated. Let $\hat{\mathbf{w}}$ be the corresponding estimate of the signal $\mathbf{w}_0$. The reconstruction error is quantified by $||\hat{\mathbf{w}} - \mathbf{w}_0||_2 / ||\mathbf{w}_0||_2$, where $||\cdot||_2$ denotes the $\ell_2$-norm. In these experiments, the hyper-parameters of each method are fixed optimally. In particular, we set $p_0$, that is, the fraction of groups initially expected to be relevant for prediction, equal to $4/128$ in GSS-EP and GSS-MCMC. In these two methods $v_0$, that is, the variance of the slab, is set equal to $1/3$, that is, the actual variance of the components of $\mathbf{w}_0$ that are different from zero. In SS-EP the same value is used for $v_0$, but $p_0 = 16/512$ since in this model there is a group for each different coefficient. In BG-LASSO we set $\gamma = \sqrt{120}$ so that the resulting prior has the same variance as the expected variance of the signal $\mathbf{w}_0$. In G-LASSO we try different values for $k$ and report the best performing value observed, which corresponds to $k = 8$. The group horseshoe prior does not have defined variances. Thus, we fix $\tau$ in G-HS so that the marginals under the group horseshoe have the same distance between the percentiles 1% and 99% as the marginals under the generalized spike-and-slab prior. We specifically use these extreme values for the percentiles because otherwise, for high levels of sparsity, that is, when $p_0$ is close to zero, the spike-and-slab prior can easily have an inter quantile range equal to zero. Finally, in all methods except G-LASSO, $\sigma_0^2$, that is, the variance of the Gaussian noise, is set equal to one.

The results of these experiments are displayed in Table 3. This table shows the average reconstruction error of each method and the corresponding average training time in seconds.[2] The figures after the symbol $\pm$ are standard deviation estimates. We note that GSS-EP obtains the best reconstruction error. Furthermore, the performance of this method is equivalent to the performance of GSS-MCMC. This indicates that the posterior approximation obtained by EP is accurate. The table also illustrates the importance of considering the grouping information for prediction. In particular, SS-EP obtains a significantly worse reconstruction error. After GSS-EP and GSS-MCMC, the best performing methods are GS-HS and G-ARD. The reconstruction error of these methods is only slightly worse than the reconstruction error obtained when generalized spike-and-slab priors are assumed. Finally, we note that the performance of G-LASSO, and especially BG-LASSO, is significantly worse than the performance of the other methods that use the grouping information. This validates the results of Section 6, where these two methods were expected to perform poorly. A paired Student t-test confirms that GSS-EP and GSS-MCMC perform better than the other methods being compared (p-value below 5%). On the contrary, the differences in performance between these two methods are not statistically significant.

| | GSS-EP | GSS-MCMC | SS-EP | G-LASSO | BG-LASSO | G-HS | G-ARD |
|---|---|---|---|---|---|---|---|
| **Error** | 0.29±0.11 | 0.29±0.10 | 0.71±0.20 | 0.54±0.11 | 0.92±0.03 | 0.35±0.12 | 0.39±0.12 |
| **Time** | 1.60±1.91 | 1168±64 | 3.89±3.42 | 2.39±1.63 | 3007±339 | 2909±328 | 2.56±1.55 |

Table 3: Average reconstruction error and training time of each method on the sparse signal reconstruction problem.

2. Training times were measured on an Intel(R) Xeon(R) 2.5Ghz CPU.

When comparing the average training time of the different methods displayed in Table 3 we observe that the fastest method is GSS-EP. In particular, GSS-EP needs approximately one second and a half for training, on average. This method is more than 500 times faster than GSS-MCMC and more than 1,000 times faster than BG-LASSO or G-HS, the induction methods that rely on Gibbs sampling for approximate inference. The training time of G-LASSO and G-ARD is also small, typically below 10 seconds, but above the training time of GSS-EP. Finally, we note that the training time of SS-EP exceeds the training time of GSS-EP. This is because in SS-EP the EP algorithm requires more iterations to converge, even though SS-EP is implemented with the same code as GSS-EP and also uses damped EP updates to improve convergence. This specific problem is also reported by Hernández-Lobato (2010) and is a consequence of the multiple modes of the posterior distribution under the standard spike-and-slab prior. When the grouping information is considered for induction, as in GSS-EP, the problem seems to be alleviated and the EP algorithm converges in fewer iterations.

Figure 4 displays, for a given instance of the signal reconstruction problem, the actual signal $\mathbf{w}_0$ and the different reconstructions generated by each method. The figure shows that GSS-EP and GSS-MCMC obtain the most accurate reconstructions and only fail to reconstruct the smallest components of the signal, probably due to the measurement noise. Furthermore, the reconstructions from these two methods look nearly identical. This gives further evidence indicating that the EP posterior approximation of the posterior mean is accurate. By contrast, SS-EP completely fails to reconstruct the original signal. In particular, without the grouping information it is impossible to reconstruct accurately this signal and SS-EP actually includes in the solution coefficients that only explain the observed data by chance. Similarly, many important coefficients are excluded from the solution found by SS-EP. The reconstructions generated by G-HS and G-ARD look accurate too. However, these methods include many coefficients with values just slightly different from zero which were not present in the original signal. In the group horseshoe this behavior can be explained because under this prior the probability of observing a group at the origin is zero. Similarly, in G-ARD the optimization process is not convex and can converge to a local and sub-optimal maximum of the type-II likelihood. The signal reconstructed by G-LASSO underestimates some values of the model coefficients and, at the same time, includes many coefficients that take values slightly different from zero. This behavior is due to the properties of the corresponding Multi-variate Laplace prior described in Section 6. In particular, this prior distribution is unable to achieve high levels of sparsity without shrinking too much the model coefficients that are different from zero. Finally, the signal reconstructed by BG-LASSO is rather inaccurate and not sparse at all. The Multi-variate Laplace prior produces an excessive shrinkage of non-zero model coefficients, while the magnitude of the coefficients that should be zero is not sufficiently reduced. Specifically, when a full Bayesian approach is used under this prior, the probability density of observing a group at the origin is zero, as illustrated by Figure 2. This questions the utility of the Bayesian group LASSO for group feature selection.

We also evaluate the utility of sequential experimental design to generate the different measurements used for the reconstruction of the sparse signal. For this, we repeat the previous experiments for an iteratively increasing number of measurements and report the reconstruction error of each method, except BG-LASSO, G-HS and GSS-MCMC. These methods are excluded because the cost of Gibbs sampling makes the corresponding computations too expensive. In these experiments, we start from an initial set of 32 measurements which are randomly generated using a design matrix $\mathbf{X}$ whose rows are sampled uniformly in the hyper-sphere of radius $\sqrt{d}$. The reason for this is that the

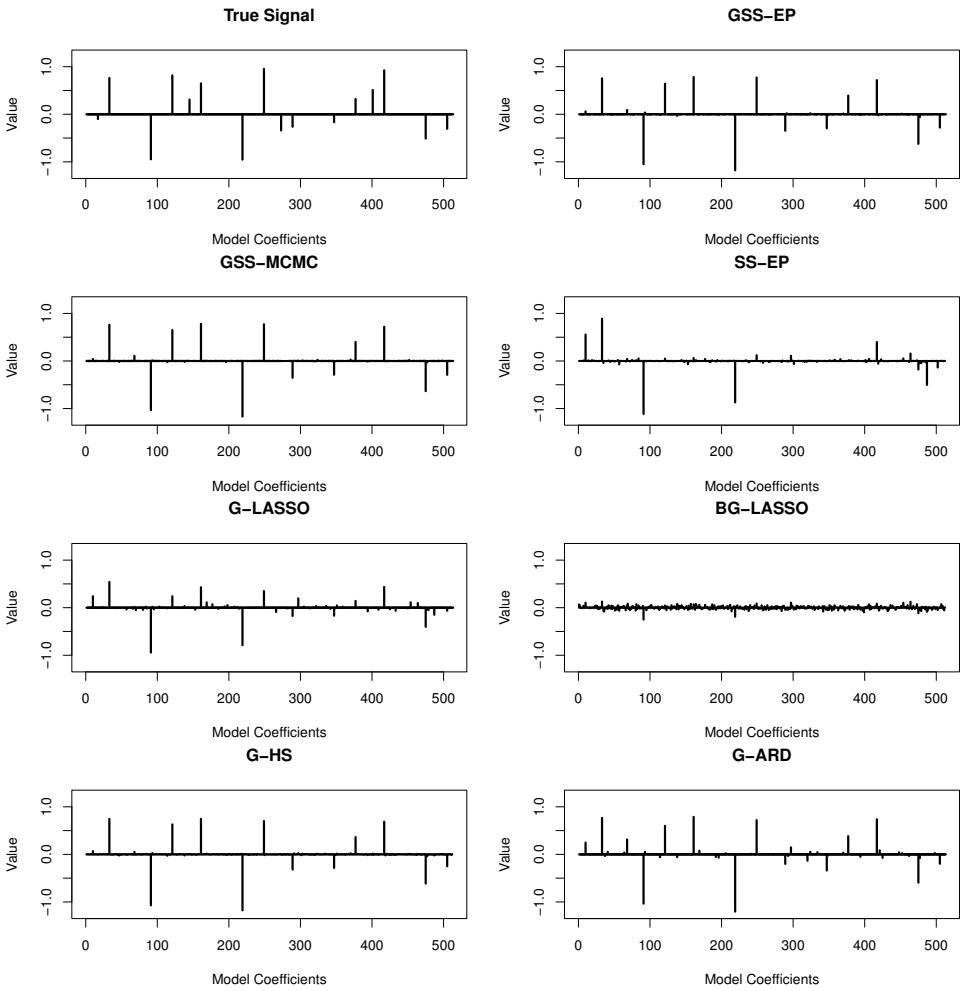



Figure 4: True signal $\mathbf{w}_0$ and the different reconstructions generated by each method for one instance of the signal reconstruction problem. Similar results are observed in other instances of this problem. The x-axis represents each of the model coefficients and the y-axis their values.

sequential design of new measurements from the beginning typically leads to over-fitting, as indicated by Ji and Carin (2007). We then iteratively generate up to 32 extra measurements and report the corresponding reconstruction error. The new measurements are generated using the sequential experimental design strategy described in Section 4. To investigate the benefits of such strategy, we also report results when these new 32 measurements are randomly generated. G-LASSO is also excluded from the experiments since it is impossible to compute posterior covariances under this method.

The results of the experiments are displayed graphically in Figure 5. This figure shows the reconstruction error of GSS-EP, SS-EP, and G-ARD as a function of the number of measurements performed when $\mathbf{X}$ is designed or randomly chosen. These curves indicate that sequential experimental design significantly improves the reconstruction error when compared to random design. In

particular, a steeper error descent with respect to the number of measurements made is produced for GSS-EP, SS-EP and G-ARD. The smallest final reconstruction error is achieved by GSS-EP followed by G-ARD, when $\mathbf{X}$ is designed. Furthermore, the reconstruction error of SS-EP significantly improves in this situation and becomes very close to the reconstruction error of the methods that consider the grouping information when $\mathbf{X}$ is chosen randomly. This illustrates the benefits of sequential experimental design. In any case, the reconstruction error of GSS-EP is much better than the reconstruction error of SS-EP when $\mathbf{X}$ is designed. This remarks again the beneficial properties of considering the grouping information during the induction process.

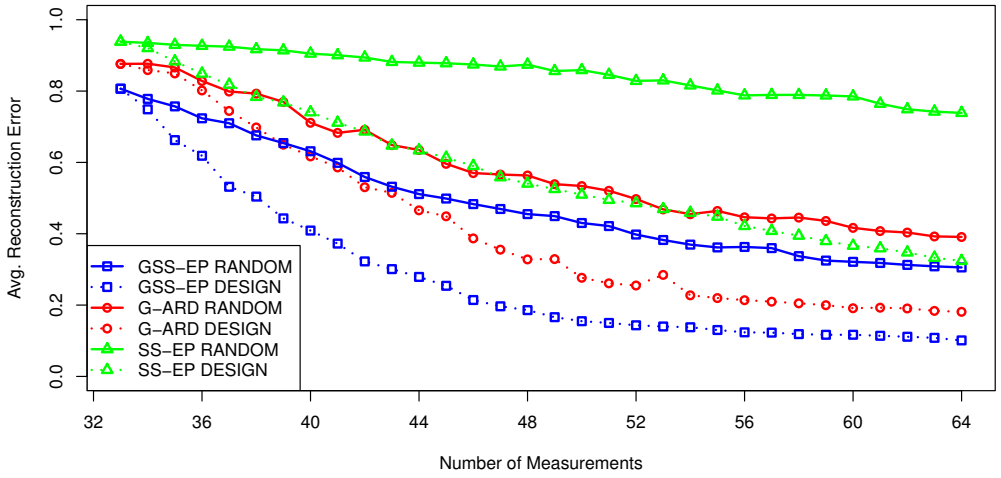



Figure 5: Average reconstruction error on the signal reconstruction problem as a function of the number of measurements performed for GSS-EP, G-ARD and SS-EP. We report results when $\mathbf{X}$ is chosen randomly and when $\mathbf{X}$ is designed.

The last experiments of this section investigate the sensitivity of the generalized spike-and-slab prior to the incorrect specification of the information about which groups of model coefficients are expected to be jointly relevant or jointly irrelevant for prediction. For this, we repeat the experiments whose results are displayed in Table 3. However, this time we introduce increasing levels of noise in the grouping information. Specifically, we permute at random 10%, 20%, 40%, 60% and 100% of the components of a vector with length equal to $d$, the total number of components in the signal to be reconstructed. Such vector summarizes the grouping information. Namely, its entries take values between 1 and the total number of groups, 128, and they respectively indicate the particular group each component belongs to. When 100% of the components of the vector are randomly permuted, the grouping information is completely random. By contrast, when a smaller fraction of the components are permuted, the vector may still contain useful information that is only partially correct.

Table 4 displays the average reconstruction error of each method for each different level of noise introduced in the grouping information. The error of a method has been high-lighted in bold-face when it is better than the error of GSS-EP and there is statistical evidence indicating a performance

difference. A paired t-test has been used for this purpose (p-value $< 5\%$). Similarly, when there is not enough statistical evidence to indicate a performance difference (GSS-EP performs similarly), the corresponding error has been underlined. The errors where GSS-EP is found to perform better according to the t-test have been left un-modified. The table shows that when the level of noise in the groups is small (i.e., 10% or 20%) the grouping information is still useful to reconstruct the sparse signal and GSS-MCMC and GSS-EP perform best. By contrast, when the level of noise introduced is large (i.e., 60% or 100%), using the grouping information for induction is harmful and the performance of the different methods degrades significantly. The best method is in this case SS-EP as a logical consequence of not considering the grouping information. Similar results have been obtained by Huang and Zhang (2010) for the group LASSO.

| Noise Level | GSS-EP | GSS-MCMC | SS-EP | G-LASSO | BG-LASSO | G-HS | G-ARD |
|---|---|---|---|---|---|---|---|
| 10% | 0.39±0.17 | 0.38±0.16 | 0.71±0.20 | 0.60±0.13 | 0.92±0.03 | 0.44±0.16 | 0.48±0.16 |
| 20% | 0.51±0.19 | **0.49±0.17** | 0.71±0.20 | 0.65±0.12 | 0.93±0.02 | 0.54±0.15 | 0.56±0.18 |
| 40% | 0.70±0.24 | 0.69±0.20 | 0.71±0.20 | 0.75±0.11 | 0.93±0.02 | 0.68±0.16 | 0.73±0.19 |
| 60% | 0.83±0.18 | 0.83±0.17 | **0.71±0.20** | 0.81±0.09 | 0.94±0.02 | **0.78±0.12** | 0.88±0.17 |
| 100% | 0.95±0.17 | **0.90±0.14** | **0.71±0.20** | **0.85±0.07** | 0.94±0.02 | **0.86±0.11** | 0.97±0.15 |

Table 4: Average reconstruction error of each method on the sparse signal reconstruction problem as a function of the fraction of components (noise level) that are randomly permuted in the vector that contains the grouping information.

### 7.2 Prediction of User Sentiment

In this section we investigate the utility of the generalized spike-and-slab prior to address the problem of sentiment prediction from user-written product reviews (Pang et al., 2002). In this task the objective is to predict the rating assigned by a user to a particular product in terms of the text contained in the product review by the user. We specifically focus on the four sentiment data sets described by Blitzer et al. (2007). These data sets contain reviews and the corresponding user ratings from different products extracted from *www.amazon.com*. The products from each data set are classified in a different category: *books*, *DVDs*, *kitchen appliances* or *electronics*, and the range of possible user ratings goes from 1 to 5. Each review is represented by a vector of features whose components correspond to the unigrams and bigrams (i.e., single words and pairs of words, respectively) that appear in at least 100 reviews of the products within the same category. The feature values are simply the number of times that the corresponding unigram or bigram appears in the review. Table 5 displays the total number of instances and the total number of features of each sentiment data set.

The predictive performance of the different methods compared is evaluated on the sentiment data sets *Books* and *Kitchen*. The data sets *DVDs* and *Electronics* are respectively used to generate the different groups of features considered for induction in the latter data sets. In particular, the *DVDs* data set is used to generate the groups for the *Books* data set and the *Electronics* data set is used to generate the groups for the *Kitchen* data set. All the features of the *Book* data set are contained in the *DVDs* data set and all the features of the *Kitchen* data set are contained in the

| Data Set | # Instances | # Features ($d$) |
|---|---|---|
| Books | 5501 | 1213 |
| DVDs | 5518 | 1303 |
| Kitchen | 5149 | 824 |
| Electronics | 5901 | 1129 |

Table 5: Number of instances and features corresponding to each sentiment data set.

*Electronics* data set. This means that we can safely consider exclusively the common features between these pairs of data sets to generate the groups. For this, we assume that the relevance or irrelevance of each group of features transfers from one data set to another. We then fit a simple linear ridge regression model using all the available data of the data set from which the groups are induced and a hierarchical clustering algorithm is run on the absolute values of the estimated model coefficients. This algorithm is stopped when 150 clusters are generated and the features associated to the coefficients contained in each cluster are grouped together. This guarantees that the features for which the associated model coefficients take similar values are contained in the same group. Thus, we expect that sets of relevant features (i.e., the features whose associated coefficients take large values) are actually placed in the same group of features. The same behavior is expected for irrelevant features (i.e., the features whose associated coefficients take small values). The results of the experiments reported in this section seem to confirm our expectation. In particular, using the grouping information for prediction has a beneficial effect on the predictive performance.

The evaluation procedure on each data set consists in generating 100 random partitions of the data into a training set with $n = 100$ instances and a test set with the remaining data. This particular size for the training set is chosen because we are interested in evaluating the performance of the different methods when $n \ll d$. In each train and test partition the features are normalized alongside with the targets to have zero mean and unit standard deviation across data instances. Furthermore, the hyper-parameters of GSS-EP, $p_0$ and $v_0$, are chosen in terms of an independent 10-fold cross-validation estimate of the prediction performance computed on the training data. The model evidence, as estimated by the EP algorithm, is not used for this purpose since it has been empirically found to provide inaccurate decisions in these data sets. The values of $p_0$ and $v_0$ are selected from a grid of $5 \times 5$ points. This grid is centered on a combination of hyper-parameters with good estimated predictive performances for GSS-EP. In SS-EP $p_0$ and $v_0$ are also selected by 10-fold cross-validation, but using a different grid of $5 \times 5$ points. This grid is also centered on a combination of hyper-parameters with good estimated predictive performances for SS-EP. In GSS-MCMC we use the same hyper-parameters as the ones found in GSS-EP. This allows to directly compare results between GSS-MCMC and GSS-EP. G-HS and BG-LASSO are too computationally expensive for a cross-validation search of the prior hyper-parameters. In BG-LASSO we set $\gamma$ so that the marginals of the resulting prior have, on average, the same variances as the variances of the marginals of the generalized spike-and-slab prior in GSS-EP. In G-HS we select $\tau$ so that the marginals of the group horseshoe prior have the same distance between the percentiles 1% and 99% as the marginals under the generalized spike-and-slab prior in GSS-EP. Recall that the horseshoe prior does not have defined variances. In G-LASSO we select $k$ by 10-fold cross-validation using a grid of 10 values. This grid contains hyper-parameter values with good estimated predictive performances for G-LASSO. Finally, $\sigma_0^2$, that is, the variance of the noise, is set equal to one in all the

methods compared. This specific value provides good results in general. Recall that the targets $\mathbf{y}$ are normalized to have zero mean and unit standard deviation in each train and test partition.

The results of these experiments are displayed in Tables 6 and 7 for the *Books* and the *Kitchen* data sets, respectively. These tables show the mean squared error (MSE) of each method on the test set and the corresponding average training time in seconds, without including the time required for finding the model hyper-parameters. The tables show that in both data sets GSS-EP obtains nearly the best prediction results. Furthermore, the prediction error of GSS-EP is again almost equivalent to the prediction error of GSS-MCMC. This gives further evidence supporting that the posterior approximation computed by EP is accurate. These results also indicate that it is important to consider the grouping information for prediction. In particular, SS-EP obtains worse reconstruction errors than GSS-EP in both data sets. After GSS-EP and GSS-MCMC, the best performing method is GS-HS. The prediction error of this method is only slightly worse than the prediction error of GSS-EP. G-ARD does not perform well in these data sets and seems to produce high levels of over-fitting. Specifically, it produces nearly the worst prediction results among the methods that consider the grouping information. This can be related to the fact that this method lacks a hyper-parameter to specify the desired level of group sparsity or to the multiple local maxima of the type-II likelihood. Finally, the performance of BG-LASSO, is also poor in both data sets while G-LASSO provides only good prediction results in the *Kitchen* data set. A Wilcoxon test confirms that GSS-EP and GSS-MCMC perform better than the other methods being evaluated in both data sets (p-value below 5%). On the other hand, the differences in performance between these two methods are not statistically significant. The non-parametric Wilcoxon test is used in these experiments where the train and test partitions are no longer independent because it is more conservative than the Student's t-test when the assumptions made by such a test are questionable (Demšar, 2006).

| | GSS-EP | GSS-MCMC | SS-EP | G-LASSO | BG-LASSO | G-HS | G-ARD |
|---|---|---|---|---|---|---|---|
| **MSE** | 2.17±0.10 | 2.17±0.10 | 2.26±0.10 | 2.30±0.10 | 2.35±0.16 | 2.23±0.08 | 2.48±0.23 |
| **Time** | 5.14±1.37 | 4202±563 | 3.09±1.00 | 1.99±1.78 | 7416±513 | 7571±397 | 4.77±2.95 |

Table 6: MSE and average training time for each method on the *Books* data set.

| | GSS-EP | GSS-MCMC | SS-EP | G-LASSO | BG-LASSO | G-HS | G-ARD |
|---|---|---|---|---|---|---|---|
| **MSE** | 1.95±0.11 | 1.94±0.11 | 2.08±0.10 | 2.04±0.14 | 2.21±0.19 | 2.02±0.08 | 2.19±0.23 |
| **Time** | 4.00±0.97 | 2967±109 | 2.24±0.61 | 1.94±1.42 | 5473±278 | 5699±436 | 3.69±1.62 |

Table 7: MSE and average training time for each method on the *Kitchen* data set.

When we compare the average training time of the different methods we observe that GSS-EP is among the fastest methods. Nevertheless, the average training time of SS-EP, G-LASSO and G-ARD is slightly better. The reason for this is that the EP algorithm takes more iterations to achieve convergence in the two sentiment data sets considered. In any case, the results reported show that GSS-EP is still significantly faster than GSS-MCMC, BG-LASSO or G-HS, the induction methods that rely on Gibbs sampling for approximate inference. Note that in these experiments we do not report plots of the values of the model coefficients, as estimated by each different method, because the differences among these cannot be appreciated by visual inspection. Furthermore, unlike in the previous experiments about the reconstruction of sparse signals, in these data sets the optimal values of the model coefficients are unknown.

In this section we also evaluate the utility of sequential experimental design to select the training instances to induce the different models. Specifically, we repeat the previous experiments when considering an iteratively increasing number of training instances for induction and report the resulting prediction performance of G-ARD, GSS-EP and SS-EP. For each data sets, that is, *Books* and *Kitchen*, we consider the train and test partitions of the data previously generated. Initially, we train each model with the set of 100 instances contained in training set. Then, we iteratively select one by one up to 64 additional training instances from a validation set. This validation set contains 256 instances randomly extracted from the test set. The instances selected are then iteratively included in the training set, which is used again for induction. Once each model has been re-trained, we report the corresponding average prediction error on the remaining test data. The training instance that is extracted from the validation set and included in the training set is chosen using the criterion for sequential experimental design described in Section 4. Namely, we select the instance $(\mathbf{x}_{\text{new}}, y_{\text{new}})$ that maximizes $\mathbf{x}_{\text{new}}^{\text{T}} \mathbf{V} \mathbf{x}_{\text{new}}$, where $\mathbf{x}_{\text{new}}$ has been normalized to have a unit $\ell_2$-norm. Note that the target $y_{\text{new}}$ is not used in this process. The corresponding target can be obtained once $\mathbf{x}_{\text{new}}$ has been selected and included in the training set. To investigate the benefits of this strategy, we also report results when these new 64 instances are selected randomly from the validation set. Finally, in each train and test partition of the data we set the different hyper-parameters of each method to the same values as the ones used in the previous experiments.

The results of those experiments are displayed in Figure 6 for each sentiment data set. The curves in this figure show the average prediction performance of each method as a function of the number of additional training instances considered for induction when these instances are selected from the validation set randomly or using sequential experimental design. We note that in both sentiment data sets sequential experimental design tends to improve the prediction error with respect to random selection. Nevertheless, these improvements are generally below the ones reported in Figure 5. Specifically, in the case of SS-EP the observed improvements are marginal. The smaller gains observed in this figure can be explained by the fact that in these experiments we do not generate $\mathbf{x}_{\text{new}}$, but select it from a validation set. In the case of GSS-EP the observed improvements in the prediction error are more significant. Furthermore, there is statistical evidence indicating that sequential experimental design provides better prediction performance in GSS-EP than random selection when 164 training instances are used for induction. Specifically, a Wilcoxon t-test comparing the prediction error of sequential experimental design and the prediction error of random selection provides a p-value smaller than 5%. This illustrates again the benefits of sequential experimental design and the favorable properties of considering the grouping information during the induction process. Finally, we note that in the case of G-ARD sequential experimental design provides much larger improvements in the prediction error. Notwithstanding, the results of this method are far from the ones of GSS-EP.

### 7.3 Reconstruction of Images of Hand-written Digits

A last batch of experiments is carried out to assess the effectiveness of the generalized spike-and-slab prior to reconstruct images of hand-written digits extracted from the MNIST data set (LeCun et al., 1998). These images can also be interpreted as random signals $\mathbf{w}$ to be reconstructed from a small set of random measurements. The MNIST data set contains 60,000 digit images of $28 \times 28$ pixels (i.e., $d = 784$). This means that in this data set there are about 6,000 images of each different digit from 0 to 9. The images are in gray scale and each pixel takes a value between 0 and 255.

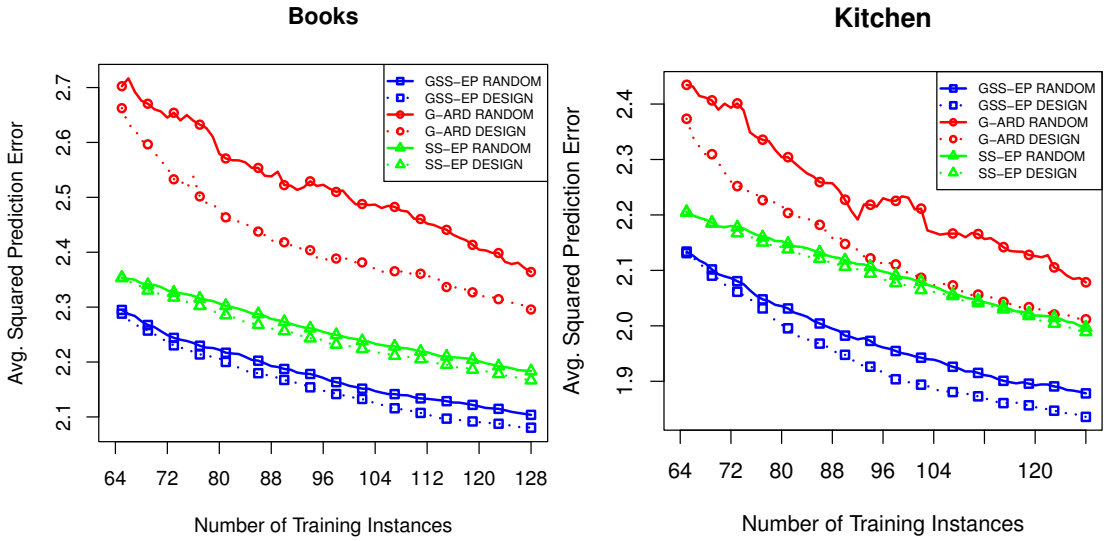



Figure 6: Mean squared prediction error for the *Books* and the *Kitchen* data sets as a function of the number of training instances for GSS-EP, G-ARD and SS-EP. We report results when the training instances are chosen randomly from a validation set and when they are selected from this data set using sequential experimental design.

Most of the pixels in each image are inactive and hence take values equal to 0. Conversely, only a few pixels are active in each image and take values near 255. Thus, these images are sparse and adequate to be reconstructed from a small set of random measurements using the method proposed.

We randomly extract 200 images of each digit from the MNIST data set. The first 100 random images are reconstructed from noisy measurements. The remaining 100 random images are used to generate the corresponding grouping information. Specifically, for each digit we generate $784/4 = 196$ groups of 4 pixels as follows.[3] First, we randomly select an initial pixel to represent each group. Then, for each different group we iteratively add, from the remaining set of pixels, the pixel that has on average the most similar activation pattern with respect to the pixels already included in the group. Similarity is measured in terms of the $\ell_2$-norm. This process is repeated until all groups contain exactly 4 pixels. The second set of 100 images corresponding to each different digit are exclusively used to estimate the similarity between the activation patterns of the pixels. Once the different groups of pixels have been generated we proceed to reconstruct the set of 100 random images of each digit. The images are first normalized so that each pixel takes a value in the interval $[0,1]$ (we divide each pixel value by 255). Then, each image is stored in the vector $\mathbf{w}_0$ of $d = 784$ model coefficients. This vector contains the image to be reconstructed. Given a particular image $\mathbf{w}_0$ we then generate a reduced amount of measurements using a design matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^{\mathrm{T}}$ whose rows are sampled uniformly in the hyper-sphere of radius $\sqrt{d}$. For the reconstruction of each image, a total of $n = 288$ noisy measurements $\mathbf{y} = \{y_1, \ldots, y_n\}$ are used. This particular number of measurements is chosen because it allows to accurately reconstruct the images considered and also to discriminate among the different methods being compared (Figure 9 also reports results

3. This particular choice of the number of groups and their size is supported by the good performance results obtained by the different methods that use the grouping information.

for other numbers of measurements). Finally, each measurement is generated as $y_i = \mathbf{w}_0^{\mathrm{T}}\mathbf{x}_i + \varepsilon_i$, for $i = 1,\ldots,n$, where $\varepsilon_i$ follows a standard Gaussian distribution, that is, the noise is introduced artificially.

Given $\mathbf{X}$ and $\mathbf{y}$ we induce $\mathbf{w}_0$ using the different methods evaluated. Let $\hat{\mathbf{w}}$ be the corresponding estimate of the image $\mathbf{w}_0$. The reconstruction error is measured as $||\hat{\mathbf{w}} - \mathbf{w}_0||_2/||\mathbf{w}_0||_2$, where $||\cdot||_2$ denotes the $\ell_2$-norm. In these experiments the hyper-parameters of each method are fixed optimally. In GSS-EP and GSS-MCMC $p_0$, that is, the fraction of groups initially expected to be relevant for prediction, is set for each different image equal to the actual fraction of groups whose associated pixels take values different from zero. For these two methods, $v_0$, that is, the variance of the slab, is set equal to the square of the average deviation from zero of each component of $\mathbf{w}_0$ that is different from zero. In SS-EP the same value is used for $v_0$, but $p_0$ is set equal to the actual fraction of pixels that take values different from zero. In BG-LASSO we set $\gamma$ so that the resulting prior has the same variance as the generalized spike-and-slab prior in GSS-EP and GSS-MCMC. In G-LASSO we try different values for $k$ and report the best performing value observed for each different digit. In G-HS we fix $\tau$ so that the marginals under the group horseshoe prior have the same distance between the percentiles 1% and 99% as the marginals under the generalized spike-and-slab prior. Finally, in all methods except G-LASSO, $\sigma_0^2$, that is, the variance of the Gaussian noise, is set equal to one.

The results of these experiments are displayed in Table 8. This table shows the average reconstruction error of each method for each different digit of the MNIST data set. The average training time in seconds of each method is also displayed on the last row of the table. We have high-lighted in bold face the reconstruction error of each method when it is better than the reconstruction error of GSS-EP and there is statistical evidence indicating a performance difference. A paired t-test has been used for this purpose (p-value $< 5\%$). Similarly, when there is not enough statistical evidence to indicate a performance difference (GSS-EP performs similarly), the corresponding error has been underlined. The results where GSS-EP is found to perform better according to a t-test have been left un-modified. The table shows that the best reconstruction errors are obtained by GSS-MCMC closely followed by GSS-EP. As a matter of fact for three digits, that is, 1, 3 and 7, there is no performance difference between the two methods. GSS-EP offers a better reconstruction error than the other methods that use the grouping information. These differences in the reconstruction error are statistically significant, except when comparing results with those of G-ARD for the digits 4 and 6. In those cases there is not enough statistical evidence to indicate a performance difference. When comparing results with respect to SS-EP we observe also better reconstruction errors. This highlights the beneficial properties of considering the grouping information in the reconstruction problem. After GSS-EP and GSS-MCMC, the best performing method that uses the grouping information is G-ARD followed by G-HS. Finally, G-LASSO does not perform very well in this task and BG-LASSO obtains the worst reconstruction errors. This validates the results of Section 6.

When comparing the average training time of the different methods displayed in Table 8 we observe again that GSS-EP is among the fastest methods. In particular, GSS-EP needs only 85 seconds for training, on average. This method is again significantly faster than GSS-MCMC, BG-LASSO or G-HS, the induction methods that rely on Gibbs sampling for approximate inference. We also note that the standard deviation estimate of GSS-EP is very large. This is a consequence of the fact that the EP algorithm in some extreme situations requires a high number of iterations to achieve convergence. In any case, the training time of GSS-EP is better than the training time of SS-EP. This shows that the EP algorithm in SS-EP requires even more iterations to converge. Finally, the

| | Digit | GSS-EP | GSS-MCMC | SS-EP | G-LASSO | BG-LASSO | G-HS | G-ARD |
|---|---|---|---|---|---|---|---|---|
| Reconstruction Error | **0** | 0.26±0.14 | **0.23±0.09** | 0.85±0.17 | 0.48±0.12 | 0.73±0.02 | 0.30±0.12 | 0.29±0.18 |
| | **1** | 0.12±0.02 | 0.12±0.02 | 0.15±0.14 | 0.36±0.06 | 0.62±0.05 | 0.13±0.02 | 0.14±0.02 |
| | **2** | 0.24±0.14 | **0.22±0.10** | 0.76±0.29 | 0.46±0.12 | 0.73±0.03 | 0.29±0.12 | 0.28±0.16 |
| | **3** | 0.20±0.09 | 0.20±0.08 | 0.73±0.32 | 0.42±0.12 | 0.71±0.03 | 0.25±0.10 | 0.24±0.14 |
| | **4** | 0.18±0.07 | **0.17±0.04** | 0.52±0.36 | 0.38±0.10 | 0.70±0.04 | 0.20±0.06 | 0.19±0.06 |
| | **5** | 0.22±0.13 | **0.19±0.07** | 0.59±0.35 | 0.41±0.10 | 0.71±0.03 | 0.24±0.09 | 0.24±0.14 |
| | **6** | 0.23±0.17 | **0.21±0.14** | 0.69±0.32 | 0.42±0.14 | 0.71±0.04 | 0.26±0.14 | 0.24±0.17 |
| | **7** | 0.17±0.07 | 0.16±0.04 | 0.52±0.37 | 0.38±0.09 | 0.69±0.04 | 0.19±0.06 | 0.19±0.08 |
| | **8** | 0.24±0.15 | **0.21±0.10** | 0.79±0.25 | 0.45±0.13 | 0.72±0.03 | 0.28±0.13 | 0.29±0.20 |
| | **9** | 0.18±0.11 | **0.17±0.08** | 0.58±0.37 | 0.39±0.11 | 0.70±0.04 | 0.21±0.09 | 0.20±0.12 |
| | **Time** | 85±100 | 20730±1472 | 211±154 | 292±191 | 25188±1495 | 25536±1842 | 59±36 |

Table 8: Average reconstruction error of each method on each digit of the MNIST data set and average training time in seconds.

training time of G-ARD is also small and slightly better than the training time of GSS-EP while the training time of G-LASSO is slightly longer.

Figure 7 displays in gray scale, for a given instance of the image reconstruction problem, the actual image to be reconstructed $\mathbf{w}_0$ and the corresponding reconstructions generated by each method. The figure shows that GSS-EP and GSS-MCMC obtain the most accurate reconstructions and only include some pixels with values slightly different from zero, probably due to the measurement noise. Furthermore, the reconstructions from these two methods look nearly identical. This gives further evidence indicating that the EP posterior approximation of the posterior mean is accurate. SS-EP completely fails to reconstruct the original image (except in the images corresponding to the digit 1) and actually includes in the solution several pixels that only explain the observed data by chance. Similarly, many important pixels are excluded from the solution found by SS-EP. The reconstructions generated by G-HS and G-ARD look accurate for some images. However, these methods include many coefficients with values just slightly different from zero which were not present in the original image. The images reconstructed by G-LASSO also include many coefficients that take values slightly different from zero. Furthermore, this method tends to underestimate the values of some pixels. Again, this behavior is due to the properties of the corresponding Multi-variate Laplace prior which is not able to achieve high levels of sparsity without shrinking too much the model coefficients that are different from zero. Finally, the image reconstructed by BG-LASSO is not sparse at all which questions again the utility of the Bayesian group LASSO for group feature selection.

In these experiments we also analyze the utility of the posterior approximation computed by EP in GSS-EP to identify relevant features for prediction. In particular, for each different digit and image, we record the value of the parameters $\sigma(p_g) \in [0,1]$, with $g = 1,\ldots,G$, of the EP approximation to $\mathcal{P}(\mathbf{z}|\mathbf{y},\mathbf{X})$. This approximate distribution is described in detail in (7) and each parameter $\sigma(p_g)$ estimates the posterior probability of using the $g$-th group of pixels for prediction. Thus, we can simply estimate the probability of using a particular pixel for the reconstruction task by looking at the corresponding parameter $\sigma(p_g)$. To evaluate the benefits of considering the grouping information for the computation of the pixel importance, we also analyze the results obtained by SS-EP. In SS-EP there is a group for each different pixel and hence $\sigma(p_g)$, with $g = 1,\ldots,d$, directly esti-

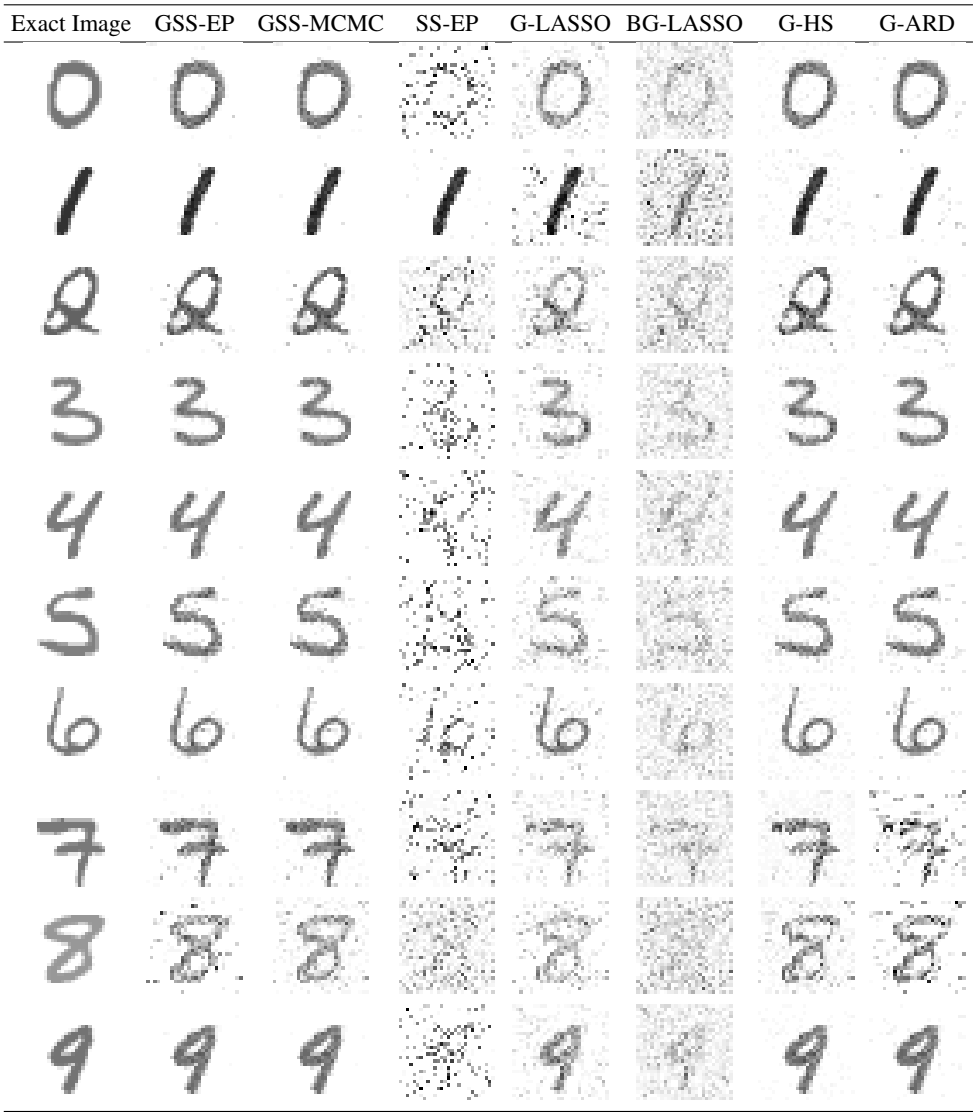



Figure 7: Representative exact images from the MNIST data set in gray scale for each different digit and the corresponding reconstructions obtained by each different method evaluated.

mates the corresponding pixel importance. Additionally, to compare results with the method based on Gibbs sampling we also report the corresponding estimates of GSS-MCMC. In GSS-MCMC the pixel importance is computed by estimating the average fraction of times the prior variance of the corresponding coefficient is different from zero within the generated samples (see Appendix A for further details). The results of these analyses are displayed graphically in Figure 8. This figure shows the importance of each pixel for the image reconstruction as estimated by GSS-EP, GSS-MCMC and SS-EP for each different digit. The results are averages over the 100 different images corresponding to each different digit. The importance of each pixels is displayed using a gray scale from 0, which corresponds to white, to 1, which corresponds to black. The results displayed show

that GSS-EP is very confident in the estimation of the pixel importance: most pixels are unlikely to be used for the reconstruction problem. Conversely, only a few pixels are very likely to be used for this task. In the case of SS-EP the results displayed show that this method is much less confident about the relative importance of each pixel for the image reconstruction. Specifically, most pixels show an intermediate level of importance, except for those of digit 1. This is probably due to the larger level of sparsity of these images, where only a few measurements are needed to identify the actual sparsity pattern. Finally, GSS-EP and GSS-MCMC offer nearly identical results. This gives further evidence supporting the accuracy of the posterior approximation computed by EP. We do not report here results for the other methods evaluated because they do not directly provide and estimate of the relative importance of each group of features for prediction.

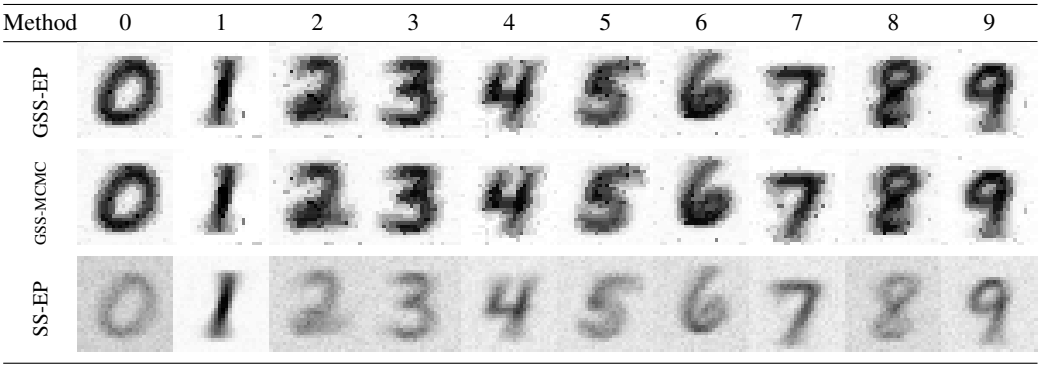



Figure 8: Average importance of each pixel for the reconstruction of the images corresponding to each different digit, as estimated by GSS-EP, GSS-MCMC and SS-EP. The feature importance of each pixel is represented in a gray scale from 0, which corresponds to the white color, to 1, which corresponds to the black color.

We also evaluate the utility of sequential experimental design to generate the different measurements used for the reconstruction of the images of hand-written digits. For this, we repeat the previous experiments for an iteratively increasing number of measurements and report the reconstruction error of each method. Again, we focus on GSS-EP, SS-EP and G-ARD. In these experiments, we start from an initial set of 128 measurements which are randomly generated using a design matrix $\mathbf{X}$ whose rows are sampled uniformly in the hyper-sphere of radius $\sqrt{d}$. We then iteratively generate up to 160 extra measurements and report the corresponding reconstruction error. The new measurements are generated using the sequential experimental design strategy described in Section 4. Finally, we also report results when these new 160 measurements are randomly generated.

The result of these experiments are displayed in Figure 9. The figure shows the reconstruction error of GSS-EP, SS-EP, and G-ARD as a function of the number of measurements performed when $\mathbf{X}$ is designed or randomly chosen for a representative subset of the ten digits contained in the MNIST data set (similar results are obtained for the digits not shown). Again, the curves displayed indicate that sequential experimental design significantly improves the reconstruction error when compared to random design. In particular, a steeper error descent with respect to the number of measurements made is produced for GSS-EP, SS-EP and G-ARD. The smallest final reconstruction error is achieved by GSS-EP followed by G-ARD, when $\mathbf{X}$ is designed. This illustrates the benefits of sequential experimental design. The reconstruction error of GSS-EP is also much better than the

reconstruction error of SS-EP when $\mathbf{X}$ is designed. This high-lights again the beneficial properties of considering the grouping information for the induction process.

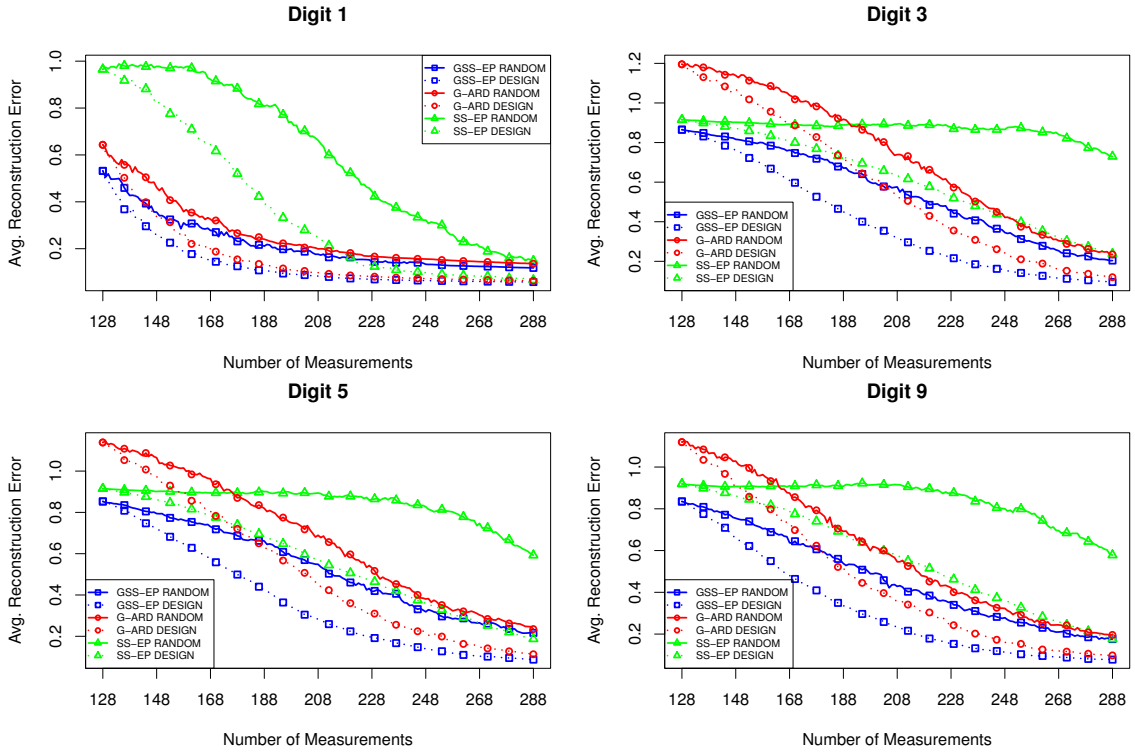



Figure 9: Reconstruction error for a representative subset of the ten digits contained in the MNIST data set as a function of the number of measurements carried out. We report results for GSS-EP, G-ARD and SS-EP when the measurements are chosen randomly and when they are chosen using sequential experimental design.

Finally, in this section we evaluate the utility of the generalized spike-and-slab prior to favor/penalize specific groups of features which are *a priori* believed to be more relevant/irrelevant for prediction. In particular, the prior distribution for $\mathbf{z}$, as defined in (4), allows to specify a different prior probability $p_{0,g}$, with $g = 1, \ldots, G$, of using the $g$-th group of features for prediction. In a typical application, all these parameters are set equal to a constant $p_0$. However, when there is prior information before hand about the relevancy or irrelevancy of specific group of features, this can be easily codified in GSS-EP by modifying these parameters. To evaluate this characteristic of the generalized spike-and-slab prior, we repeat the previous experiments where we report the reconstruction error of GSS-EP as a function of the number of random measurements performed, from 128 to 288 measurements. In these experiments we compare the results of GSS-EP when no specific prior information is used and when this information is actually used in the induction process. For this, we consider the 100 images that were used to identify each group of features. For each different digit, we evaluate the average $\ell_2$-norm of the pixels contained in each group. Then, we double the $p_{0,g}$ parameter for the 25 groups with the largest estimated $\ell_2$-norm. Similarly, we

reduce by half the $p_{0,g}$ parameter for the 25 groups with the smallest estimated $\ell_2$-norm. Recall that initially we set all these parameters equal to the actual fraction of groups that were relevant for prediction. To further evaluate the properties of the mechanism described to introduce prior information in the induction process, we also report results when we randomly reduce by half or double the value of the parameter $p_{0,g}$ for 50 randomly chosen groups. Such a protocol evaluates the effect of wrongly choosing the prior information. We do not evaluate here the other methods described for group feature selection because they do not allow to introduce this type of prior information during the induction process.

The results of these experiments are displayed graphically in Figure 10. This figure shows the reconstruction error of GSS-EP as a function of the number of random measurements performed for a representative subset of the ten digits contained in the MNIST data set (similar results are obtained for the digits not shown). The curves report the corresponding reconstruction errors when no prior information is actually used, when the prior information is used during the induction process, and when this information is chosen randomly. The figure shows that using the prior information for induction has a beneficial effect. In particular, it leads to a significant decrease of the reconstruction error in GSS-EP. The improvements obtained are more significant when the number of measurements used for induction is small. When the prior information is chosen randomly, we observe only a slight increase of the reconstruction error of GSS-EP, which is very small compared to the gains obtained when the prior information is correctly specified. In fact, it is difficult to visually differentiate from the results obtained when no prior information is actually used. Thus, introducing prior information can be certainly beneficial for the reconstruction error, whenever such information is correct. By contrast, it barely affects the resulting model when the information is inaccurate.

## 8. Conclusions

In this document we have described a method for carrying out feature selection at the group level in linear regression problems. This method is able to use prior information about groups of features that are expected to be jointly relevant or irrelevant for prediction. More precisely, it is based on a linear model that considers a generalized spike-and-slab prior for group feature selection. Specifically, under this prior a set of binary latent variables is introduced, one for each different group of features, and each latent variable indicates whether or not the corresponding group is used for prediction. Exact inference under this prior is infeasible in typical regression problems. However, expectation propagation (EP) can be used as a practical alternative to carry out approximate Bayesian inference. The computational cost of EP is in $O(n^2 d)$, where $n$ is the number of training instances and $d$ is the number of features. This linear cost with respect to $d$ is very efficient when $n \ll d$, which is the typical scenario we consider. Furthermore, the EP approximation provides an estimate of the posterior covariances of $\mathbf{w}$, that is, the vector of model coefficients. These covariances are shown to be very useful to carry out sequential experimental design in the linear regression model. In particular, they can be used to determine which instance to include in the training set to obtain the most information about $\mathbf{w}$, saving on costly experiments. The generalized spike-and-slab prior is also shown to be very useful to introduce prior knowledge about specific groups of features that are *a priori* expected to be more relevant or more irrelevant for prediction than the other groups. When this information is not available, the prior considered has only two hyper-parameters: $p_0$ and $v_0$. The interpretation of these parameters is very intuitive. They respectively describe the prior fraction of groups expected to be relevant for prediction and the prior variance of the model coefficients of

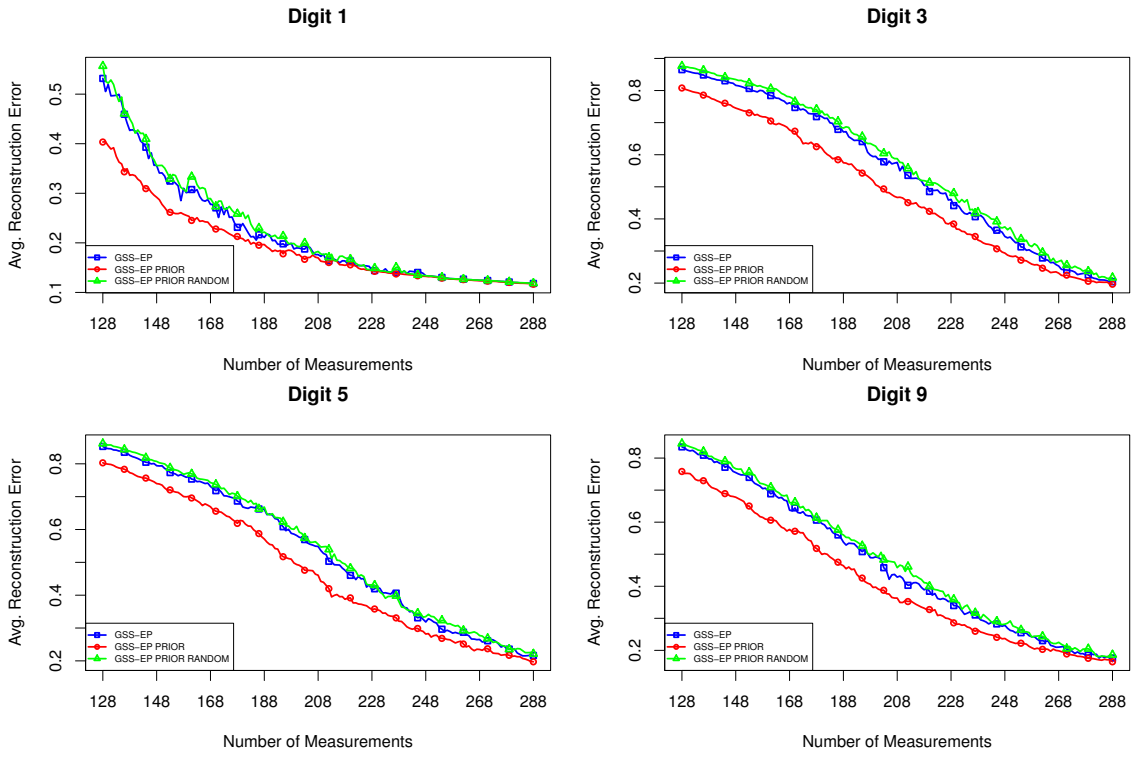



Figure 10: Reconstruction error for a representative subset of the ten digits contained in the MNIST data set as a function of the number of measurements carried out. We report results for GSS-EP when the measurements are chosen randomly. Furthermore, we consider different cases of prior information: (i) no prior information is used about the relevancy of each group of features (GSS-EP), (ii) the prior information is actually employed (GSS-EP PRIOR), and (iii) the prior information is chosen randomly (GSS-EP PRIOR RANDOM).

the relevant groups. Thus, unlike in other methods for group feature selection, in the generalized spike-and-slab prior it is very easy to specify the expected level of group sparsity and the expected deviation from zero of the relevant coefficients. If this information is available beforehand, or can be deduced from additional data, it can be readily introduced in the prior. Finally, the proposed method has the advantage of providing a posterior estimate of the importance of each group of features. This estimate can be used to identify the most relevant groups.

A detailed analysis compares the regularization properties of the generalized spike-and-slab prior considered in this document with the properties of the priors used in other methods that can also be used for group feature selection: namely, the group LASSO, the Bayesian group LASSO, the group horseshoe and the group ARD principle. This analysis shows that the generalized spike-and-slab prior is very effective for group feature selection. In particular, it is the only prior that can put a positive probability mass at the origin for the coefficients corresponding to the different groups. This probability is specified by the hyper-parameter $p_0$. This hyper-parameter determines the sparsity at the group level and mainly affects the regularization of irrelevant coefficients for prediction. The

smaller its value, the stronger the regularization of the model coefficients that actually take small values. By contrast, the regularization of the coefficients that are actually relevant for prediction are barely affected by $p_0$. These coefficients are fully regularized by the second hyper-parameter of this prior, $v_0$. The larger its value, the smaller the regularization of these coefficients and, as in the previous case, $v_0$ has little impact on the coefficients that are irrelevant for prediction. In summary, under the generalized spike-and-slab prior it is possible to provide solutions that are sparse at the group level in a selective manner. More precisely, under this prior we can model very high levels of sparsity at the group level (small values of $p_0$), while at the same time allowing for model coefficients that are significantly different from zero (large values of $v_0$). This is not possible, for example, in the case of the group LASSO or the Bayesian group LASSO. The group horseshoe also enjoys this selective shrinkage property. However, the resulting prior does not have a closed form convolution with the Gaussian distribution which makes difficult to apply the EP algorithm for fast approximate inference.

An extensive collection of experiments which considers real and synthetic data sets compares the performance of a model based on the generalized spike-and-slab prior and the EP algorithm with the other methods for group feature selection. In these experiments we also compare results when Gibbs sampling is used for approximate inference instead of EP. A model which does not use the grouping information for induction is also included in the comparison. Our results indicate that when accurate prior information about relevant or irrelevant groups of features for prediction is available, group feature selection significantly improves the results of single feature selection. In addition, from the models that use the grouping information for induction, the model based on the generalized spike-and-slab prior is shown to perform best. Furthermore, the performance of this model is very similar when the EP algorithm is used for induction or when Gibbs sampling is used instead. This confirms the accuracy of the EP approximation of the posterior distribution. Additionally, the computational cost of EP is significantly better than the computational cost of the methods based on Gibbs sampling, including the Bayesian group LASSO and the group horseshoe. The computational cost of the EP algorithm is also similar or better than the computational cost of the methods based on the group ARD principle or the group LASSO. Our results also show the utility of the EP algorithm in the proposed model for carrying out sequential experimental design. In particular, sequential experimental design in this model provides better results for a smaller number of training instances. Finally, our experiments show the benefits of introducing information about specific groups of features that are expected to be more relevant or more irrelevant *a priori* for prediction. When this prior information is introduced in the model based on the generalized spike-and-slab prior, better prediction results are obtained. By contrast, if this information is misspecified (chosen randomly), the prediction performance of the resulting model does not vary significantly.

A practical issue with the generalized spike-and-slab prior is the selection of the model hyper-parameters $p_0$ and $v_0$, which can be difficult. However, unlike the other methods for group feature selection, the interpretation of these parameters is very intuitive and they can be easily set by hand or chosen to match some specific model properties. Initially, we considered the model evidence, as approximated by the EP algorithm, for this purpose. Nevertheless, our results indicate that this is not adequate in some situations. Specifically, if the assumptions made by the model are not satisfied in practice, the model evidence can lead to wrong decisions. The more general technique of cross-validation is suggested instead. Another possibility, which is left for future exploration, is to specify hyper-priors for the model hyper-parameters and to learn them simultaneously alongside with the model coefficients $\mathbf{w}$. This has already been considered in the standard spike-and-slab

prior when Markov chain Monte Carlo is used for approximate inference (West, 2003). The model described is also restricted to work with non-overlapping groups. Thus, future research directions can also consider dealing with groups of features that overlap or that are considered to be relevant for prediction following some particular hierarchy. Another path for future investigation includes considering non-Gaussian additive noise in the estimation of the model parameters or developing some method for extracting the grouping information from the data.

## Acknowledgments

Daniel Hernández-Lobato and Pierre Dupont acknowledge support from the Spanish Dirección General de Investigación, project ALLS (TIN2010-21575-C02-02).

## Appendix A.

In this section we show how to implement an efficient Gibbs sampler for the regression models based on the generalized spike-and-slab prior, the Bayesian group LASSO and the group horseshoe prior. As described in Table 2, all these models are very similar and only differ in the prior distribution assumed for $\lambda_g^2$, that is, the prior variance for the coefficients $\mathbf{w}_g$ corresponding to the $g$-th group of features. In the case of the group horseshoe, the latent variances $\lambda_g^2$ are also multiplied by the squared value of the hyper-parameter $\tau$. For simplicity, we include this hyper-parameter in all the derivations of this section and assume that $\tau = 1$ for the Bayesian group LASSO and the generalized spike-and-slab prior. Consider the vector of latent variables $\boldsymbol{\lambda} = (\lambda_1^2, \ldots, \lambda_G^2)^{\mathrm{T}}$, where $G$ is the total number of groups. The method described in this section consists in first conditionally sampling each component of $\boldsymbol{\lambda}$ from the corresponding posterior distribution given $\mathbf{X}$ and $\mathbf{y}$. Then, $\mathbf{w}$ is sampled conditioned to each sample of $\boldsymbol{\lambda}$. Once this is done, the samples of $\boldsymbol{\lambda}$ are discarded and the samples of $\mathbf{w}$ are used to approximate the expectations with respect to the posterior distribution of $\mathbf{w}$. This is the approach followed by the Gibbs sampling algorithm described by George and McCulloch (1997) and by Lee et al. (2003) for the standard spike-and-slab algorithm. However, we incorporate some characteristics of the framework introduced by Tipping and Faul (2003) to speed-up the computations.

First, we describe how to conditionally sample from the posterior of $\lambda_g^2$, for $g = 1, \ldots, G$. Assume $\mathcal{P}(\lambda_g^2)$ is the corresponding prior distribution of $\lambda_g^2$. Then, the logarithm of the posterior of $\boldsymbol{\lambda} = (\lambda_1^2, \ldots, \lambda_G^2)^{\mathrm{T}}$ is:

$$\log \mathcal{P}(\boldsymbol{\lambda}|\mathbf{X}, \mathbf{y}) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{C}) + \sum_{g=1}^{G} \log \mathcal{P}(\lambda_g^2) + \text{constant}, \tag{37}$$

where $\mathbf{C} = \sigma_0^2 \mathbf{I} + \mathbf{X}\mathbf{A}\mathbf{X}^{\mathrm{T}}$ is an $n \times n$ matrix and $\mathbf{A}$ is a $d \times d$ diagonal matrix whose entries are defined as $A_{jj} = \tau^2 \lambda_g^2$ if the $j$-th feature belongs to the $g$-th group for $j = 1, \ldots, d$. Consider now that all the components of $\boldsymbol{\lambda}$ are fixed except for a particular $\lambda_g^2$ corresponding to the $g$-th group of features. Furthermore, denote by $\boldsymbol{\lambda}_{-g}$ the vector $(\lambda_1^2, \ldots, \lambda_G^2)^{\mathrm{T}}$ where the $g$-th component has been omitted. Then,

$$\log \mathcal{P}(\lambda_g^2|\mathbf{X}, \mathbf{y}, \boldsymbol{\lambda}_{-g}) = \mathcal{L}(\boldsymbol{\lambda}_{-g}) + \mathcal{L}(\lambda_g^2), \tag{38}$$

where

$$\mathcal{L}(\boldsymbol{\lambda}_{-g}) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{C}_{-g}) + \sum_{g' \neq g} \log \mathcal{P}(\lambda_{g'}^2) + \text{constant},$$

$\mathbf{C}_{-g} = \sigma_0^2 \mathbf{I} + \mathbf{X}\mathbf{A}_{-g}\mathbf{X}^{\mathrm{T}}$ is a $n \times n$ matrix and $\mathbf{A}_{-g}$ is a $d \times d$ diagonal matrix equal to $\mathbf{A}$, except for the diagonal entries corresponding to the $g$-th group of features. These entries are set to zero. Assume that $d_g$ is the size of the $g$-th group of features and that $\mathbf{X}_g$ is a $n \times d_g$ matrix which contains in each row, for each instance, only the features of the $g$-th group. Consider the $d_g \times d_g$ matrix $\mathbf{M} = \mathbf{X}_g^{\mathrm{T}} \mathbf{C}_{-g}^{-1} \mathbf{X}_g$. Denote by $s_j$ and by $\mathbf{e}_j$ the $j$-th eigenvalue and the $j$-th eigenvector of $\mathbf{M}$, respectively. Then,

$$\mathcal{L}(\lambda_g^2) = \sum_{j=1}^{d_g} \mathcal{L}_j(\lambda_g^2) + \log \mathcal{P}(\lambda_g^2), \tag{39}$$

where

$$\mathcal{L}_j(\lambda_g^2) = \frac{1}{2} \left( \frac{q_j^2 \tau^2 \lambda_g^2}{1 + \tau^2 \lambda_g^2 s_j} - \log\left(1 + \tau^2 \lambda_g^2 s_j\right) \right), \tag{40}$$

and $q_j = \mathbf{e}_j^{\mathrm{T}} \mathbf{y}^{\mathrm{T}} \mathbf{C}_{-g}^{-1} \mathbf{X}_g$. From (38) and (39) it follows that

$$\mathcal{P}(\lambda_g^2|\mathbf{X}, \mathbf{y}, \boldsymbol{\lambda}_{-g}) \propto \left[ \prod_{j=1}^{d_g} \exp(\mathcal{L}_j(\lambda_g^2)) \right] \mathcal{P}(\lambda_g^2). \tag{41}$$

We generate a Gibbs sampling of $\boldsymbol{\lambda} = (\lambda_1^2, \ldots, \lambda_G^2)^{\mathrm{T}}$ by running through all the components of this vector to generate a value for $\lambda_g^2$ according to the distribution in (41). Once the process of sampling each $\lambda_g^2$ has been completed, we consider that we have generated a single sample of $\boldsymbol{\lambda}$. This process is repeated until 11,000 samples of $\boldsymbol{\lambda}$ are generated. From these, the first 1,000 are discarded and the remaining are kept. Computing $s_j$ and $q_j$ for $j = 1, \ldots, d_g$ can be done very efficiently since we typically assume that $d_g$, that is, the size of the $g$-th group, is relatively small. Furthermore, instead of storing the matrices $\mathbf{C}_{-g}$ and $\mathbf{C}_{-g}^{-1}$ in memory, we exclusively work with the Cholesky decompositions of these matrices which are iteratively updated for each different value of $g$ in time $O(d_g n^2)$ (Gill et al., 1974). In particular, $\mathbf{C} = \mathbf{C}_{-g} + \tau^2 \lambda_g^2 \mathbf{X}_g \mathbf{X}_g^{\mathrm{T}}$ and $\mathbf{C}_{-g} = \mathbf{C} - \tau^2 \lambda_g^2 \mathbf{X}_g \mathbf{X}_g^{\mathrm{T}}$. Those represent $d_g$ rank-one updates of $\mathbf{C}_{-g}$ and $\mathbf{C}$, respectively. To avoid numerical errors, the Cholesky decompositions are recomputed from scratch each time ten new Gibbs samples of $\boldsymbol{\lambda}$ are generated. Finally, since $\sum_{g=1}^{G} d_g = d$, the cost of generating one sample of $\boldsymbol{\lambda}$ is $O(n^2 d)$, where $d$ is the total number of features.

Sampling from the distribution described in (41) for $g = 1, \ldots, G$ is straight-forward in the case of the generalized spike-and-slab prior as a consequence of the simpler form of $\mathcal{P}(\lambda_g^2)$. For the Bayesian LASSO and the group horseshoe, the method described by Damien et al. (1999) is used. For this, we sample $d_g$ auxiliary latent variables $u_j$, with $j = 1, \ldots, d_g$ so that $\exp(u_j) \sim \mathcal{U}[0, \exp(\mathcal{L}_j(\lambda_g^2)]$ where $\mathcal{L}_j(\cdot)$ is defined as in (40). Then, we sample $\lambda_g^2$ from $\mathcal{P}(\lambda_g^2)$ but restricted to the set $\mathcal{A}_u = \cap_{j=1}^{d_g} \mathcal{A}_{u_j}$, where $\mathcal{A}_{u_j} = \{\lambda_g^2 : \mathcal{L}_j(\lambda_g^2) > u_j\}$. Note that each function $\mathcal{L}_j(\cdot)$ has a single global maximum (Faul and Tipping, 2001), which is found at zero when $q_j^2 < s_j$, and at

$(q_j^2 - s_j)/(\tau^2 s_j^2)$ otherwise. Let $\tilde{\lambda}_g^2(j)$ be the global maximum of $\mathcal{L}_j(\cdot)$. Each set $\mathcal{A}_{u_j}$ can be identified by finding the roots of $\mathcal{L}_j(\lambda_g^2) - u_j$ in the intervals $[0, \tilde{\lambda}_g^2(j)]$ and $[\tilde{\lambda}_g^2(j), \infty]$. If $\mathcal{A}_u$ is found to be empty (this rarely occurs in practice) we do not sample a new value for $\lambda_g^2$ and use the previous one. Finally, in the group horseshoe and the Bayesian group LASSO, $\boldsymbol{\lambda}$ is initialized to a vector whose components are all equal to 1 as described by Scott (2010). In the sampler for the model based on the generalized spike-and-slab prior, $\boldsymbol{\lambda}$ is initialized to contain only $\rho_0 G$ components different from zero and equal to $v_0$. The chosen components are those with the smallest least squares training error.

Given the samples of the vector $\boldsymbol{\lambda}$, we sample from the conditional posterior of $\mathbf{w}$ to approximate the posterior distribution of this vector. Conditioning on $\boldsymbol{\lambda}$, we can sample $\mathbf{w}$ from a Gaussian distribution with covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\lambda}} = \mathbf{A} + 1/\sigma_0^2 \mathbf{X}^{\mathrm{T}}\mathbf{X}$ and mean vector $1/\sigma_0^2 \boldsymbol{\Sigma}_{\boldsymbol{\lambda}} \mathbf{X}\mathbf{y}$. When $d \gg n$, this procedure has a cost in $O(n^2 d)$. See Appendix B.2 of Seeger (2008). The total cost of Gibbs sampling is in $O(kn^2 d)$, where $k$ is the number of samples to be generated from the posterior distribution. However, often $k \gg d$ for accurate inference.

## Appendix B.

In this section we show how to implement the group ARD method for group feature selection. Section 5.4 shows that this method consists in finding the maximum *a posteriori* (MAP) solution of $\alpha_g$, for $g = 1, \ldots, G$, where $\alpha_g$ is the inverse of the prior Gaussian variance of $\mathbf{w}_g$, that is, the model coefficients of the $g$-th group, and $G$ is the total number of groups. A uniform prior for each $\alpha_g$ is assumed in this process. For simplicity, we use here the notation of Appendix A and work with $\lambda_g^2 = \alpha_g^{-1}$. Consider the vector of latent variables $\boldsymbol{\lambda} = (\lambda_1^2, \ldots, \lambda_G^2)^{\mathrm{T}}$. From (37) we have that

$$\log \mathcal{P}(\boldsymbol{\lambda}|\mathbf{X}, \mathbf{y}) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{C}) + \text{constant}, \tag{42}$$

where $\mathbf{C} = \sigma_0^2 \mathbf{I} + \mathbf{X}\mathbf{A}\mathbf{X}^{\mathrm{T}}$ is a $n \times n$ matrix and $\mathbf{A}$ is a $d \times d$ diagonal matrix whose entries are defined as $A_{jj} = \lambda_g^2$ if the $j$-th feature belongs to the $g$-th group for $j = 1, \ldots, d$. To find the maximum of (42) we use a coordinate ascent method. Consider that all the components of $\boldsymbol{\lambda}$ are fixed except for a particular $\lambda_g^2$ corresponding to the $g$-th group of features. Furthermore, denote by $\boldsymbol{\lambda}_{-g}$ the vector $(\lambda_1^2, \ldots, \lambda_G^2)^{\mathrm{T}}$, where the $g$-th component has been omitted. From (38) and (39), it follows that

$$\log \mathcal{P}(\lambda_g^2|\mathbf{X}, \mathbf{y}, \boldsymbol{\lambda}_{-g}) = \sum_{j=1}^{d_g} \mathcal{L}_j(\lambda_g^2) + \text{constant}, \tag{43}$$

where $d_g$ is the number of features in the $g$-th group and $\mathcal{L}_j(\lambda_g^2)$ is defined as in (40) for $\tau^2 = 1$. We can optimize (42) with respect to $\boldsymbol{\lambda}$ by iteratively optimizing (43) for $g = 1, \ldots, G$. However, unlike the single feature selection method (Faul and Tipping, 2001), the maximum of this function has no analytical solution (Ji et al., 2009). Thus, we have to resort to non-linear optimization to find the location of the maximum. For this, the gradient of $\mathcal{L}_j(\lambda_g^2)$ with respect to $\lambda_g^2$ can be very useful:

$$\frac{d\mathcal{L}_j(\lambda_g^2)}{d\lambda_g^2} = \frac{1}{2} \sum_{j=1}^{d_g} \left( \frac{q_j^2 - s_j - \lambda_g^2 s_j^2}{(1 + \lambda_g^2 s_j)^2} \right),$$

where $q_j$ and $s_j$ are defined as in (40).

In our implementation each $\lambda_g^2$ is initialized to 0. Furthermore, as in Appendix A, instead of storing the matrices $\mathbf{C}_{-g}$ and $\mathbf{C}_{-g}^{-1}$ in memory, we exclusively work with the Cholesky decompositions of these matrices which are iteratively updated for each different value of $g$ in time $O(d_g n^2)$

(Gill et al., 1974). We take care of doing the updates of these matrices only when they are strictly needed. For example, if the optimal $\lambda_g^2$ is found to be equal to zero then $\mathbf{C} = \mathbf{C}_{-g}$. The total cost of the algorithm described is in $O(n^2 d)$ under the assumption that the number of relevant groups is in $O(d)$, where $d$ is the total number of features.

## References

H. Attias. A variational Bayesian framework for graphical models. In *In Advances in Neural Information Processing Systems*, volume 12, pages 209–215. MIT Press, 2000.

F. R. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008. ISSN 1532-4435.

J. A. Bazerque, G. Mateos, and G. B. Giannakis. Group-lasso on splines for spectrum cartography. *Transactions on Signal Proccesing*, 59(10):4648–4663, October 2011. ISSN 1053-587X.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, August 2006. ISBN 0387310738.

J. Blitzer, M. Dredze, and F. Pereira. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–407, Prague, Czech Republic, 2007.

E. J. Candes and M. B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21 –30, march 2008. ISSN 1053-5888.

C. M. Carvalho, N. G. Polson, and J. G. Scott. Handling sparsity via the horseshoe. *Journal of Machine Learning Research W&CP*, 5:73–80, 2009.

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10: 273–304, 1995.

P. Damien, J. Wakefield, and S. Walker. Gibbs sampling for Bayesian non-conjugate and hierarchical models by using auxiliary variables. *Journal of the Royal Statistical Society Series B*, 61(2): 331–344, 1999.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006. ISSN 1533-7928.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

A. C. Faul and M. E. Tipping. Analysis of sparse Bayesian learning. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 383–389. MIT Press, 2001.

V. Fedorov. *Theory of Optimal Experiments*. Academic Press, 1972.

E. I. George and R. E. McCulloch. Approaches for Bayesian variable selection. *Statistica Sinica*, 7 (2):339–373, 1997.

M. Van Gerven, B. Cseke, R. Oostenveld, and T. Heskes. Bayesian source localization with the multivariate Laplace prior. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1901–1909, 2009.

J. Geweke. Variable selection and model comparison in regression. *Bayesian Statistics*, 5:609–620, 1996.

P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974.

D. Hernández-Lobato. *Prediction Based on Averages over Automatically Induced Learners: Ensemble Methods and Bayesian Techniques*. PhD thesis, Computer Science Department, Universidad Autónoma de Madrid, 2009. Online available at: http://arantxa.ii.uam.es/d̃hernan/docs/Thesis_color_links.pdf.

D. Hernández-Lobato, J. M. Hernández-Lobato, T. Helleputte, and P. Dupont. Expectation propagation for Bayesian multi-task feature selection. In José L. Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, *Proceedings of the European Conference on Machine Learning*, volume 6321, pages 522–537. Springer, 2010.

J. M. Hernández-Lobato. *Balancing Flexibility and Robustness in Machine Learning: Semiparametric Methods and Sparse Linear Models*. PhD thesis, Computer Science Department, Universidad Autónoma de Madrid, 2010.

J. M. Hernández-Lobato and T. Dijkstra. Hub gene selection methods for the reconstruction of transcription networks. In José L. Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, *Machine Learning and Knowledge Discovery in Databases, European Conference, Proceedings, Part I*, volume 6321, pages 506–521, 2010.

J. M. Hernández-Lobato, T. Dijkstra, and T. Heskes. Regulator discovery from gene expression time series of malaria parasites: A hierarchical approach. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 649–656. The MIT Press, 2008.

J. M. Hernández-Lobato, D. Hernández-Lobato, and A. Suárez. Network-based sparse Bayesian classification. *Pattern Recognition*, 44:886–900, 2011.

J. Huang and T. Zhang. The benefit of group sparsity. *The Annals of Statistics*, 38:1978–2004, 2010.

T. S. Jaakkola. Tutorial on variational approximation methods. In M. Opper and D. Saad, editors, *Advances in Mean Field Methods: Theory and Practice*, pages 129–159. MIT Press, 2001.

S. Ji and L. Carin. Bayesian compressive sensing and projection optimization. In *Proceedings of the 24th International Conference on Machine Learning*, pages 377–384. ACM, 2007. ISBN 978-1-59593-793-3.

S. Ji, Y. Xue, and L. Carin. Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56(6):2346 –2356, june 2008. ISSN 1053-587X.

S. Ji, D. Dunson, and L. Carin. Multitask compressive sensing. *IEEE Transactions on Signal Processing*, 57(1):92 –106, January 2009. ISSN 1053-587X.

I. M. Johnstone and D. M. Titterington. Statistical challenges of high-dimensional data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367 (1906):4237, 2009.

H. J. Kappen and Vicenç Gómez. The variational garrote. *Machine Learning*, pages 1–17, 2013. Submitted.

S. Kim and E. P. Xing. Feature selection via block-regularized regression. In David A. McAllester and Petri Myllymäki, editors, *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence*, pages 325–332. AUAI Press, 2008.

Y. Kim, J. Kim, and Y. Kim. Blockwise sparse regression. *Statistica Sinica*, 16(2):375, 2006. ISSN 1017-0405.

N. Lawrence, M. W. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 609–616. MIT Press, Cambridge, MA, 2003.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

K. E. Lee, N. Sha, E. R. Dougherty, M. Vannucci, and B. K. Mallick. Gene selection: A Bayesian variable selection approach. *Bioinformatics*, 19(1):90–97, 2003.

Y. Li, C. Campbell, and M. Tipping. Bayesian automatic relevance determination algorithms for classifying gene expression data. *Bioinformatics*, 18(10):1332–1339, 2002.

D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4:590–604, 1991.

D. J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, Cambridge, UK., 2003. ISBN 0521642981.

D. M. Malioutov, M. Çetin, and A. S. Willsky. A sparse signal reconstruction perspective for source localization with sensor arrays. *IEEE Transactions on Signal Processing*, 53(8-2):3010–3022, 2005.

L. Meier, S. Van De Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008. ISSN 1467-9868.

T. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

T. Minka and J. Lafferty. Expectation-propagation for the generative aspect model. In Adnan Darwiche and Nir Friedman, editors, *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 352–359. Morgan Kaufmann, 2002.

T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993.

H. Nickisch and C. E. Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9:2035–2078, 10 2008.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002.

A. T. Puig, A. Wiesel, G. F., and A. O. Hero. Multidimensional shrinkage-thresholding operator and group lasso penalties. *IEEE Signal Processing Letters*, 18(6):363–366, 2011.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. URL `http://www.R-project.org/`. ISBN 3-900051-07-0.

S. Raman, T. J. Fuchs, P. J. Wild, E. Dahl, and V. Roth. The Bayesian group-lasso for analyzing contingency tables. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 881–888, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1.

V. Roth and B. Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In A. McCallum and S. Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning*, pages 848–855, 2008.

T. Sandler, J. Blitzer, P. P. Talukdar, and L. H. Ungar. Regularized learning with networks of features. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1401–1408. 2009.

F. Scheipl, L. Fahrmeir, and T. Kneib. Spike-and-slab priors for function selection in structured additive regression models. *Journal of the American Statistical Association*, 107:1518–1532, 2012.

J. G. Scott. Parameter expansion in local-shrinkage models. *ArXiv E-prints*, 2010. arXiv:1010.5265v1.

M. W. Seeger. Bayesian inference and optimal design for the sparse linear model. *Journal of Machine Learning Research*, 9:759–813, 2008.

M. W. Seeger, H. Nickisch, R. Pohmann, and B. Schölkopf. Optimization of k-space trajectories for compressed sensing by Bayesian experimental design. *Magnetic Resonance in Medicine*, 63(1): 116–126, 2009.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

M. E. Tipping and A. Faul. Fast marginal likelihood maximisation for sparse Bayesian models. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, pages 3–6, 2003.

J. E. Vogt and V. Roth. The group-lasso: $\ell_{1,\infty}$ regularization versus $\ell_{1,2}$ regularization. In Goesele et al., editor, *32nd Anual Symposium of the German Association for Pattern Recognition*, volume 6376, pages 252–261. Springer, 2010.

M. Wakin, M. Duarte, S. Sarvotham, D. Baron, and R. Baraniuk. Recovery of jointly sparse signals from few random projections. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1433–1440. MIT Press, Cambridge, MA, 2006.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. In *Bayesian Statistics 7*, pages 723–732. Oxford University Press, 2003.

T. J. Yen and Y. M. Yen. Grouped variable selection via nested spike and slab priors. *ArXiv E-prints*, 2011. arXiv:1106.5837v1.

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. ISSN 1467-9868.

H. Zhu and R. Rohwer. Bayesian invariant measurements of generalization. *Neural Processing Letters*, 2:28–31, 1995. ISSN 1370-4621.

Journal of Machine Learning Research 14 (2013) 1947-1988 Submitted 6/12; Revised 1/13; Published 7/13

# Alleviating Naive Bayes Attribute Independence Assumption by Attribute Weighting

**Nayyar A. Zaidi** NAYYAR.ZAIDI@MONASH.EDU
*Faculty of Information Technology*
*Monash University*
*VIC 3800, Australia*

**Jesús Cerquides** CERQUIDE@IIIA.CSIC.ES
*IIIA-CSIC, Artificial Intelligence Research Institute*
*Spanish National Research Council*
*Campus UAB*
*08193 Bellaterra, Spain*

**Mark J. Carman** MARK.CARMAN@MONASH.EDU
**Geoffrey I. Webb** GEOFF.WEBB@MONASH.EDU
*Faculty of Information Technology*
*Monash University*
*VIC 3800, Australia*

**Editor:** Russ Greiner

## Abstract

Despite the simplicity of the Naive Bayes classifier, it has continued to perform well against more sophisticated newcomers and has remained, therefore, of great interest to the machine learning community. Of numerous approaches to refining the naive Bayes classifier, attribute weighting has received less attention than it warrants. Most approaches, perhaps influenced by attribute weighting in other machine learning algorithms, use weighting to place more emphasis on highly predictive attributes than those that are less predictive. In this paper, we argue that for naive Bayes attribute weighting should instead be used to alleviate the conditional independence assumption. Based on this premise, we propose a weighted naive Bayes algorithm, called WANBIA, that selects weights to minimize either the negative conditional log likelihood or the mean squared error objective functions. We perform extensive evaluations and find that WANBIA is a competitive alternative to state of the art classifiers like Random Forest, Logistic Regression and A1DE.

**Keywords:** classification, naive Bayes, attribute independence assumption, weighted naive Bayes classification

## 1. Introduction

Naive Bayes (also known as *simple Bayes* and *Idiot's Bayes*) is an extremely simple and remarkably effective approach to classification learning (Lewis, 1998; Hand and Yu, 2001). It infers the probability of a class label given data using a simplifying assumption that the attributes are independent given the label (Kononenko, 1990; Langley et al., 1992). This assumption is motivated by the need to estimate high-dimensional multi-variate probabilities from the training data. If there is sufficient data present for every possible combination of attribute values, direct estimation of each relevant multi-variate probability will be reliable. In practice, however, this is not the case and most com-

©2013 Nayyar A. Zaidi, Jesús Cerquides, Mark J. Carman and Geoffrey I. Webb.

binations are either not represented in the training data or not present in sufficient numbers. Naive Bayes circumvents this predicament by its conditional independence assumption. Surprisingly, it has been shown that the prediction accuracy of naive Bayes compares very well with other more complex classifiers such as decision trees, instance-based learning and rule learning, especially when the data quantity is small (Hand and Yu, 2001; Cestnik et al., 1987; Domingos and Pazzani, 1996; Langley et al., 1992).

In practice, naive Bayes' attribute independence assumption is often violated, and as a result its probability estimates are often suboptimal. A large literature addresses approaches to reducing the inaccuracies that result from the conditional independence assumption. Such approaches can be placed into two categories. The first category comprises *semi-naive Bayes methods*. These methods are aimed at enhancing naive Bayes' accuracy by relaxing the assumption of conditional independence between attributes given the class label (Langley and Sage, 1994; Friedman and Goldszmidt, 1996; Zheng et al., 1999; Cerquides and De Mántaras, 2005a; Webb et al., 2005, 2011; Zheng et al., 2012). The second category comprises *attribute weighting methods* and has received relatively little attention (Hilden and Bjerregaard, 1976; Ferreira et al., 2001; Hall, 2007). There is some evidence that attribute weighting appears to have primarily been viewed as a means of increasing the influence of highly predictive attributes and discounting attributes that have little predictive value. This is not so much evident from the explicit motivation stated in the prior work, but rather from the manner in which weights have been assigned. For example, weighting by mutual information between an attribute and the class is directly using a measure of how predictive is each individual attribute (Zhang and Sheng, 2004). In contrast, we argue that the primary value of attribute weighting is its capacity to reduce the impact on prediction accuracy of violations of the assumption of conditional attribute independence.

Contributions of this paper are two-fold:

- This paper reviews the state of the art in weighted naive Bayesian classification. We provide a compact survey of existing techniques and compare them using the bias-variance decomposition method of Kohavi and Wolpert (1996). We also use Friedman test and Nemenyi statistics to analyze error, bias, variance and root mean square error.

- We present novel algorithms for learning attribute weights for naive Bayes. It should be noted that the motivation of our work differs from most previous *attribute weighting methods*. We view weighting as a way to reduce the effects of the violations of the attribute independence assumption on which naive Bayes is based. Also, our work differs from *semi-naive Bayes methods*, as we weight the attributes rather than modifying the structure of naive Bayes.

We propose a weighted naive Bayes algorithm, Weighting attributes to Alleviate Naive Bayes' Independence Assumption (WANBIA), that introduces weights in naive Bayes and learns these weights in a discriminative fashion that is minimizing either the negative conditional log likelihood or the mean squared error objective functions. Naive Bayes probabilities are set to be their maximum a posteriori (MAP) estimates.

The paper is organized as follows: we provide a formal description of the weighted naive Bayes model in Section 2. Section 3 provides a survey of related approaches. Our novel techniques for learning naive Bayes weights are described in Section 4 where we also discuss their connection with naive Bayes and Logistic Regression in terms of parameter optimization. Section 5 presents experimental evaluation of our proposed methods and their comparison with related approaches. Section 6 presents conclusions and directions for future research.

| Notation | Description |
|---|---|
| $\mathrm{P}(e)$ | the unconditioned probability of event $e$ |
| $\mathrm{P}(e \mid g)$ | conditional probability of event $e$ given $g$ |
| $\hat{\mathrm{P}}(\bullet)$ | an estimate of $\mathrm{P}(\bullet)$ |
| $a$ | the number of attributes |
| $n$ | the number of data points in $\mathcal{D}$ |
| $\mathbf{x} = \langle x_1, \ldots, x_a \rangle$ | an object ($a$-dimensional vector) and $\mathbf{x} \in \mathcal{D}$ |
| $y \in \mathcal{Y}$ | the class label for object $\mathbf{x}$ |
| $\lvert\mathcal{Y}\rvert$ | the number of classes |
| $\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$ | data consisting of $n$ objects |
| $\mathcal{L} = \{y^{(1)}, \ldots, y^{(n)}\}$ | labels of data points in $\mathcal{D}$ |
| $\mathcal{X}_i$ | discrete set of values for attribute $i$ |
| $\lvert\mathcal{X}_i\rvert$ | the cardinality of attribute $i$ |
| $v = \frac{1}{a}\sum_i \lvert\mathcal{X}_i\rvert$ | the average cardinality of the attributes |

Table 1: List of symbols used

## 2. Weighted Naive Bayes

We wish to estimate from a training sample $\mathcal{D}$ consisting of $n$ objects, the probability $\mathrm{P}(y \mid \mathbf{x})$ that an example $\mathbf{x} \in \mathcal{D}$ belongs to a class with label $y \in \mathcal{Y}$. All the symbols used in this work are listed in Table 1. From the definition of conditional probability we have

$$\mathrm{P}(y \mid \mathbf{x}) = \mathrm{P}(y, \mathbf{x}) / \mathrm{P}(\mathbf{x}). \tag{1}$$

As $\mathrm{P}(\mathbf{x}) = \sum_{i=1}^{|\mathcal{Y}|} \mathrm{P}(y_i, \mathbf{x})$, we can always estimate $\mathrm{P}(y|\mathbf{x})$ in Equation 1 from the estimates of $\mathrm{P}(y, \mathbf{x})$ for each class as:

$$\mathrm{P}(y, \mathbf{x}) / \mathrm{P}(\mathbf{x}) = \frac{\mathrm{P}(y, \mathbf{x})}{\sum_{i=1}^{|\mathcal{Y}|} \mathrm{P}(y_i, \mathbf{x})}. \tag{2}$$

In consequence, in the remainder of this paper we consider only the problem of estimating $\mathrm{P}(y, \mathbf{x})$.

Naive Bayes estimates $\mathrm{P}(y, \mathbf{x})$ by assuming the attributes are independent given the class, resulting in the following formula:

$$\hat{\mathrm{P}}(y, \mathbf{x}) = \hat{\mathrm{P}}(y) \prod_{i=1}^{a} \hat{\mathrm{P}}(x_i \mid y). \tag{3}$$

Weighted naive Bayes extends the above by adding a weight to each attribute. In the most general case, this weight depends on the attribute value:

$$\hat{\mathrm{P}}(y, \mathbf{x}) = \hat{\mathrm{P}}(y) \prod_{i=1}^{a} \hat{\mathrm{P}}(x_i|y)^{w_{i,x_i}}. \tag{4}$$

Doing this results in $\sum_i^a |\mathcal{X}_i|$ weight parameters (and is in some cases equivalent to a "binarized logistic regression model" see Section 4 for a discussion). A second possibility is to give a single weight per attribute:

$$\hat{\mathrm{P}}(y, \mathbf{x}) = \hat{\mathrm{P}}(y) \prod_{i=1}^{a} \hat{\mathrm{P}}(x_i \mid y)^{w_i}. \tag{5}$$

One final possibility is to set all weights to a single value:

$$\hat{\mathrm{P}}(y,\mathbf{x}) = \hat{\mathrm{P}}(y)\left(\prod_{i=1}^{a}\hat{\mathrm{P}}(x_i\,|\,y)\right)^{w}. \tag{6}$$

Equation 5 is a special case of Equation 4, where $\forall_{i,j}\ w_{ij} = w_i$, and Equation 6 is a special case of Equation 5 where $\forall_i\ w_i = w$. Unless explicitly stated, in this paper we intend the intermediate form when we refer to attribute weighting, as we believe it provides an effective trade-off between computational complexity and inductive power.

Appropriate weights can reduce the error that results from violations of naive Bayes' conditional attribute independence assumption. Trivially, if data include a set of $a$ attributes that are identical to one another, the error due to the violation of the conditional independence assumption can be removed by assigning weights that sum to 1.0 to the set of attributes in the set. For example, the weight for one of the attributes, $x_i$ could be set to 1.0, and that of the remaining attributes that are identical to $x_i$ set to 0.0. This is equivalent to deleting the remaining attributes. Note that, any assignment of weights such that their sum is 1.0 for the $a$ attributes will have the same effect, for example, we could set the weights of all $a$ attributes to $1/a$.

Attribute weighting is strictly more powerful than attribute selection, as it is possible to obtain identical results to attribute selection by setting the weights of selected attributes to 1.0 and of discarded attributes to 0.0, and assignment of other weights can create classifiers that cannot be expressed using attribute selection.

### 2.1 Dealing with Dependent Attributes by Weighting: A Simple Example

This example shows the relative performance of naive Bayes and weighted naive Bayes as we vary the conditional dependence between attributes. In particular it demonstrates how optimal assignment of weights will never result in higher error than attribute selection or standard naive Bayes, and that for certain violations of the attribute independence assumption it can result in lower error than either.

We will constrain ourselves to a binary class problem with two binary attributes. We quantify the conditional dependence between the attributes using the Conditional Mutual Information (CMI):

$$I(X_1,X_2|Y) \quad = \quad \sum_{y}\sum_{x_2}\sum_{x_1}\mathrm{P}(x_1,x_2,y)\log\frac{\mathrm{P}(x_1,x_2|y)}{\mathrm{P}(x_1|y)\mathrm{P}(x_2|y)}.$$

The results of varying the conditional dependence between the attributes on the performance of the different classifiers in terms of their Root Mean Squared Error (RMSE) is shown in Figure 1.

To generate these curves, we varied the probabilities $\mathrm{P}(y|x_1,x_2)$ and $\mathrm{P}(x_1,x_2)$ and plotted average results across distinct values of the Conditional Mutual Information. For each of the 4 possible attribute value combinations $(x_1,x_2) \in \{(0,0),(0,1),(1,0),(1,1)\}$, we selected values for the class probability given the attribute value combination from the set: $\mathrm{P}(y|x_1,x_2) \in \{0.25,0.75\}$. Note that $\mathrm{P}(\neg y|x_1,x_2) = 1-P(y|x_1,x_2)$, so this process resulted in $2^4$ possible assignments to the vector $\mathrm{P}(y|\bullet,\bullet)$.

We then set the values for the attribute value probabilities $\mathrm{P}(x_1,x_2)$ by fixing the marginal distributions to a half $\mathrm{P}(x_1) = \mathrm{P}(x_2) = 1/2$, and varying the correlation between the attributes using

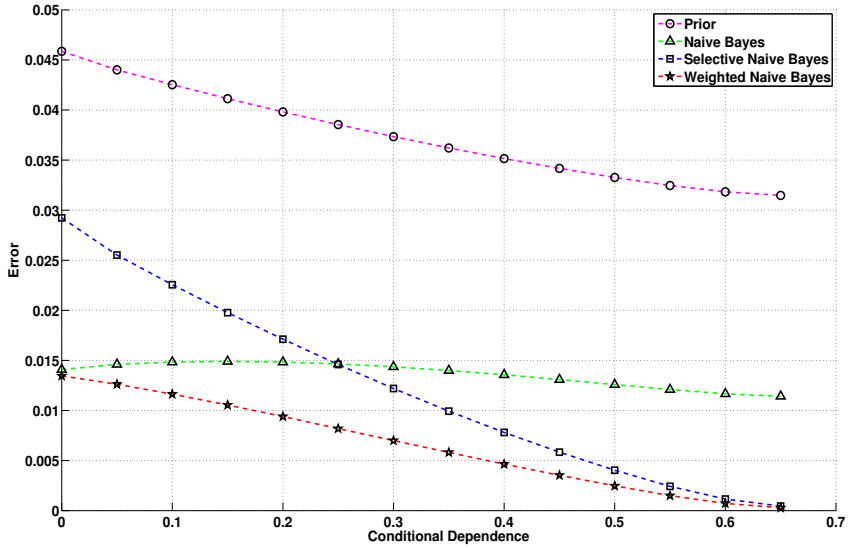



Figure 1: Variation of Error of naive Bayes, selective naive Bayes, weighted naive Bayes and classifier based only on prior probabilities of the class as a function of conditional dependence (conditional mutual information) between the two attributes.

Pearson's correlation coefficient, denoted $\rho$, as follows:[1]

$$\begin{aligned} P(X_1=0,X_2=0) &= P(X_1=1,X_2=1) = \frac{(1+\rho)}{4}, \\ P(X_1=0,X_2=1) &= P(X_1=1,X_2=0) = \frac{(1-\rho)}{4}, \\ & \text{where } -1 \leq \rho \leq 1. \end{aligned}$$

Note that when $\rho=-1$ the attributes are perfectly anti-correlated ($x_1=\neg x_2$), when $\rho=0$ the attributes are independent (since the joint distribution $P(x_1,x_2)$ is uniform) and when $\rho=1$ the attributes are perfectly correlated.

For the graph, we increased values of $\rho$ in increments of 0.00004, resulting in 50000 distributions (vectors) for $P(\bullet,\bullet)$ for each vector $P(y|\bullet,\bullet)$. Near optimal weights $(w_1,w_2)$ for the weighted naive Bayes classifier were found using grid search over the range $\{\{0.0,0.1,0.2,\ldots,0.9,1.0\} \times \{0.0,0.1,0.2,\ldots,0.9,1.0\}\}$. Results in Figure 1 are plotted by taking average across conditional mutual information values, with a window size of 0.1.

1. Note that from the definition of Pearson's correlation coefficient we have:

$$\rho = \frac{E[(X_1-E[X_1])(X_2-E[X_2])]}{\sqrt{E[(X_1-E[X_1])^2]E[(X_2-E[X_2])^2]}} = 4P(X_1=1,X_2=1)-1,$$

since $E[X_1]=E[X_2]=P(1)=1/2$ and $E[X_1X_2]=P(X_1=1,X_2=1)$.

We compare the expected RMSE of naive Bayes ($w_1 = 1, w_2 = 1$), weighted naive Bayes, naive Bayes based on feature 1 only (selective Bayes with $w_1 = 1, w_2 = 0$), naive Bayes based on feature 2 only (selective Bayes with $w_1 = 0, w_2 = 1$), and naive Bayes using only the prior (equivalent to weighted naive Bayes with both weights set to 0.0). It can be seen that when conditional mutual information (CMI) is small, naive Bayes performs better than selective naive Bayes and the prior classifier. Indeed, when CMI is 0.0, naive Bayes is optimal. As CMI is increased, naive Bayes performance deteriorates compared to selective naive Bayes. Weighted naive Bayes, on the other hand, has the best performance in all circumstances. Due to the symmetry of the problem, the two selective Bayes classifiers give exactly the same results.

Note that in this experiment we have used the optimal weights to calculate the results. We have shown that weighted naive Bayes is capable of *expressing* more accurate classifiers than selective naive Bayes. In the remaining sections we will examine and evaluate techniques for learning from data the weights those models require.

## 3. Survey of Attribute Weighting and Selecting Methods for Naive Bayes

*Attribute weighting* is well-understood in the context of nearest-neighbor learning methods and is used for reducing bias in high-dimensional problems due to the presence of redundant or irrelevant features (Friedman, 1994; Guyon et al., 2004). It is also used for mitigating the effects of the curse-of-dimensionality which results in exponential increase in the required training data as the number of features are increased (Bellman, 1957). Attribute weighting for naive Bayes is comparatively less explored.

Before discussing these techniques, however, it is useful to briefly examine the closely related area of feature selection for naive Bayes. As already pointed out, weighting can achieve feature selection by settings weights to either 0.0 or 1.0, and so can be viewed as a generalization of feature selection.

Langley and Sage (1994) proposed the Selective Bayes (SB) classifier, using feature selection to accommodate redundant attributes in the prediction process and to augment naive Bayes with the ability to exclude attributes that introduce dependencies. The technique is based on searching through the entire space of all attribute subsets. For that, they use a forward sequential search with a greedy approach to traverse the search space. That is, the algorithm initializes the subset of attributes to an empty set, and the accuracy of the resulting classifier, which simply predicts the most frequent class, is saved for subsequent comparison. On each iteration, the method considers adding each unused attribute to the subset on a trial basis and measures the performance of the resulting classifier on the training data. The attribute that most improves the accuracy is permanently added to the subset. The algorithm terminates when addition of any attribute results in reduced accuracy, at which point it returns the list of current attributes along with their ranks. The rank of the attribute is based on the order in which they are added to the subset.

Similar to Langley and Sage (1994), Correlation-based Feature Selection (CFS) used a correlation measure as a metric to determine the relevance of the attribute subset (Hall, 2000). It uses a best-first search to traverse through feature subset space. Like SB, it starts with an empty set and generates all possible single feature expansions. The subset with highest evaluation is selected and expanded in the same manner by adding single features. If expanding a subset results in no improvement, the search drops back to the next best unexpanded subset and continues from there.

The best subset found is returned when the search terminates. CFS uses a stopping criterion of five consecutive fully expanded non-improving subsets.

There has been a growing trend in the use of decision trees to improve the performance of other learning algorithms and naive Bayes classifiers are no exception. For example, one can build a naive Bayes classifier by using only those attributes appearing in a C4.5 decision tree. This is equivalent to giving zero weights to attributes not appearing in the decision tree. The Selective Bayesian Classifier (SBC) of Ratanamahatana and Gunopulos (2003) also employs decision trees for attribute selection for naive Bayes. Only those attributes appearing in the top three levels of a decision tree are selected for inclusion in naive Bayes. Since decision trees are inherently unstable, five decision trees (C4.5) are generated on samples generated by bootstrapping 10% from the training data. Naive Bayes is trained on an attribute set which comprises the union of attributes appearing in all five decision trees.

One of the earliest works on weighted naive Bayes is by Hilden and Bjerregaard (1976), who used weighting of the form of Equation 6. This strategy uses a single weight and therefore is not strictly performing attribute weighting. Their approach is motivated as a means of alleviating the effects of violations of the attribute independence assumption. Setting $w$ to unity is appropriate when the conditional independence assumption is satisfied. However, on their data set (acute abdominal pain study in Copenhagen by Bjerregaard et al. 1976), improved classification was obtained when $w$ was small, with an optimum value as low as 0.3. The authors point out that if symptom variables of a clinical field trial are not independent, but pair-wise correlated with independence between pairs, then $w = 0.5$ will be the correct choice since using $w = 1$ would make all probabilities the square of what they ought be. Looking at the optimal value of $w = 0.3$ for their data set, they suggested that out of ten symptoms, only three are providing independent information. The value of $w$ was obtained by maximizing the log-likelihood over the entire testing sample.

Zhang and Sheng (2004) used the gain ratio of an attribute with the class labels as its weight. Their formula is shown in Equation 7. The gain ratio is a well-studied attribute weighting technique and is generally used for splitting nodes in decision trees (Duda et al., 2006). The weight of each attribute is set to the gain ratio of the attribute relative to the average gain ratio across all attributes. Note that, as a result of the definition at least one (possibly many) of the attributes have weights greater than 1, which means that they are not only attempting to lessen the effects of the independence assumption—otherwise they would restrict the weights to be no more than one.

$$w_i = \frac{\mathrm{GR}(i)}{\frac{1}{a}\sum_{i=1}^{a}\mathrm{GR}(i)}. \tag{7}$$

The gain ratio of an attribute is then simply the Mutual Information between that attribute and the class label divided by the entropy of that attribute:

$$\mathrm{GR}(i) = \frac{I(X_i, Y)}{H(X_i)} = \frac{\sum_y \sum_{x_1} \mathrm{P}(x_1, y) \log \frac{\mathrm{P}(x_1, y)}{\mathrm{P}(x_1)\mathrm{P}(y)}}{\sum_{x_1} \mathrm{P}(x_1) \log \frac{1}{\mathrm{P}(x_1)}}.$$

Several other wrapper-based methods are also proposed in Zhang and Sheng (2004). For example, they use a simple hill climbing search to optimize weight $\mathbf{w}$ using Area Under Curve (AUC) as an evaluation metric. Another Markov-Chain-Monte-Carlo (MCMC) method is also proposed.

An attribute weighting scheme based on differential evolution algorithms for naive Bayes classification have been proposed in Wu and Cai (2011). First, a population of attribute weight vectors

is randomly generated, weights in the population are constrained to be between 0 and 1. Second, typical genetic algorithmic steps of mutation and cross-over are performed over the the population. They defined a fitness function which is used to determine if mutation can replace the current individual (weight vector) with a new one. Their algorithm employs a greedy search strategy, where mutated individuals are selected as offspring only if the fitness is better than that of target individual. Otherwise, the target is maintained in the next iteration.

A scheme used in Hall (2007) is similar in spirit to SBC where the weight assigned to each attribute is inversely proportional to the minimum depth at which they were first tested in an unpruned decision tree. Weights are stabilized by averaging across 10 decision trees learned on data samples generated by bootstrapping 50% from the training data. Attributes not appearing in the decision trees are assigned a weight of zero. For example, one can assign weight to an attribute $i$ as:

$$w_i = \frac{1}{T}\sum_{t}^{T}\frac{1}{\sqrt{d_{ti}}}. \tag{8}$$

where $d_{ti}$ is the minimum depth at which the attribute $i$ appears in decision tree $t$, and $T$ is the total number of decision trees generated. To understand whether the improvement in naive Bayes accuracy was due to attribute weighting or selection, a variant of the above approach was also proposed where all non-zero weights are set to one. This is equivalent to SBC except using a bootstrap size of 50% with 10 iterations.

Both SB and CFS are feature selection methods. Since selecting an optimal number of features is not trivial, Hall (2007) proposed to use SB and CFS for feature weighting in naive Bayes. For example, the weight of an attribute $i$ can be defined as:

$$w_i = \frac{1}{\sqrt{r_i}}. \tag{9}$$

where $r_i$ is the rank of the feature based on SB and CFS feature selection.

The feature weighting method proposed in Ferreira et al. (2001) is the only one to use Equation 4, weighting each attribute value rather than each attribute. They used entropy-based discretization for numeric attributes and assigned a weight to each partition (value) of the attribute that is proportional to its predictive capability of the class. Different weight functions are proposed to assign weights to the values. These functions measure the difference between the distribution over classes for the particular attribute-value pair and a "baseline class distribution". The choice of weight function reduces to a choice of baseline distribution and the choice of measure quantifying the difference between the distributions. They used two simple baseline distribution schemes. The first assumes equiprobable classes, that is, uniform class priors. In that case the weight of for value $j$ of the attribute $i$ can be written as:

$$w_{ij} \quad \propto \quad \left(\sum_{y} |\,\mathrm{P}(y|X_i=j) - \frac{1}{|\mathcal{Y}|}\,|^{\alpha}\right)^{1/\alpha}.$$

where $\mathrm{P}(y|X_i=j)$ denotes the probability that the class is $y$ given that the $i$-th attribute of a data point has value $j$. Alternatively, the baseline class distribution can be set to the class probabilities across all values of the attribute (i.e., the class priors). The weighing function will take the form:

$$w_{ij} \quad \propto \quad \left(\sum_{y} |\,\mathrm{P}(y|X_i=j) - \mathrm{P}(y|X_i\neq \texttt{miss})\,|^{\alpha}\right)^{1/\alpha}.$$

where $P(y|X_i \neq \texttt{miss})$ is the class prior probability across all data points for which the attribute $i$ is not missing. Equation 10 and 10 assume an $L_\alpha$ distance metric where $\alpha = 2$ corresponds to the $L_2$ norm. Similarly, they have also proposed to use distance based on Kullback-Leibler divergence between the two distributions to set weights.

Many researchers have investigated techniques for extending the basic naive Bayes independence model with a small number of additional dependencies between attributes in order to improve classification performance (Zheng and Webb, 2000). Popular examples of such *semi-naive Bayes methods* include Tree-Augmented Naive Bayes (TAN) (Friedman et al., 1997) and ensemble methods such as Averaged n-Dependence Estimators (AnDE) (Webb et al., 2011). While detailed discussion of these methods is beyond the scope of this work, we will describe both TAN and AnDE in Section 5.10 for the purposes of empirical comparison.

Semi-naive Bayes methods usually limit the structure of the dependency network to simple structures such as trees, but more general graph structures can also be learnt. Considerable research has been done in the area of learning general Bayesian Networks (Greiner et al., 2004; Grossman and Domingos, 2004; Roos et al., 2005), with techniques differing on whether the network structure is chosen to optimize a generative or discriminative objective function, and whether the same objective is also used for optimizing the parameters of the model. Indeed optimizing network structure using a discriminative objective function can quickly become computationally challenging and thus recent work in this area has looked at efficient heuristics for discriminative structure learning (Pernkopf and Bilmes, 2010) and at developing decomposable discriminative objective functions (Carvalho et al., 2011).

In this paper we are interested in improving performance of the NB classifier by reducing the effect of attribute independence violations through attribute weighting. We do not attempt to identify the particular dependencies between attributes that cause the violations and thus are not attempting to address the much harder problem of inducing the dependency network structure. While it is conceivable that semi-naive Bayes methods and more general Bayesian Network classifier learning could also benefit from attribute weighting, we leave its investigation to future work.

A summary of different methods compared in this research is given in Table 2.

## 4. Weighting to Alleviate the Naive Bayes Independence Assumption

In this section, we will discuss our proposed methods to incorporate weights in naive Bayes.

### 4.1 WANBIA

Many previous approaches to attribute weighting for naive Bayes have found weights using some form of mechanism that increases the weights of attributes that are highly predictive of the class and decreases the weights of attributes that are less predictive of the class. We argue that this is not appropriate. Naive Bayes delivers Bayes optimal classification if the attribute independence assumption holds. Weighting should only be applied to remedy violations of the attribute independence assumption. For example, consider the case where there are three attributes, $x_1$, $x_2$ and $x_3$, such that $x_1$ and $x_2$ are conditionally independent of one another given the class and $x_3$ is an exact copy of $x_1$ (and hence violates the independence assumption). Irrespective of any measure of how well these three attributes each predict the class, Bayes optimal classification will be obtained by setting the weights of $x_1$ and $x_3$ to sum to 1.0 and setting the weight of $x_2$ to 1.0. In contrast, a method that uses a measure such as mutual information with the class to weight the attribute will

| Name | Description |
|---|---|
| | **Naive Bayes.** |
| NB | Naive Bayes Classifier. |
| | **Weighted Naive Bayes (using Typical Feature Weighting Methods).** |
| GRW | Use gain ratio as attribute weights in naive Bayes, shown in Equation 7 (Zhang and Sheng, 2004). |
| SBC | Assign weight to attribute $i$ as given in Equation 8 where $L = 5$ with a bootstrap size of 10%. Also $d_i = 0$ if $d_i > 3$ (Ratanamahatana and Gunopulos, 2003). |
| MH | Assign weight to attribute $i$ as given in Equation 8 where $L = 10$ with a bootstrap size of 50% (Hall, 2007). |
| SB | Use Selective Bayes method to determine the rank of individual features and assign weights according to Equation 9 (Langley and Sage, 1994). |
| CFS | Use correlation-based feature selection method to determine the rank of individual features and assign weights according to Equation 9 (Langley and Sage, 1994; Hall, 2007). |
| | **Selective Naive Bayes (using Typical Feature Selection Methods).** |
| SBC-FS | Similar to SBC except $w_i = 1$ if $w_i > 0$. |
| MH-FS | Similar to MH except $w_i = 1$ if $w_i > 0$ (Hall, 2007). |
| | **Weighted Naive Bayes (Ferreira et al., 2001).** |
| FNB-d1 | Weights computed per attribute value using Equation 10 with $\alpha = 2$. |
| FNB-d2 | Weights computed per attribute value using Equation 10 with $\alpha = 2$. |
| | **Semi-naive Bayes Classifiers.** |
| AnDE | Average $n$-Dependent Estimator (Webb et al., 2011). |
| TAN | Tree Augmented Naive Bayes (Friedman et al., 1997). |
| | **State of the Art Classification Techniques.** |
| RF | Random Forests (Breiman, 2001). |
| LR | Logistic Regression (Roos et al., 2005). |
| | **Weighted Naive Bayes (Proposed Methods, will be discussed in Section 4).** |
| $\text{WANBIA}^{\text{CLL}}$ | Naive Bayes weights obtained by maximizing Conditional Log-Likelihood. |
| $\text{WANBIA}^{\text{MSE}}$ | Naive Bayes weights obtained by minimizing Mean-Square-Error. |

Table 2: Summary of techniques compared in this research.

reduce the accuracy of the classifier relative to using uniform weights in any situation where $x_1$ and $x_3$ receive higher weights than $x_2$.

Rather than selecting weights based on measures of predictiveness, we suggest it is more profitable to pursue approaches such as those of Zhang and Sheng (2004) and Wu and Cai (2011) that optimize the weights to improve the prediction performance of the weighted classifier as a whole.

Following from Equations 1, 2 and 5, let us re-define the weighted naive Bayes model as:

$$\hat{\mathrm{P}}(y|\mathbf{x};\pi,\Theta,\mathbf{w}) = \frac{\pi_y \prod_i \theta_{X_i=x_i|y}^{w_i}}{\sum_{y'} \pi_{y'} \prod_i \theta_{X_i=x_i|y'}^{w_i}}, \tag{10}$$

with constraints:

$$\sum_y \pi_y = 1 \;\text{ and }\; \forall_{y,i} \sum_j \theta_{X_i=x_i|y} = 1,$$

where

- $\{\pi_y, \theta_{X_i=x_i|y}\}$ are naive Bayes parameters.
- $\pi \in [0,1]^{|\mathcal{Y}|}$ is a class probability vector.
- The matrix $\Theta$ consist of class and attribute-dependent probability vectors $\theta_{i,y} \in [0,1]^{|\mathcal{X}_i|}$.
- $\mathbf{w}$ is a vector of class-independent weights, $w_i$ for each attribute $i$.

Our proposed method WANBIA is inspired by Cerquides and De Mántaras (2005b) where weights of different classifiers in an ensemble are calculated by maximizing the conditional log-likelihood (CLL) of the data. We will follow their approach of maximizing the CLL of the data to determine weights $\mathbf{w}$ in the model. In doing so, we will make the following assumptions:

- Naive Bayes parameters $(\pi_y, \theta_{X_i=x_i|y})$ are fixed. Hence we can write $\hat{\mathrm{P}}(y|\mathbf{x};\pi,\Theta,\mathbf{w})$ in Equation 10 as $\hat{\mathrm{P}}(y|\mathbf{x};\mathbf{w})$.
- Weights lie in the interval between zero and one and hence $\mathbf{w} \in [0,1]^a$.

For notational simplicity, we will write conditional probabilities as $\theta_{x_i|y}$ instead of $\theta_{X_i=x_i|y}$. Since our prior is constant, let us define our supervised posterior as follows:

$$\hat{\mathrm{P}}(\mathcal{L} \,|\, \mathcal{D}, \mathbf{w}) \quad = \quad \prod_{j=1}^{|\mathcal{D}|} \hat{\mathrm{P}}(y^{(j)}|\mathbf{x}^{(j)};\mathbf{w}). \tag{11}$$

Taking the log of Equation 11, we get the Conditional Log-Likelihood (CLL) function, so our objective function can be defined as:

$$\begin{aligned} \mathrm{CLL}(\mathbf{w}) &= \log \hat{\mathrm{P}}(\mathcal{L} \,|\, \mathcal{D}, \mathbf{w}) \\ &= \sum_{j=1}^{|\mathcal{D}|} \log \hat{\mathrm{P}}(y^{(j)}|\mathbf{x}^{(j)};\mathbf{w}) \\ &= \sum_{j=1}^{|\mathcal{D}|} \log \frac{\gamma_{y\mathbf{x}}(\mathbf{w})}{\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})}, \end{aligned} \tag{12}$$

where

$$\gamma_{y\mathbf{x}}(\mathbf{w}) = \pi_y \prod_i \theta_{x_i|y}^{w_i}.$$

The proposed method WANBIA$^{\mathrm{CLL}}$ is aimed at solving the following optimization problem: find the weights $\mathbf{w}$ that maximizes the objective function $\mathrm{CLL}(\mathbf{w})$ in Equation 12 subject to $0 \leq w_i \leq 1 \ \forall i$. We can solve the problem by using the L-BFGS-M optimization procedure (Zhu et al., 1997). In order to do that, we need to be able to assess $\mathrm{CLL}(\mathbf{w})$ in Equation 12 and its gradient.

Before calculating the gradient of $\mathrm{CLL}(\mathbf{w})$ with respect to $\mathbf{w}$, let us find out the gradient of $\gamma_{y\mathbf{x}}(\mathbf{w})$ with respect to $w_i$, we can write:

$$\begin{aligned} \frac{\partial}{\partial w_i} \gamma_{y\mathbf{x}}(\mathbf{w}) &= (\pi_y \prod_{i' \neq i} \theta_{x_{i'}|y}^{w_{i'}}) \frac{\partial}{\partial w_i} \theta_{x_i|y}^{w_i} \\ &= (\pi_y \prod_{i' \neq i} \theta_{x_{i'}|y}^{w_{i'}}) \, \theta_{x_i|y}^{w_i} \log(\theta_{x_i|y}) \\ &= \gamma_{y\mathbf{x}}(\mathbf{w}) \log(\theta_{x_i|y}). \end{aligned} \tag{13}$$

Now, we can write the gradient of $\mathrm{CLL}(\mathbf{w})$ as:

$$\frac{\partial}{\partial w_i} \mathrm{CLL}(\mathbf{w}) \quad = \quad \frac{\partial}{\partial w_i} \sum_{\mathbf{x} \in \mathcal{D}} \left( \log(\gamma_{y\mathbf{x}}(\mathbf{w})) - \log(\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})) \right)$$

$$= \sum_{\mathbf{x}\in\mathcal{D}} \left( \frac{\gamma_{y\mathbf{x}}(\mathbf{w}) \log(\theta_{x_i|y})}{\gamma_{y\mathbf{x}}(\mathbf{w})} - \frac{\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w}) \log(\theta_{x_i|y'})}{\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})} \right)$$

$$= \sum_{\mathbf{x}\in\mathcal{D}} \left( \log(\theta_{x_i|y}) - \sum_{y'} \hat{\mathrm{P}}(y'|\mathbf{x};\mathbf{w}) \log(\theta_{x_i|y'}) \right). \tag{14}$$

WANBIA$^{\mathrm{CLL}}$ evaluates the function in Equation 12 and its gradient in Equation 14 to determine optimal values of weight vector $\mathbf{w}$.

Instead of maximizing the supervised posterior, one can also minimize Mean Square Error (MSE). Our second proposed weighting scheme WANBIA$^{\mathrm{MSE}}$ is based on minimizing the MSE function. Based on MSE, we can define our objective function as follows:

$$\mathrm{MSE}(\mathbf{w}) = \frac{1}{2} \sum_{\mathbf{x}^{(j)}\in\mathcal{D}} \sum_{y} (\mathrm{P}(y|\mathbf{x}^{(j)}) - \hat{\mathrm{P}}(y|\mathbf{x}^{(j)}))^2, \tag{15}$$

where we define

$$\mathrm{P}(y|\mathbf{x}^{(j)}) = \begin{cases} 1 & \text{if } y = y^{(j)} \\ 0 & \text{otherwise} \end{cases}.$$

The gradient of $\mathrm{MSE}(\mathbf{w})$ in Equation 15 with respect to $\mathbf{w}$ can be derived as:

$$\frac{\partial \mathrm{MSE}(\mathbf{w})}{\partial w_i} = -\sum_{\mathbf{x}\in\mathcal{D}} \sum_{y} \left( \mathrm{P}(y|\mathbf{x}) - \hat{\mathrm{P}}(y|\mathbf{x}) \right) \frac{\partial \hat{\mathrm{P}}(y|\mathbf{x})}{\partial w_i}, \tag{16}$$

where

$$\frac{\partial \hat{\mathrm{P}}(y|\mathbf{x})}{\partial w_i} = \frac{\frac{\partial}{\partial w_i} \gamma_{y\mathbf{x}}(\mathbf{w})}{\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})} - \frac{\gamma_{y\mathbf{x}}(\mathbf{w}) \frac{\partial}{\partial w_i} \sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})}{(\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w}))^2}$$

$$= \frac{1}{\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})} \left( \frac{\partial \gamma_{y\mathbf{x}}(\mathbf{w})}{\partial w_i} - \hat{\mathrm{P}}(y|\mathbf{x}) \sum_{y'} \frac{\partial \gamma_{y'\mathbf{x}}(\mathbf{w})}{\partial w_i} \right).$$

Following from Equation 13, we can write:

$$\frac{\partial \hat{\mathrm{P}}(y|\mathbf{x})}{\partial w_i} = \frac{1}{\sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w})} \left( \gamma_{y\mathbf{x}}(\mathbf{w}) \log(\theta_{x_i|y}) - \hat{\mathrm{P}}(y|\mathbf{x}) \sum_{y'} \gamma_{y'\mathbf{x}}(\mathbf{w}) \log(\theta_{x_i|y'}) \right)$$

$$= \hat{\mathrm{P}}(y|\mathbf{x}) \log(\theta_{x_i|y}) - \hat{\mathrm{P}}(y|\mathbf{x}) \sum_{y'} \hat{\mathrm{P}}(y'|\mathbf{x}) \log(\theta_{x_i|y'})$$

$$= \hat{\mathrm{P}}(y|\mathbf{x}) \left( \log(\theta_{x_i|y}) - \sum_{y'} \hat{\mathrm{P}}(y'|\mathbf{x}) \log(\theta_{x_i|y'}) \right). \tag{17}$$

Plugging the value of $\frac{\partial \hat{\mathrm{P}}(y|\mathbf{x})}{\partial w_i}$ from Equation 17 in Equation 16, we can write the gradient as:

$$\frac{\partial \mathrm{MSE}(\mathbf{w})}{\partial w_i} = -\sum_{\mathbf{x}\in\mathcal{D}} \sum_{y} \left( \mathrm{P}(y|\mathbf{x}) - \hat{\mathrm{P}}(y|\mathbf{x}) \right) \hat{\mathrm{P}}(y|\mathbf{x})$$

$$\left( \log(\theta_{x_i|y}) - \sum_{y'} \hat{\mathrm{P}}(y'|\mathbf{x}) \log(\theta_{x_i|y'}) \right). \tag{18}$$

WANBIA$^{\text{MSE}}$ evaluates the function in Equation 15 and its gradient in Equation 18 to determine the optimal value of weight vector $\mathbf{w}$.

### 4.2 Connections with Logistic Regression

In this section, we will re-visit naive Bayes to illustrate WANBIA's connection with the logistic regression.

#### 4.2.1 BACKGROUND: NAIVE BAYES AND LOGISTIC REGRESSION

As discussed in Section 2 and 4.1, the naive Bayes (NB) model for estimating $\mathrm{P}(y|\mathbf{x})$ is parameterized by a class probability vector $\pi \in [0,1]^{|\mathcal{Y}|}$ and a matrix $\Theta$, consisting of class and attribute dependent probability vectors $\theta_{i,y} \in [0,1]^{|\mathcal{X}_i|}$. The NB model thus contains

$$(|\mathcal{Y}|-1)+|\mathcal{Y}|\sum_i(|\mathcal{X}_i|-1)$$

free parameters, which are estimated by maximizing the likelihood function:

$$\mathrm{P}(\mathcal{D},\mathcal{L};\pi,\Theta)=\prod_j \mathrm{P}(y^{(j)},\mathbf{x}^{(j)}),$$

or the posterior over model parameters (in which case they are referred to as maximum a posteriori or MAP estimates). Importantly, these estimates can be calculated analytically from attribute-value count vectors.

Meanwhile a multi-class logistic regression model is parameterized by a vector $\alpha \in \mathcal{R}^{|\mathcal{Y}|}$ and matrix $\mathcal{B} \in \mathcal{R}^{|\mathcal{Y}|\times a}$ each consisting of real values, and can be written as:

$$\mathrm{P}_{\mathrm{LR}}(y|\mathbf{x};\alpha,\mathcal{B})=\frac{\exp(\alpha_y+\sum_i \beta_{i,y}x_i)}{\sum_{y'}\exp(\alpha_{y'}+\sum_i \beta_{i,y'}x_i)},$$

where

$$\alpha_1=0 \;\;\&\;\; \forall_i\, \beta_{i,1}=0.$$

The constraints arbitrarily setting all parameters for $y=1$ to the value zero are necessary only to prevent over-parameterization. The LR model, therefore, has:

$$(|\mathcal{Y}|-1)\times(1+a)$$

free parameters. Rather than maximizing the likelihood, LR parameters are estimated by maximizing the conditional likelihood of the class labels given the data:

$$\mathrm{P}(\mathcal{L}|\mathcal{D};\alpha,\mathcal{B})=\prod_j \mathrm{P}(y^{(j)}|\mathbf{x}^{(j)}),$$

or the corresponding posterior distribution. Estimating the parameters in this fashion requires search using gradient-based methods.

Mathematically the relationship between the two models is simple. One can compare the models, by considering that the "multiplicative contribution" of an attribute value $x_i$ in NB is found

by simply looking up the corresponding parameter $\theta_{X_i=x_i|y}$ in $\Theta$, while for LR it is calculated as $\exp(\beta_{i,y}x_i)$, that is, by taking the exponent of the product of the value with an attribute (but *not value*) dependent parameter $\beta_{i,y}$ from $\mathcal{B}$.[2]

#### 4.2.2 PARAMETERS OF WEIGHTED ATTRIBUTE NAIVE BAYES

The WANBIA model is an extension of the NB model where we introduce a weight vector $\mathbf{w} \in [0,1]^a$ containing a class-independent weight $w_i$ for each attribute $i$. The model as written in Equation 10 includes the NB model as a special case (where $\mathbf{w} = \mathbf{1}$). We do not treat the NB parameters of the model as free however, but instead fix them to their MAP estimates (assuming the weights were all one), which can be computed analytically and therefore does not require any search. We then estimate the parameter vector $w$ by maximizing the Conditional Log Likelihood (CLL) or by minimizing the Mean Squared Error (MSE).[3]

Thus in terms of the number of parameters that needs to be estimated using gradient-based search, a WANBIA model can be considered to have $a$ free parameters, which is always less than the corresponding LR model with $(|\mathcal{Y}|-1)(1+a)$ free parameters to be estimated. Thus for a binary class problems containing only binary attributes, WANBIA has 1 less free parameter than LR, but for multi-class problems with binary attributes it results in a *multiplicative factor* of $|\mathcal{Y}|-1$ fewer parameters. Since parameter estimation using CLL or MSE (or even Hinge Loss) requires search, fewer free parameters to estimate means faster learning and therefore scaling to larger problems.

For problems containing non-binary attributes, WANBIA allows us to build (more expressive) *non-linear classifiers*, which are not possible for Logistic Regression unless one "binarizes" all attributes, with the resulting blow-out in the number of free parameters as mentioned above. One should note that LR can only operate on nominal data by binarizing it. Therefore, on discrete problems with nominal data, WANBIA offers significant advantage in terms of the number of free parameters.

Lest the reader assume that the only goal of this work is to find a more computationally efficient version of LR, we note that the real advantage of the WANBIA model is to make use of the information present in the easy to compute naive Bayes MAP estimates to guide the search toward reasonable settings for parameters of a model that is *not hampered by the assumption of attribute independence*.

A summary of the comparison of naive Bayes, WANBIA and Logistic Regression is given in Table 3.

## 5. Experiments

In this section, we compare the performance of our proposed methods WANBIA$^{\text{CLL}}$ and WANBIA$^{\text{MSE}}$ with state of the art classifiers, existing semi-naive Bayes methods and weighted naive Bayes methods based on both attribute selection and attribute weighting. The performance is analyzed in terms of 0-1 loss, root mean square error (RMSE), bias and variance on 73 natural domains from the UCI repository of machine learning (Frank and Asuncion, 2010). Table 4 describes the details of each data set used, including the number of instances, attributes and classes.

2. Note that unlike the NB model, the LR model does not require that the domain of attribute values be discrete. Non-discrete data can also be handled by Naive Bayes models, but a different parameterization for the distribution over attribute values must be used.
3. Note that we cannot maximize the Log Likelihood (LL) since the model is not generative.

| | **Naive Bayes** | **WANBIA** | **Logistic Regression** |
|---|---|---|---|
| Approach | Estimate parameters by maximizing the likelihood function $P(\mathcal{D},\mathcal{L})$ | Estimate parameters by maximizing conditional log-likelihood $P(\mathcal{L}\|\mathcal{D})$ or minimizing Mean-Squared-Error | Estimate parameters by maximizing conditional likelihood $P(\mathcal{L}\|\mathcal{D})$ |
| Form | $\hat{P}(y\|\mathbf{x};\pi,\Theta)$ | $\hat{P}(y\|\mathbf{x};\pi,\Theta,\mathbf{w})$ | $\hat{P}(y\|\mathbf{x};\alpha,\mathcal{B})$ |
| Formula | $\frac{\pi_y\prod_i\theta_{x_i\|i,y}}{\sum_{y'}\pi_{y'}\prod_i\theta_{x_i\|i,y'}}$ | $\frac{\pi_y\prod_i\theta_{x_i\|i,y}^{w_i}}{\sum_{y'}\pi_{y'}\prod_i\theta_{x_i\|i,y'}^{w_i}}$ | $\frac{\exp(\alpha_y+\sum_i\beta_{i,y}x_i)}{\sum_{y'}\exp(\alpha_{y'}+\sum_i\beta_{i,y'}x_i)}$ |
| Constraints | $\pi\in[0,1]^{\|\mathcal{Y}\|}$, $\theta_{i,y}\in[0,1]^{\|\mathcal{X}_i\|}$, $\sum_y\pi_y=1$, $\forall_{y,i}\sum_j\theta_{X_i=x_i\|y}=1$ | $\pi\in[0,1]^{\|\mathcal{Y}\|}$, $\theta_{i,y}\in[0,1]^{\|\mathcal{X}_i\|}$, $\mathbf{w}\in[0,1]^a$, $\sum_y\pi_y=1$, $\forall_{y,i}\sum_j\theta_{X_i=x_i\|y}=1$ | $\alpha_1=0,\forall_i\,\beta_{i,1}=0$ |
| No. of 'Fixed' Parameters | $(\|\mathcal{Y}\|-1)+\|\mathcal{Y}\|\sum_i(\|\mathcal{X}_i\|-1)$ | $(\|\mathcal{Y}\|-1)+\|\mathcal{Y}\|\sum_i(\|\mathcal{X}_i\|-1)$ | None |
| No. of 'Fixed' Parameters (Binary Case) | $1+(2\times a)$ | $1+(2\times a)$ | None |
| Strategy to calculate Fixed Parameters | $\pi$ and $\Theta$ are fixed to their MAP estimates | $\pi$ and $\Theta$ are fixed to their MAP estimates when $\mathbf{w}=\mathbf{1}$ | Not applicable |
| No. of 'Free' Parameters | None | $a$ | $(\|\mathcal{Y}\|-1)\times(1+a)$ |
| No. of 'Free' Parameters (Binary case) | None | $a$ | $(1+a)$ |

Table 3: Comparison of naive Bayes, weighted naive Bayes and Logistic Regression

This section is organized as follows: we will discuss our experimental methodology with details on statistics employed and miscellaneous issues in Section 5.1. Section 5.2 illustrates the impact of a single weight on bias, variance, 0-1 loss and RMSE of naive Bayes as shown in Equation 6. The performance of our two proposed weighting methods WANBIA$^{\text{CLL}}$ and WANBIA$^{\text{MSE}}$ is compared in Section 5.3. We will discuss the calibration performance of WANBIA in Section 5.4. In Section 5.5, we will discuss results when the proposed methods are constrained to learn only a single weight. In Section 5.6 and 5.7, WANBIA$^{\text{CLL}}$ and WANBIA$^{\text{MSE}}$ are compared with naive Bayes where weights are induced through various feature weighting and feature selection schemes respectively. We compare the performance of our proposed methods with per-attribute value weight learning method of Ferreira et al. (2001) in Section 5.8. We will discuss the significance of these results in Section 5.9. The performance of our proposed methods is compared with state of the art classifiers like Average $n$-Dependent Estimators (AnDE), Tree Augmented Networks (TAN), Random Forests (RF) and Logistic Regression (LR) in Section 5.10, 5.11 and 5.12 respectively. Results are summarized in Section 5.13.

### 5.1 Experimental Methodology

The experiments are conducted in the Weka work-branch (version 3.5.7) on data sets described in Table 4. Each algorithm is tested on each data set using 20 rounds of 2-fold cross validation. We employed Friedman and Nemenyi tests with a significance level of 0.05 to evaluate the performance

| Domain | Case | Att | Class | Domain | Case | Att | Class |
|---|---|---|---|---|---|---|---|
| Abalone | 4177 | 9 | 3 | MAGIC Gamma Telescope | 19020 | 11 | 2 |
| Adult | 48842 | 15 | 2 | Mushrooms | 8124 | 23 | 2 |
| Annealing | 898 | 39 | 6 | Musk1 | 476 | 167 | 2 |
| Audiology | 226 | 70 | 24 | Musk2 | 6598 | 167 | 2 |
| Auto Imports | 205 | 26 | 7 | Nettalk(Phoneme) | 5438 | 8 | 52 |
| Balance Scale | 625 | 5 | 3 | New-Thyroid | 215 | 6 | 3 |
| Breast Cancer (Wisconsin) | 699 | 10 | 2 | Nursery | 12960 | 9 | 5 |
| Car Evaluation | 1728 | 8 | 4 | Optical Digits | 5620 | 49 | 10 |
| Census-Income (KDD) | 299285 | 40 | 2 | Page BlocksClassification | 5473 | 11 | 5 |
| Chess | 551 | 40 | 2 | Pen Digits | 10992 | 17 | 10 |
| Connect-4 Opening | 67557 | 43 | 3 | Pima Indians Diabetes | 768 | 9 | 2 |
| Contact-lenses | 24 | 5 | 3 | Pioneer | 9150 | 37 | 57 |
| Contraceptive Method Choice | 1473 | 10 | 3 | Poker-hand | 1175067 | 11 | 10 |
| Covertype | 581012 | 55 | 7 | Postoperative Patient | 90 | 9 | 3 |
| Credit Screening | 690 | 16 | 2 | Primary Tumor | 339 | 18 | 22 |
| Cylinder | 540 | 40 | 2 | Promoter Gene Sequences | 106 | 58 | 2 |
| Dermatology | 366 | 35 | 6 | Satellite | 6435 | 37 | 6 |
| Echocardiogram | 131 | 7 | 2 | Segment | 2310 | 20 | 7 |
| German | 1000 | 21 | 2 | Sick-euthyroid | 3772 | 30 | 2 |
| Glass Identification | 214 | 10 | 3 | Sign | 12546 | 9 | 3 |
| Haberman's Survival | 306 | 4 | 2 | Sonar Classification | 208 | 61 | 2 |
| Heart Disease (Cleveland) | 303 | 14 | 2 | Spambase | 4601 | 58 | 2 |
| Hepatitis | 155 | 20 | 2 | Splice-junction Gene Sequences | 3190 | 62 | 3 |
| Horse Colic | 368 | 22 | 2 | Statlog (Shuttle) | 58000 | 10 | 7 |
| House Votes 84 | 435 | 17 | 2 | Syncon | 600 | 61 | 6 |
| Hungarian | 294 | 14 | 2 | Teaching Assistant Evaluation | 151 | 6 | 3 |
| Hypothyroid(Garavan) | 3772 | 30 | 4 | Thyroid | 9169 | 30 | 20 |
| Ionosphere | 351 | 35 | 2 | Tic-Tac-Toe Endgame | 958 | 10 | 2 |
| Iris Classification | 150 | 5 | 3 | Vehicle | 846 | 19 | 4 |
| King-rook-vs-king-pawn | 3196 | 37 | 2 | Volcanoes | 1520 | 4 | 4 |
| Labor Negotiations | 57 | 17 | 2 | Vowel | 990 | 14 | 11 |
| LED | 1000 | 8 | 10 | Wall-following | 5456 | 25 | 4 |
| Letter Recognition | 20000 | 17 | 26 | Waveform-5000 | 5000 | 41 | 3 |
| Liver Disorders (Bupa) | 345 | 7 | 2 | Wine Recognition | 178 | 14 | 3 |
| Localization | 164860 | 7 | 3 | Yeast | 1484 | 9 | 10 |
| Lung Cancer | 32 | 57 | 3 | Zoo | 101 | 17 | 7 |
| Lymphography | 148 | 19 | 4 | | | | |

Table 4: Data sets

of each algorithm. The experiments were conducted on a Linux machine with 2.8 GHz processor and 16 GB of RAM.

#### 5.1.1 TWO-FOLD CROSS-VALIDATION BIAS-VARIANCE ESTIMATION

The Bias-variance decomposition provides valuable insights into the components of the error of learned classifiers. *Bias* denotes the systematic component of error, which describes how closely the learner is able to describe the decision surfaces for a domain. *Variance* describes the component of error that stems from sampling, which reflects the sensitivity of the learner to variations in the training sample (Kohavi and Wolpert, 1996; Webb, 2000). There are a number of different bias-variance decomposition definitions. In this research, we use the bias and variance definitions of Kohavi and Wolpert (1996) together with the repeated cross-validation bias-variance estimation method proposed by Webb (2000). Kohavi and Wolpert define bias and variance as follows:

$$\text{bias}^2 = \frac{1}{2}\sum_{y\in\mathcal{Y}}\left(\text{P}(y|\mathbf{x}) - \hat{\text{P}}(y\,|\,\mathbf{x})\right)^2,$$

and

$$\text{variance} = \frac{1}{2}\left(1 - \sum_{y \in \mathcal{Y}} \hat{\mathrm{P}}(y \,|\, \mathbf{x})^2\right).$$

In the method of Kohavi and Wolpert (1996), which is the default bias-variance estimation method in Weka, the randomized training data are randomly divided into a training pool and a test pool. Each pool contains 50% of the data. 50 (the default number in Weka) local training sets, each containing half of the training pool, are sampled from the training pool. Hence, each local training set is only 25% of the full data set. Classifiers are generated from local training sets and bias, variance and error are estimated from the performance of the classifiers on the test pool. However, in this work, the repeated cross-validation bias-variance estimation method is used as it results in the use of substantially larger training sets. Only two folds are used because, if more than two are used, the multiple classifiers are trained from training sets with large overlap, and hence the estimation of variance is compromised. A further benefit of this approach relative to the Kohavi Wolpert method is that every case in the training data is used the same number of times for both training and testing.

A reason for performing bias/variance estimation is that it provides insights into how the learning algorithm will perform with varying amount of data. We expect low variance algorithms to have relatively low error for small data and low bias algorithms to have relatively low error for large data (Damien and Webb, 2002).

#### 5.1.2 STATISTICS EMPLOYED

We employ the following statistics to interpret results:

- **Win/Draw/Loss (WDL) Record -** When two algorithms are compared, we count the number of data sets for which one algorithm performs better, equally well or worse than the other on a given measure. A standard binomial sign test, assuming that wins and losses are equiprobable, is applied to these records. We assess a difference as significant if the outcome of a one-tailed binomial sign test is less than 0.05.

- **Average -** The average (arithmetic mean) across all data sets provides a gross indication of relative performance in addition to other statistics. In some cases, we normalize the results with respect to one of our proposed method's results and plot the geometric mean of the ratios.

- **Significance (Friedman and Nemenyi) Test -** We employ the Friedman and the Nemenyi tests for comparison of multiple algorithms over multiple data sets (Demšar, 2006; Friedman, 1937, 1940). The Friedman test is a non-parametric equivalent of the repeated measures ANOVA (analysis of variance). We follow the steps below to compute results:

  - Calculate the rank of each algorithm for each data set separately (assign average ranks in case of a tie). Calculate the average rank of each algorithm.
  - Compute the Friedman statistics as derived in Kononenko (1990) for the set of average ranks:

$$F_F = \frac{(D-1)\chi_F^2}{D(g-1) - \chi_F^2}, \tag{19}$$

where

$$\chi_F^2 = \frac{12D}{g(g+1)}\left(\sum_i R_i^2 - \frac{g(g+1)^2}{4}\right),$$

$g$ is the number of algorithms being compared, $D$ is the number of data sets and $R_i$ is the average rank of the $i$-th algorithm.

- Specify the null hypothesis. In our case the null hypothesis is that there is no difference in the average ranks.
- Check if we can reject the null hypothesis. One can reject the null hypothesis if the Friedman statistic (Equation 19) is larger than the critical value of the $F$ distribution with $g-1$ and $(g-1)(D-1)$ degrees of freedom for $\alpha = 0.05$.
- If null hypothesis is rejected, perform Nemenyi tests which is used to further analyze which pairs of algorithms are significantly different. Let $d_{ij}$ be the difference between the average ranks of the $i^{th}$ algorithm and $j^{th}$ algorithm. We assess the difference between the algorithms to be significant if $d_{ij} > \textit{critical difference}$ (CD) $= q_{0.05}\sqrt{\frac{g(g+1)}{6D}}$, where $q_{0.05}$ are the critical values that are calculated by dividing the values in the row for the infinite degree of freedom of the table of Studentized range statistics ($\alpha = 0.05$) by $\sqrt{2}$.

#### 5.1.3 MISCELLANEOUS ISSUES

This section explains other issues related to the experiments.

- **Probability Estimates -** The base probabilities of each algorithm are estimated using $m$-estimation, since in our initial experiments it leads to more accurate probabilities than Laplace estimation for naive Bayes and weighted naive Bayes. In the experiments, we use $m = 1.0$, computing the conditional probability as:

$$\hat{\mathrm{P}}(x_i|y) = \frac{N_{x_i,y} + \frac{m}{|X_i|}}{(N_y - N_?) + m}, \tag{20}$$

where $N_{x_i,y}$ is the count of data points with attribute value $x_i$ and class label $y$, $N_y$ is the count of data points with class label $y$, $N_?$ is the number of missing values of attribute $i$.

- **Numeric Values -** To handle numeric attributes we tested the following techniques in our initial experiments:
  - Quantitative attributes are discretized using three bin discretization.
  - Quantitative attributes are discretized using Minimum Description Length (MDL) discretization (Fayyad and Keki, 1992).
  - Kernel Density Estimation (KDE) computing the probability of numeric attributes as:

$$\hat{\mathrm{P}}(x_i|y) = \frac{1}{n}\sum_{\mathbf{x}^{(j)} \in \mathcal{D}} \exp\left(\frac{||x_i^j - x_i||^2}{\lambda^2}\right).$$

| | NB vs. NB$^w$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| W/D/L | $w$=0.1 | $w$=0.2 | $w$=0.3 | $w$=0.4 | $w$=0.5 | $w$=0.6 | $w$=0.7 | $w$=0.8 | $w$=0.9 | $w$=1.0 |
| **0/1 Loss** | 61/2/10 | 58/1/14 | 53/1/19 | 51/3/19 | 46/3/24 | 39/5/29 | 36/8/29 | 28/11/34 | 23/12/38 | 0/73/0 |
| | <**0.001** | <**0.001** | <**0.001** | <**0.001** | 0.011 | 0.532 | 0.457 | 0.525 | 0.072 | 2 |
| | **NB vs. NB$^w$** | | | | | | | | | |
| W/D/L | $w$=0.1 | $w$=0.2 | $w$=0.3 | $w$=0.4 | $w$=0.5 | $w$=0.6 | $w$=0.7 | $w$=0.8 | $w$=0.9 | $w$=1.0 |
| **RMSE** | 53/1/19 | 44/1/28 | 37/1/35 | 26/1/46 | 21/1/51 | 18/1/54 | 17/1/55 | 13/1/59 | 12/2/59 | 0/73/0 |
| | <**0.001** | 0.076 | 0.906 | **0.024** | <**0.001** | <**0.001** | <**0.001** | <**0.001** | <**0.001** | 2 |
| | **NB vs. NB$^w$** | | | | | | | | | |
| W/D/L | $w$=0.1 | $w$=0.2 | $w$=0.3 | $w$=0.4 | $w$=0.5 | $w$=0.6 | $w$=0.7 | $w$=0.8 | $w$=0.9 | $w$=1.0 |
| **Bias** | 59/2/12 | 54/1/18 | 51/3/19 | 49/3/21 | 48/5/20 | 44/4/25 | 43/6/24 | 35/9/29 | 29/14/30 | 0/73/0 |
| | <**0.001** | <**0.001** | <**0.001** | <**0.001** | <**0.001** | **0.029** | **0.027** | 0.532 | 1 | 2 |
| | **NB vs. NB$^w$** | | | | | | | | | |
| W/D/L | $w$=0.1 | $w$=0.2 | $w$=0.3 | $w$=0.4 | $w$=0.5 | $w$=0.6 | $w$=0.7 | $w$=0.8 | $w$=0.9 | $w$=1.0 |
| **Variance** | 39/1/33 | 38/3/32 | 30/7/36 | 31/4/38 | 30/6/37 | 31/4/38 | 23/11/39 | 22/18/33 | 25/18/30 | 0/73/0 |
| | 0.556 | 0.550 | 0.538 | 0.470 | 0.463 | 0.470 | 0.055 | 0.177 | 0.590 | 2 |

Table 5: Win/Draw/Loss comparison of NB with weighted NB of form Equation 6

  - k-Nearest neighbor (k-NN) estimation to compute the probability of numeric attributes. The probabilities are computed using Equation 20, where $N_{x_i,y}$ and $N_y$ are calculated over a neighborhood spanning $k$ neighbors of $x_i$. We use $k = 10$, $k = 20$ and $k = 30$.

  While a detailed analysis of the results of this comparison is beyond the scope of this work, we summarize our findings as follows: the k-NN approach with $k = 50$ achieved the best 0-1 loss results in terms of Win/Draw/Loss. The k-NN with $k = 20$ resulted in the best bias performance, KDE in the best variance and MDL discretization in best RMSE performance. However, we found KDE and k-NN schemes to be extremely slow at classification time. We found that MDL discretization provides the best trade-off between the accuracy and computational efficiency. Therefore, we chose to discretize numeric attributes with MDL scheme.

- **Missing Values -** For the results reported in from Section 5.2 to Section 5.9, the missing values of any attributes are incorporated in probability computation as depicted in Equation 20. Starting from Section 5.10, missing values are treated as a distinct value. The motivation behind this is to have a fair comparison between WANBIA and other state of the art classifiers, for instance, Logistic Regression and Random Forest.

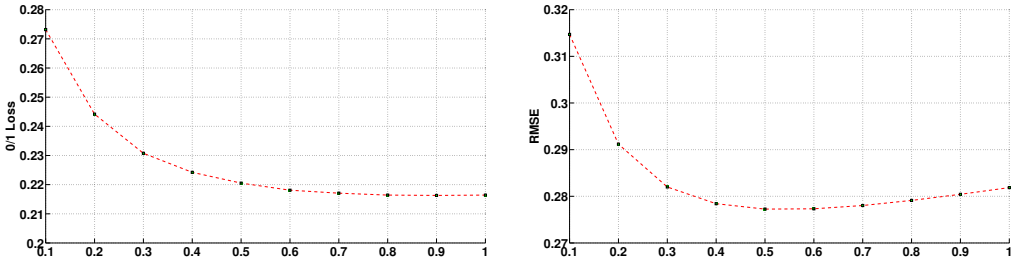
- **Notation -** We will categorize data sets in terms of their size. For example, data sets with instances $\leq 1000$, $> 1000$ and $\leq 10000$, $> 10000$ are denoted as bottom size, medium size and top size respectively. We will report results on these sets to discuss suitability of a classifier for data sets of different sizes.

### 5.2 Effects of a Single Weight on Naive Bayes

In this section, we will employ the Win/Draw/Loss record (WDL) and simple arithmetic mean to summarize the effects of weights on naive Bayes' classification performance. Table 5 compares the WDL of naive Bayes in Equation 3 with weighted naive Bayes in Equation 6 as the weight $w$ is varied from 0.1 to 1.0. The WDL are presented for 0-1 loss, RMSE, bias and variance. The results reveal that higher value of $w$, for example, $w = 0.9$ results in significantly better performance in

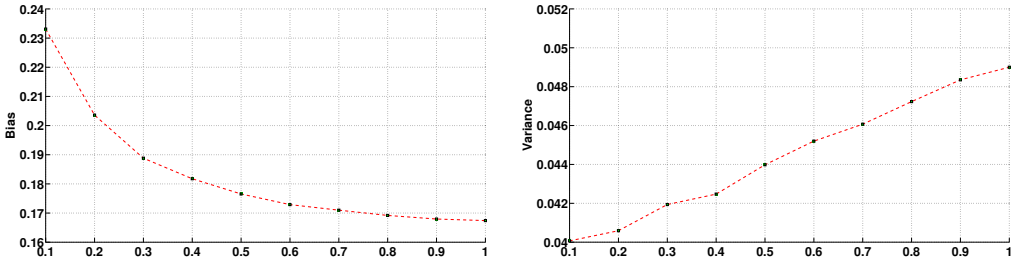
terms of 0-1 loss and RMSE and non-significantly better performance in terms of bias and variance as compared to the lower values.

Averaged (arithmetic mean) 0-1 loss, RMSE, bias and variance results across 73 data sets as a function of weight are plotted in Figure 2 and 3. As can be seen from the figures that as $w$




(a) (b)

Figure 2: Averaged 0-1 Loss (2(a)), RMSE (2(b)) across 73 data sets, as function of $w$.




(a) (b)

Figure 3: Averaged Bias (3(a)) Variance (3(b)) across 73 data sets, as function of $w$.

is increased from 0.1 to 1.0, bias decreases and variance increases. It is hard to characterize 0-1 loss and RMSE curves in Figure 2. 0-1 loss is decreased as we increase the value of $w$ and is almost constant when $w > 0.7$. However, RMSE drops as $w$ is increased to 0.5 and then increases for $w > 0.5$. As the results are averaged across 73 data sets, it is hard to say anything conclusive, however, we conjecture that the optimal value of 0.5 for $w$ in case of RMSE metric suggests that in most data sets, only half of the attributes are providing independent information.

### 5.3 Mean-Square-Error versus Conditional-Log-Likelihood Objective Function

The Win/Draw/Loss comparison of our two proposed methods WANBIA$^{\text{CLL}}$ and WANBIA$^{\text{MSE}}$ is given in Table 6. It can be seen that WANBIA$^{\text{MSE}}$ has significantly better bias but significantly worst variance than WANBIA$^{\text{CLL}}$. Also, WANBIA$^{\text{MSE}}$ wins on the majority of data sets in terms of 0-1 loss and RMSE but the results are not significant. The two methods are also compared against naive Bayes in Table 7. The two versions of WANBIA win significantly in terms of bias, 0-1 loss and RMSE against naive Bayes.

| | **WANBIA$^{\text{CLL}}$ vs. WANBIA$^{\text{MSE}}$** | |
|---|---|---|
| | W/D/L | *p* |
| Bias | 19/10/44 | **0.002** |
| Variance | 42/7/24 | **0.035** |
| 0-1 Loss | 26/8/39 | 0.136 |
| RMSE | 28/12/33 | 0.608 |

Table 6: Win/Draw/Loss: WANBIA$^{\text{CLL}}$ versus WANBIA$^{\text{MSE}}$

| | **WANBIA$^{\text{CLL}}$ vs. NB** | | **WANBIA$^{\text{MSE}}$ vs. NB** | |
|---|---|---|---|---|
| | W/D/L | *p* | W/D/L | *p* |
| Bias | 55/7/11 | <**0.001** | 57/7/9 | <**0.001** |
| Variance | 24/8/41 | **0.046** | 24/7/42 | **0.035** |
| 0-1 Loss | 46/8/19 | **0.001** | 49/7/17 | <**0.001** |
| RMSE | 55/8/10 | <**0.001** | 54/6/13 | <**0.001** |

Table 7: Win/Draw/Loss: WANBIA$^{\text{CLL}}$ versus NB, WANBIA$^{\text{MSE}}$ versus NB

Since the performance of WANBIA$^{\text{MSE}}$ and WANBIA$^{\text{CLL}}$ is similar, from the following section onwards, for simplicity we will only consider WANBIA$^{\text{MSE}}$ when comparing with other weighted NB and state of the art classification methods and denote it by WANBIA.

### 5.4 Comparing the Calibration of WANBIA and NB Probability Estimates

One benefit of Bayesian classifiers (and indeed also Logistic Regression) over Support Vector Machine and Decision-Tree based classifiers is that the former implicitly produce interpretable confidence values for each classification in the form of class membership probability estimates $\hat{\text{P}}(y|\mathbf{x})$. Unfortunately, the probability estimates that naive Bayes produces can often be poorly calibrated as a result of the conditional independence assumption. Whenever the conditional independence assumption is violated, which is usually the case in practice, the probability estimates tend to be more extreme (closer to zero or one) than they should otherwise be. In other words, the NB classifier tends to be more confident in its class membership predictions than is warranted given the training data. Poor calibration and over-confidence do not always affect performance in terms of 0-1 Loss, but in many applications accurate estimates of the probability of $\mathbf{x}$ belonging to class $y$ are needed (Zadrozny and Elkan, 2002).

Since, WANBIA is based on alleviating the attribute-independence assumption, it also corrects for naive Bayes' poor calibration as can be seen in Figure 4.

Formally, we say a classifier is well-calibrated (Murphy and Winkler, 1977), if the empirical class membership probability $\tilde{\text{P}}(y|\hat{\text{P}}(y|\mathbf{x}))$ conditioned on the predicted probability $\hat{\text{P}}(y|\mathbf{x})$ converges to the latter, as the number of training examples goes to infinity. Putting it more simply, if we count the number of data points for which a classifier assigned a particular class probability of say $\hat{\text{P}}(y|\mathbf{x}) = 0.3$, then if the classifier is well-calibrated we would expect approximately 30% of these data points to be members of class $y$ in the data set.

Figure 4 shows reliability diagrams showing the relative calibration for naive Bayes and WANBIA (DeGroot and Fienbert, 1982). Reliability diagrams plot empirical class membership probability $\tilde{\mathrm{P}}(y|\hat{\mathrm{P}}(y|\mathbf{x}))$ versus predicted class membership probability for $\hat{\mathrm{P}}(y|\mathbf{x})$ at various levels of the latter. If a classifier is well-calibrated, all points will lie on the diagonal indicating that estimates are equal to their empirical probability. In the diagrams, the empirical probability $\tilde{\mathrm{P}}(y|\hat{\mathrm{P}}(y|\mathbf{x}) = p)$ is the ratio of the number of training points with predicted probability $p$ belonging to class $y$ to the total number of training points with predicted probability $p$. Since, the number of different predicted values is large as compared to the number of data points, we can not calculate reliable empirical probabilities for each data point, but instead bin the predicted values along the x-axis. For plots in Figure 4, we have used a bin size of 0.05.

Reliability diagrams are shown for sample data sets Adult, Census-income, Connect-4, Localization, Magic, Page-blocks, Pendigits, Satellige and Sign. One can see that WANBIA often attains far better calibrated class membership probability estimates.

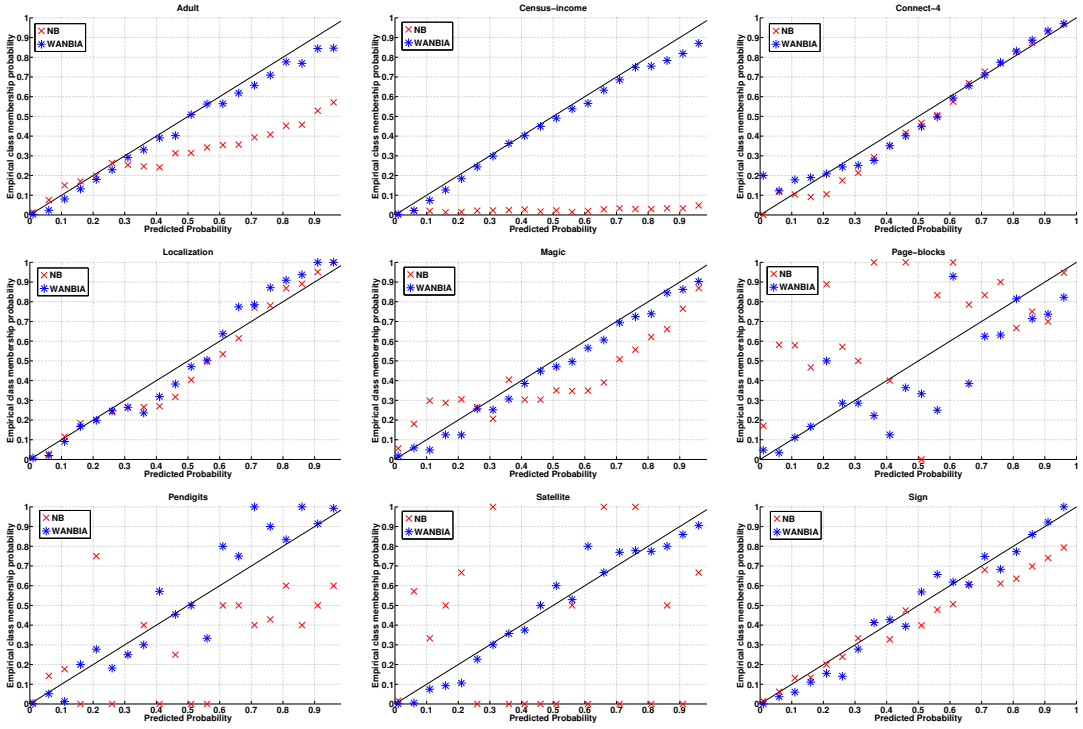



Figure 4: Reliability diagrams of naive Bayes and WANBIA on nine data sets.

### 5.5 Single versus Multiple Naive Bayes Weights Learning

To study the effect of single versus multiple weight learning for naive Bayes (naive Bayes in Equation 5 versus naive Bayes in Equation 6), we constrained WANBIA to learn only a single weight for all attributes. The method is denoted by WANBIA-S and compared with WANBIA and naive Bayes in Table 8.

It can be seen that learning multiple weights result in significantly better 0-1 loss, bias and RMSE as compared to learning a single weight but significantly worst variance. This is again

| | vs. WANBIA | | vs. NB | |
|---|---|---|---|---|
| | W/D/L | $p$ | W/D/L | $p$ |
| Bias | 5/7/61 | <**0.001** | 27/18/28 | 1 |
| Variance | 46/7/20 | **0.001** | 29/21/23 | 0.488 |
| 0-1 Loss | 17/7/49 | <**0.001** | 30-18/25 | 0.590 |
| RMSE | 19/7/47 | <**0.001** | 52/15/6 | <**0.001** |

Table 8: Win/Draw/Loss: WANBIA-S vs. WANBIA and NB

the effect of the bias-variance trade-off. Learning multiple weights result in lowering the bias but increases the variance of classification. As can be seen from the table, the performance of WANBIA-S compared to NB is fairly even in terms of 0-1 loss, bias and variance and WDL results are non-significant. However, RMSE is significantly improved as WANBIA-S improves naive Bayes probability estimates on 52 of the 73 data sets.

### 5.6 WANBIA versus Weighted Naive Bayes Using Feature Weighting Methods

The Win/Draw/Loss results of WANBIA against GRW, SBC, MH and CFS weighting NB techniques are given in Table 9. It can be seen that WANBIA has significantly better 0-1 loss, bias and RMSE than all other methods. Variance is, however, worst comparing to GRW, CFS and SB.

| | | vs. GRW | vs. SBC | vs. MH | vs. CFS | vs. SB |
|---|---|---|---|---|---|---|
| Bias | | 60/0/13 | 64/1/8 | 62/3/8 | 63/4/6 | 61/5/7 |
| | $p$ | <**0.001** | **0.048** | <**0.001** | **0.048** | <**0.001** |
| Variance | | 31/1/41 | 46/1/26 | 28/2/43 | 21/4/48 | 29/3/41 |
| | $p$ | 0.288 | **0.012** | 0.095 | **0.001** | 0.188 |
| 0-1 Loss | | 58/0/15 | 66/1/6 | 57/2/14 | 50/3/20 | 52/3/18 |
| | $p$ | <**0.001** | <**0.001** | <**0.001** | <**0.001** | <**0.001** |
| RMSE | | 65/1/7 | 62/2/9 | 54/2/17 | 50/4/19 | 52/3/18 |
| | $p$ | <**0.001** | <**0.001** | <**0.001** | <**0.001** | <**0.001** |

Table 9: Win/Draw/Loss: WANBIA versus Feature Weighting Methods

### 5.7 WANBIA versus Selective Naive Bayes

In this section, we will compare WANBIA performance with that of selective naive Bayes classifiers SBC-FS and MH-FS. The Win/Draw/Loss results are given in Table 10. It can be seen that WANBIA has significantly better 0-1 loss, bias and RMSE as compared to SBC-FS and MH-FS. It also has better variance as compared to the other methods.

### 5.8 WANBIA versus Ferreira et al. Approach

WANBIA comparison with Ferreira et al. (2001) approaches FNB-d1 and FNB-d2 in terms of 0-1 loss, RMSE, bias and variance is given in Table 11. WANBIA has significantly better 0-1 loss, bias and RMSE and non-significantly worst variance as compared to the other methods.

| | **vs. SBC-FS** | | **vs. MH-FS** | |
|---|---|---|---|---|
| | W/D/L | $p$ | W/D/L | $p$ |
| Bias | 58/3/12 | <**0.001** | 58/6/9 | <**0.001** |
| Variance | 52/3/18 | <**0.001** | 37/6/30 | 0.463 |
| 0-1 Loss | 65/3/5 | <**0.001** | 56/6/11 | <**0.001** |
| RMSE | 65/3/5 | <**0.001** | 64/5/4 | <**0.001** |

Table 10: Win/Draw/Loss: WANBIA$^{\text{CLL}}$, WANBIA$^{\text{MSE}}$ vs. SBC-FS and MH-FS

| | **vs. FNB-d1** | | **vs. FNB-d2** | |
|---|---|---|---|---|
| | W/D/L | $p$ | W/D/L | $p$ |
| Bias | 70/2/1 | <**0.001** | 64/1/8 | <**0.001** |
| Variance | 27/3/43 | **0.072** | 30/1/42 | 0.194 |
| 0-1 Loss | 58/2/13 | <**0.001** | 59/1/13 | <**0.001** |
| RMSE | 56/1/16 | <**0.001** | 59/1/13 | <**0.001** |

Table 11: Win/Draw/Loss: WANBIA vs. FNB-d1 and FNB-d2

### 5.9 Discussion

In this section, we discuss the significance of the results presented in the Sections 5.6, 5.7 and 5.8 using Friedman and Nemenyi tests. Following the graphical representation of Demšar (2006), we show the comparison of techniques WANBIA, GRW, SBC, MH, CFS, SB, FNB-d1, FNB-d2, SBC-FS and MH-FS against each other on each metric, that is, 0-1 loss, RMSE, bias and variance.

We plot the algorithms on a vertical line according to their ranks, the lower the better. Ranks are also displayed on a parallel vertical line. Critical difference is also plotted. Algorithms are connected by a line if their differences are not significant. This comparison involves 10 ($a = 10$) algorithms with 73 ($D = 73$) data sets. The Friedman statistic is distributed according to the $F$ distribution with $a-1 = 9$ and $(a-1)(D-1) = 648$ degrees of freedom. The critical value of $F(9,648)$ for $\alpha = 0.05$ is 1.8943. The Friedman statistics for 0-1 loss, bias, variance and RMSE in our experiments are 18.5108, 24.2316, 9.7563 and 26.6189 respectively. Therefore, the null hypotheses were rejected. The comparison using Nemenyi test on bias, variance, 0-1 loss and RMSE is shown in Figure 5.9.

As can be seen from the figure, the rank of WANBIA is significantly better than that of other techniques in terms of the 0-1 loss and bias. WANBIA ranks first in terms of RMSE but its score is not significantly better better than that of SB. Variance-wise, FNB-d1, GRW, CFS, FNB-d2 and MH have the top five ranks with performance not significantly different among them, whereas WANBIA stands eighth, with rank not significantly different from GRW, fnbd1, MH, SB, fnbd2 and MH-FS.

### 5.10 WANBIA versus Semi-naive Bayes Classification

In this section, we will compare WANBIA with semi-naive Bayes methods Tree Augmented Naive Bayes (TAN) and Average $n$-Dependence Estimators (AnDE). AnDE provides a family of classification algorithms that includes naive Bayes when $n = 0$. As $n$ increases, bias decreases and variance of classification increases. We will constrain to A1DE in this work. A1DE relaxes NB's attribute independence assumption by (only) making each attribute independent given the class and one at-

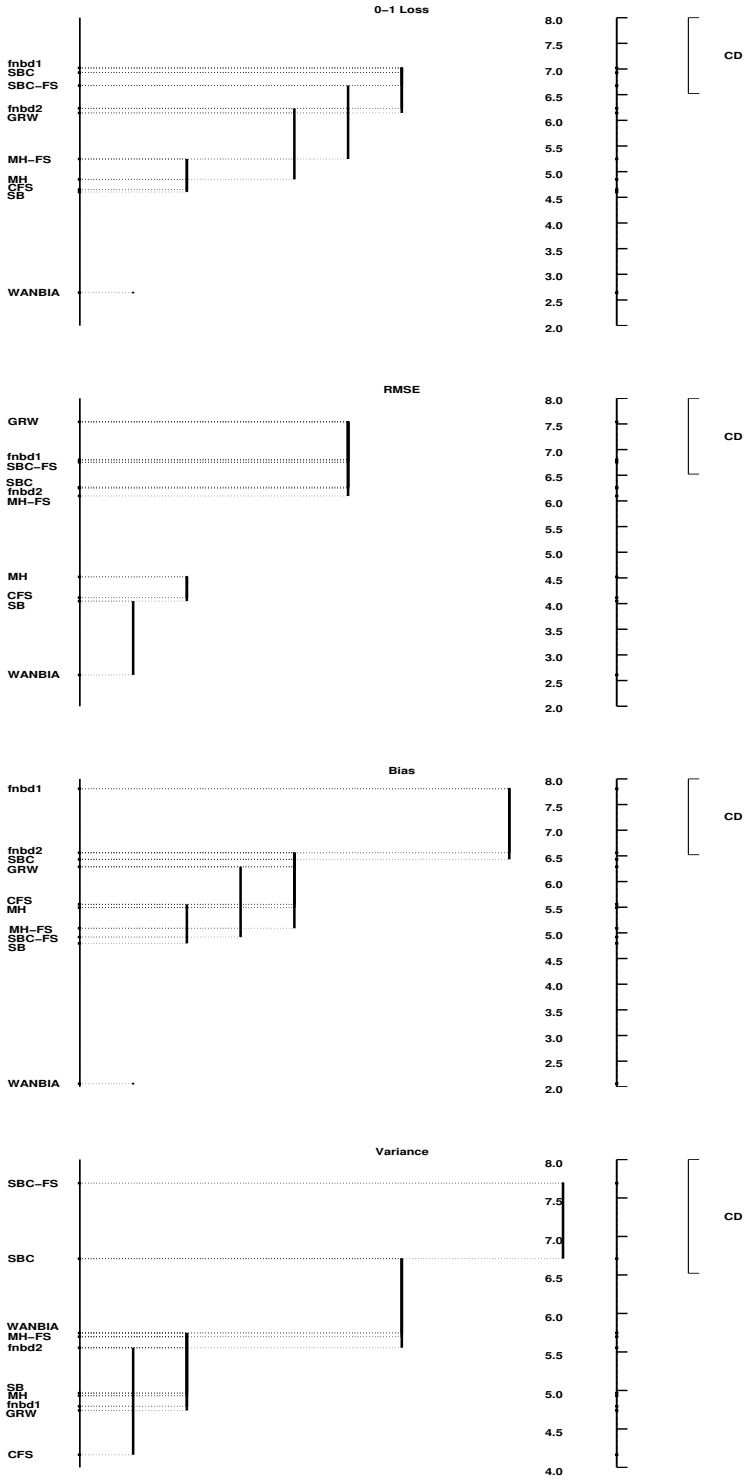



Figure 5: 0-1 Loss, RMSE, Bias, Variance comparison of 10 algorithms (GRW, SBC, MH, CFS, SB, SBC-FS, MH-FS, FNB-d1, FNB-d2, WANBIA) with the Nemenyi test on 73 data sets. CD = 1.4778.

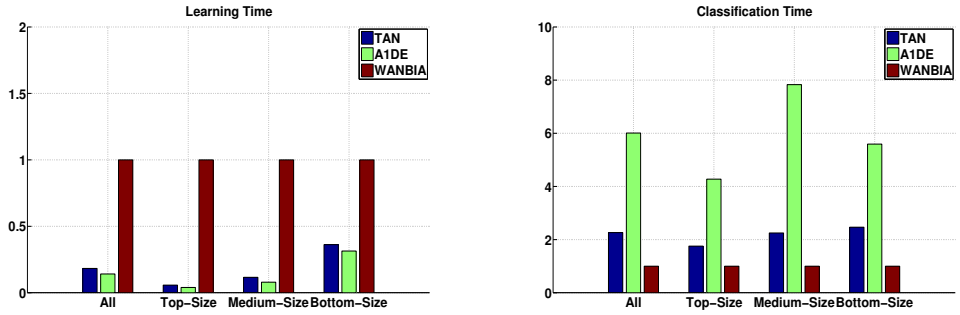



Figure 6: Averaged learning time (left) and classification time (right) of TAN, A1DE and WANBIA on all 73, Top, Medium and Bottom size data sets. Results are normalized with respect to WANBIA and geometric mean is reported.

tribute, the super-parent. This results in a one-dependence classifier. A1DE is an ensemble of these one-dependence classifiers. As A1DE is based on learning without search, every attribute takes a turn to be a super-parent. A1DE estimates by averaging over all estimates of $\mathrm{P}(y,\mathbf{x})$, that is:

$$\hat{\mathrm{P}}(y,\mathbf{x}) \quad = \quad \frac{1}{a}\sum_{i=1}^{a}\hat{\mathrm{P}}(y,x_i)\hat{\mathrm{P}}(\mathbf{x}|y,x_i).$$

Similarly, TAN augments the naive Bayes structure by allowing each attribute to depend on at most one non-class attribute. Unlike A1DE, it is not an ensemble and uses an extension of the Chow-Liu tree that uses conditional mutual information to find a maximum spanning tree as a classifier. The estimate is:

$$\hat{\mathrm{P}}(y,\mathbf{x}) \quad = \quad \hat{\mathrm{P}}(y)\prod_{i=1}^{a}\hat{\mathrm{P}}(x_i|y,\pi(x_i)),$$

where $\pi(x_i)$ is the parent of attribute $x_i$.

Bias-variance analysis of WANBIA with respect to TAN and A1DE is given in Table 12 showing that WANBIA has similar bias-variance performance to A1DE and significantly better variance performance to TAN with slightly worst bias. Considering, TAN is a low bias high variance learner, it should be suitable for large data. This can be seen in Table 13 where TAN has significantly better 0-1 Loss and RMSE performance than WANBIA on large data sets and significantly worst on small. The average learning and classification time comparison of WANBIA and TAN is given in Figure 6 and scatter of the actual time values is given in Figures 7 and 8. Even though, TAN is competitive to WANBIA in terms of learning time (training TAN involves a simple optimization step), we claim that WANBIA's improved performance on medium and small size data sets is very encouraging.

WANBIA has similar bias-variance profile to A1DE and one can expect it to perform more or less like A1DE. This can be seen from the 0-1 Loss and RMSE comparison results given in Table 14. Most of the results are not significant, except on large data sets where A1DE is more effective. However, this improved performance for larger data sets has a toll associated with it. As can be seen from Figure 6 (right) and Figure 8, A1DE is extremely computationally expensive especially at (critical) classification time. Even though, WANBIA and A1DE performs in a similar

| | vs. TAN | | vs. A1DE | |
|---|---|---|---|---|
| | W/D/L | $p$ | W/D/L | $p$ |
| Bias | 31/2/40 | 0.342 | 35/3/35 | 1.09 |
| Variance | 61/2/10 | <**0.001** | 35/3/35 | 1.09 |

Table 12: Win/Draw/Loss: Bias-variance analysis of WANBIA, TAN and A1DE

| | | | Size | | |
|---|---|---|---|---|---|
| | | All | Top | Medium | Bottom |
| 0-1 Loss | | 48/2/23 | 2/0/10 | 14/1/6 | 32/1/7 |
| | $p$ | **0.004** | **0.038** | 0.115 | <**0.001** |
| RMSE | | 46/1/26 | 2/0/10 | 14/1/6 | 30/0/10 |
| | $p$ | **0.024** | **0.038** | 0.115 | **0.002** |

Table 13: Win/Draw/Loss: WANBIA versus TAN

fashion on small and medium size data sets, WANBIA offers a huge improvement over the state of the art by offering a faster algorithm at classification time. Note, that training A1DE does not involve any optimization step and hence offers a fast training step as compared to other traditional learning algorithms.

| | | | Size | | |
|---|---|---|---|---|---|
| | | All | Top | Medium | Bottom |
| 0-1 Loss | | 31/4/38 | 2/1/9 | 10/1/10 | 19/2/19 |
| | $p$ | 0.470 | 0.065 | 1.176 | 1.128 |
| RMSE | | 30/3/40 | 2/0/10 | 9/1/11 | 19/2/19 |
| | $p$ | 0.282 | **0.038** | 0.823 | 1.128 |

Table 14: Win/Draw/Loss: WANBIA versus A1DE

### 5.11 WANBIA versus Random Forest

Random Forest (RF) (Breiman, 2001) is considered to be a state of the art classification scheme. RFs consist of multiple decision trees, each tree is trained on data selected at random but with replacement from the original data (bagging). For example, if there are $N$ data points, select $N$ data points at random with replacement. If there are $n$ attributes, a number $m$ is specified, such that $m < n$. At each node of the decision tree, $m$ attributes are randomly selected out of $n$ and are evaluated, the best being used to split the node. Each tree is grown to its largest possible size and no pruning is done. Classifying an instance encompasses passing it through each decision tree and the output is determined by the mode of the output of decision trees. We used 100 decision trees in this work.

Bias-variance comparison of WANBIA and RF in Table 15 suggests that RF is a low bias and high variance classifier. Like TAN, one can expect it to work extremely well on large data sets. This is evident from Table 16 where 0-1 Loss and RMSE of RF and WANBIA is compared. Note, we were unable to compute results for RF on our two largest data sets Poker-hand and Covertype

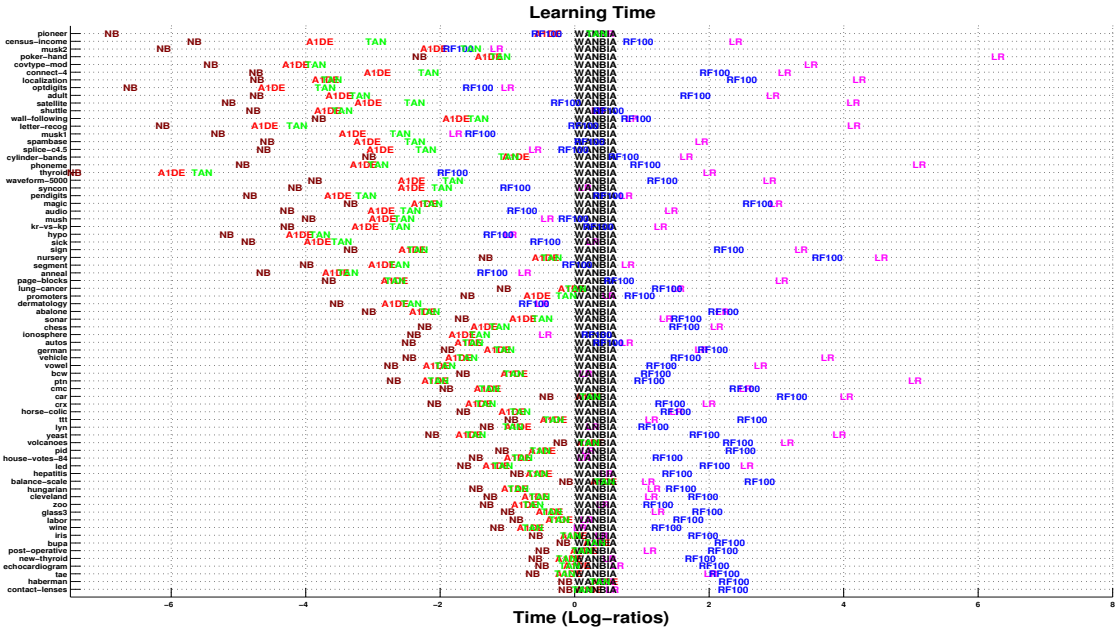



Figure 7: Learning time comparison of NB, A1DE, TAN, RF, LR and WANBIA on all 73 data sets. Results are normalized with respect to WANBIA and log-ratios are plotted.

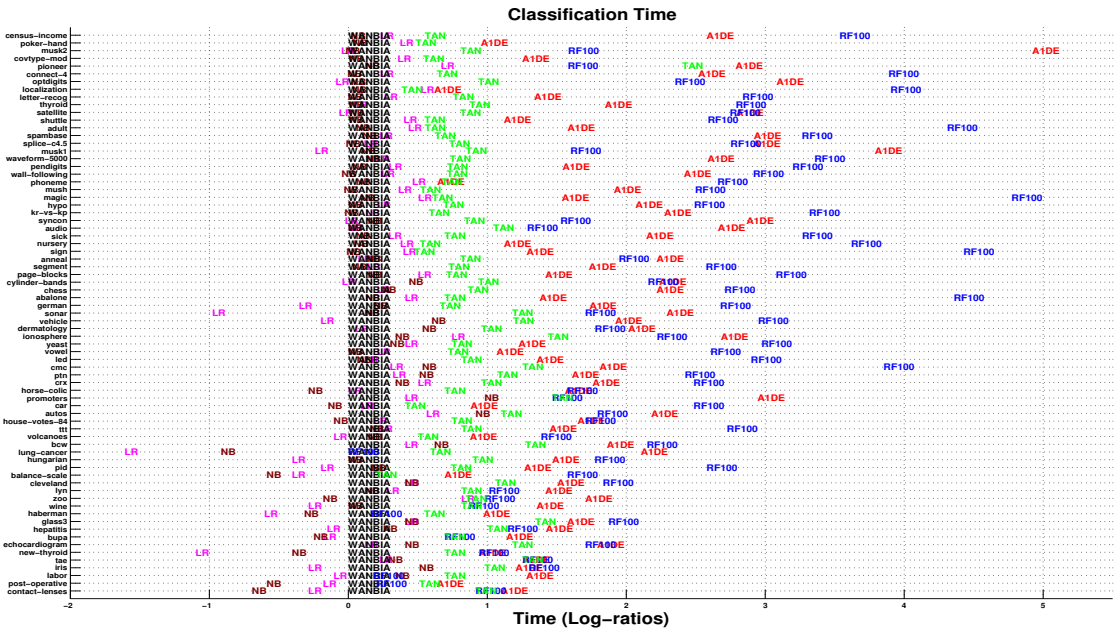



Figure 8: Classification time comparison of NB, A1DE, TAN, RF, LR and WANBIA on all 73 data sets. Results are normalized with respect to WANBIA and log-ratios are plotted.

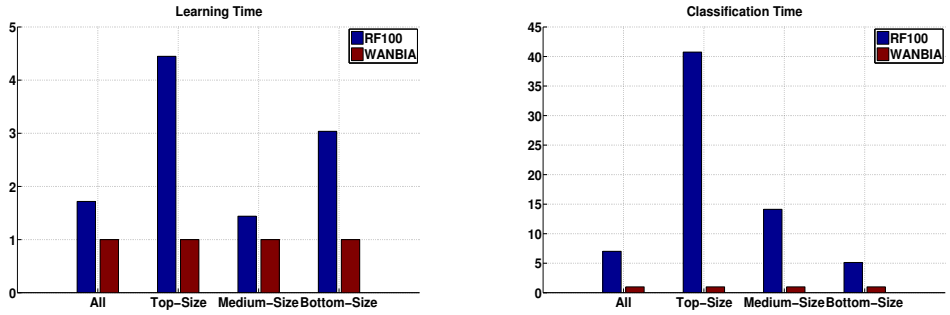



Figure 9: Averaged learning time (left) and classification time (right) of RF and WANBIA on all 73, Top, Medium and Bottom size data sets. Results are normalized with respect to WANBIA and geometric mean is reported.

(Table 4). Even with 32 GB of RAM, Weka exhausted heap memory during cross-validation experiments on RF for these data sets. However, due to its low bias one would expect RF to beat WANBIA on these two data sets, resulting in W/D/L of $2/0/10$ and $3/0/9$ on largest data sets for 0-1 Loss and RMSE with a significance of **0.038** and 0.146.

| | **vs. RF** | |
|---|---|---|
| | W/D/L | $p$ |
| Bias | 21/2/48 | **0.001** |
| Variance | 53/3/16 | $<$**0.001** |

Table 15: Win/Draw/Loss: Bias-variance analysis of WANBIA and RF

| | | | Size | | |
|---|---|---|---|---|---|
| | | All | Top | Medium | Bottom |
| 0-1 Loss | | 38/1/32 | 2/0/8 | 7/1/13 | 29/0/11 |
| | $p$ | 0.550 | 0.109 | 0.263 | **0.006** |
| RMSE | | 42/1/28 | 3/0/7 | 12/1/8 | 27/0/13 |
| | $p$ | 0.119 | 0.343 | 0.503 | **0.038** |

Table 16: Win/Draw/Loss: WANBIA versus RF

On smaller data sets WANBIA has a better 0-1 Loss performance and significantly better RMSE than RF. This is packaged with WANBIA's far superior learning and classification timings over RF as can be seen from Figures 7, 8 and 9,. This makes WANBIA an excellent alternative to RF especially for small data.

### 5.12 WANBIA versus Logistic Regression

In this section, we compare the performance of WANBIA with state of the art discriminative classifier Logistic Regression (LR). We implemented LR as described in Roos et al. (2005). The following

objective function is optimized:

$$\begin{aligned} \text{CLL}(\beta) &= \sum_{j=1}^{|\mathcal{D}|} \log \hat{\text{P}}(y|\mathbf{x}) \\ &= \sum_{j=1}^{|\mathcal{D}|} \left( \beta_y^T \mathbf{x} - \log \left( \sum_{k'=1}^{K} \exp(\beta_{k'}^T \mathbf{x}) \right) \right). \end{aligned}$$

The gradient of $\text{CLL}(\beta)$ is:

$$\begin{aligned} \frac{\partial \text{CLL}(\beta)}{\partial \beta_{i,k}} &= \sum_{j=1}^{|\mathcal{D}|} \delta(y=k) x_i - \frac{\exp(\beta_k^T \mathbf{x})}{\sum_{k'=1}^{K} \exp(\beta_{k'}^T \mathbf{x})} x_i \\ &= \sum_{j=1}^{|\mathcal{D}|} \left( \delta(y=k) - \hat{\text{P}}(k \,|\, \mathbf{x}) \right) x_i. \end{aligned}$$

The same L-BFGS-M optimization procedure of Zhu et al. (1997) that is used for optimizing WANBIA parameters is used to learn LR parameters. For $L_2$ regularization, the following objective function is optimized:

$$\text{CLL}(\beta) = \sum_{j=1}^{|\mathcal{D}|} \log \hat{\text{P}}(y|\mathbf{x}) + C\|\beta\|^2,$$

where the value of $C$ is found using 3-fold cross validation over the training data by searching $C$ from the list: $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}, 10^{5}, 10^{6}\}$. The value of $C$ resulting in lowest 0-1 error is chosen.

Table 17 compares the bias and variance of WANBIA with respect to LR and regularized LR. Like RF, LR is a low bias classifier. Regularizing LR reduces its variance at the expense of increasing its bias. However, it is encouraging to see that WANBIA still has lower variance than regularized LR winning on 47, drawing on five and losing only on 20 data sets.

| | **vs. LR** | | **vs. Regularized LR** | |
|---|---|---|---|---|
| | W/D/L | $p$ | W/D/L | $p$ |
| Bias | 18/3/51 | $<$**0.001** | 10/2/60 | $<$**0.001** |
| Variance | 50/5/17 | $<$**0.001** | 47/5/20 | **0.001** |

Table 17: Win/Draw/Loss: Bias-variance analysis of WANBIA, LR and regularized LR

The error of WANBIA is compared with LR in Table 18. It can be seen that LR is superior to WANBIA on large data sets. Regularized LR is very difficult to beat as can be seen from Table 19. Regularized LR results in significantly better performance on not only large but also on medium size data sets. However, WANBIA still maintains its effectiveness on small data. This is, again, extremely encouraging. The comparison of LR and WANBIA's learning and classification time is given in Figures 7, 8 and 10. Note, that we have only reported timing results for un-regularized LR, which is more efficient than regularized LR. The regularized results are given in the appendix, but are not compared as, due to their computational intensity, they were computed on a Grid comprising

| | | | Size | | |
|---|---|---|---|---|---|
| | | All | Top | Medium | Bottom |
| 0-1 Loss | | 32/3/37 | 0/1/11 | 6/1/13 | 26/1/13 |
| | $p$ | 0.630 | <**0.001** | 0.167 | 0.053 |
| RMSE | | 40/4/28 | 0/1/11 | 8/1/11 | 32/2/6 |
| | $p$ | 0.181 | <**0.001** | 0.647 | <**0.001** |

Table 18: Win/Draw/Loss: WANBIA versus LR

| | | | Size | | |
|---|---|---|---|---|---|
| | | All | Top | Medium | Bottom |
| 0-1 Loss | | 20/2/50 | 0/1/11 | 2/1/17 | 21/0/19 |
| | $p$ | <**0.001** | <**0.001** | <**0.001** | 0.874 |
| RMSE | | 30/2/40 | 0/1/11 | 3/1/16 | 27/0/13 |
| | $p$ | 0.282 | <**0.001** | **0.004** | **0.038** |

Table 19: Win/Draw/Loss: WANBIA versus Regularized LR

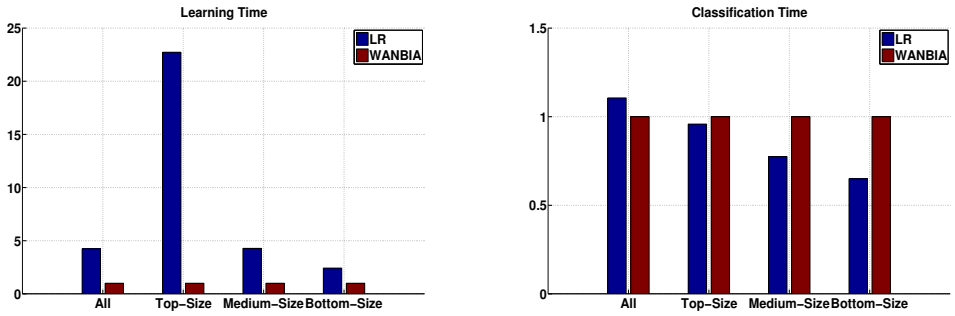



Figure 10: Averaged learning time (left) and classification time (right) of LR and WANBIA on all 73, Top, Medium and Bottom size data sets. Results are normalized with respect to WANBIA and geometric mean is reported.

computers with 4 GB of RAM and 2.0 Ghz processor. Since the environment was not controlled, we do not include them in our comparison.

Logistic Regression's better performance on large data sets and marginally better performance on medium size data sets has a computational cost associated with it. This can be seen in Figure 7. A cross-validation procedure to tune $C$ parameters, as required for regularization, increases already relatively high costs to new heights. Therefore, WANBIA can be viewed as a substitute over LR for medium size and regularized LR on smaller data sets.

### 5.13 Summary of Experimental Results

We summarize our results as follows:

- WANBIA is shown to greatly improve upon naive Bayes' generalization performance. This performance gain is attributed to the fact that WANBIA successfully alleviates the conditional attribute independence assumption. Traditionally, NB is preferred for small and medium size data due to its high bias and low variance. Our results suggest that WANBIA is likely to be preferable to NB for small and medium size data sets.

- WANBIA has significantly better 0-1 loss, RMSE, bias and variance than most existing weighted naive Bayes schemes based on attribute selection and attribute weighting. As a result, WANBIA sets a new standard for attribute weighing for naive Bayes.

- WANBIA is competitive to state-of-the-art semi-naive Bayes methods TAN and A1DE. WANBIA has an edge over TAN on medium and small data sets, whereas its computational efficiency makes it a useful alternative over A1DE. However, it is credible that WANBIA's strategy for alleviating NB's attribute independence assumption is complementary to A1DE and TAN, allowing both to be applied to obtain even better classification accuracy.

- WANBIA performs significantly better on smaller data sets compared to Random Forest with 100 decision trees. While Random Forest is likely to be more accurate on larger data sets, WANBIA provides a computationally efficient alternative that may be attractive when computational burden is an issue.

- WANBIA is more accurate than both regularized and unregularized Logistic Regression on smaller data. Also, for multi-class and multi-valued data, WANBIA is based on optimizing far fewer parameters and, therefore, is computationally more efficient than LR.

## 6. Conclusions and Further Research

In this work we have introduced weighting schemes to incorporate weights in naive Bayes. Our work has been primarily motivated by the observation that naive Bayes conditional attribute independence assumption is often violated and, therefore, it is useful to alleviate it. We build an argument that in current research, weighting in naive Bayes has been viewed as a way of enhancing the impact of attributes that are highly correlated with the class. We argue that weighting provides a natural framework for alleviating the attribute independence assumption. Our two proposed naive Bayes weighting methods WANBIA$^{\text{CLL}}$ and WANBIA$^{\text{MSE}}$ fix naive Bayes' parameters to be the MAP estimates and learn weights by maximizing conditional log-likelihood and minimizing Mean Square Error respectively. This scheme results in the need to optimize significantly fewer parameters than LR.

Conditional log likelihood and mean square error are not the only objective functions that can be optimized to learn weights for WANBIA. One can use, for instance, Hinge Loss (generally used with SVM) and exponential loss (boosting). Another alternative is using a different form of mean square error that is $\frac{1}{2}\sum_{x^{(j)} \in |\mathcal{D}|}(1-\hat{\mathrm{P}}(y|\mathbf{x}))^2$ instead of $\frac{1}{2}\sum_{x^{(j)} \in |\mathcal{D}|}\sum_{y}(\mathrm{P}(y|\mathbf{x})-\hat{\mathrm{P}}(y|\mathbf{x}))^2$. A comparison of WANBIA trained using these objective functions has been left to future work.

In interpreting the results presented in this work, it is important to keep in mind that attribute weighting and semi-naive Bayesian relaxation of the attribute independence assumption are mutually compatible. It remains a direction for future research to explore techniques for attribute weighting in the context of semi-naive Bayes classifiers.

We have constrained ourselves in this work to weighting of the form of Equation 5. It will be interesting to optimize weights as in Equation 4, that is, optimizing a weight for each attribute's value. A next step will be to learn a weight for each attribute value per class and a weight for the prior probabilities. Such a variant of WANBIA would have the same number of parameters to optimize as Logistic Regression. For example, for $i$-th attribute and $y$-th class, weight term constitutes $\beta_{i,y}$ for LR and $w_i \log \theta_{x_i|y}$ for WANBIA.

In conclusion, with modest computation, WANBIA substantially decreases the bias of naive Bayes without unduly increasing its variance. The resulting classifier is highly competitive with the state of the art when learning from small and medium size data sets.

## Acknowledgments

This research has been supported by the Australian Research Council under grant DP110101427 and Asian Office of Aerospace Research and Development, Air Force Office of Scientific Research under contract FA23861214030. The authors would like to thank Mark Hall for providing the code for CFS and MH. The authors would also like to thank anonymous reviewers for their insightful comments that helped improving the paper tremendously.

## Appendix A. Code and Detailed Results

The code of the methods proposed in this work can be obtained from the website, `http://sourceforge.net/projects/rawnaivebayes/`. This appendix presents the detailed results for Error (Table 20), RMSE (Table 21), Bias (Table 22), Variance (Table 23), Train time (Table 24) and Test time (Table 25).

## References

R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

B. Bjerregaard, S. Brynitz, J. Holst-Christensen, E. Kalaja, J. Lund-Kristensen, J. Hilden, F. T. deDombal, and J. C. Horrocks. Computer-aided diagnosis of acute abdomen: a system from leeds used on copenhagen patients. In *In Decision Making and Medical Care: Can Information Science Help*. North-Holland Publishing Company, 1976.

L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

A. Carvalho, T. Roos, A. Oliveira, and P. Myllymaki. Discriminative learning of Bayesian networks via factorized conditional log-likelihood. *Journal of Machine Learning Research*, 2011.

J. Cerquides and R. L. De Mántaras. Tan classifiers based on decomposable distributions. *Machine Learning*, 59(3):323–354, 2005a.

J. Cerquides and R. L. De Mántaras. Robust Bayesian linear classifier ensembles. In *Proceedings of the Sixteenth European Conference on Machine Learning*, pages 70–81, 2005b.

| | A1DE | LR | LR-Reg | NB | WANBIA$^{CLL}$ | WANBIA$^{MSE}$ | RF100 | TAN |
|---|---|---|---|---|---|---|---|---|
| Mush | 0.0003 | 0 | 0 | 0.0261 | 0.0012 | 0.001 | 0 | 0.0004 |
| Shuttle | 0.0012 | 0.0004 | 0.0004 | 0.004 | 0.0014 | 0.0012 | 0.0009 | 0.001 |
| Pioneer | 0.0025 | 0.0079 | 0.0079 | 0.0036 | 0.0002 | 0.0001 | 0.0008 | 0.0057 |
| Syncon | 0.0033 | 0.0102 | 0.0107 | 0.0133 | 0.0114 | 0.0118 | 0.0127 | 0.0115 |
| Hypo | 0.0115 | 0.0062 | 0.0049 | 0.0146 | 0.0083 | 0.0082 | 0.0122 | 0.0187 |
| Wine | 0.0174 | 0.0185 | 0.0199 | 0.014 | 0.0079 | 0.0112 | 0.0211 | 0.0225 |
| Anneal | 0.0188 | 0.0117 | 0.0094 | 0.0361 | 0.0199 | 0.019 | 0.0122 | 0.0266 |
| Pendigits | 0.0197 | 0.0413 | 0.0288 | 0.1179 | 0.1038 | 0.1018 | 0.0339 | 0.0411 |
| Dermatology | 0.0213 | 0.0288 | 0.031 | 0.0201 | 0.019 | 0.021 | 0.0367 | 0.0434 |
| Sick | 0.0263 | 0.0269 | 0.0256 | 0.0312 | 0.0263 | 0.0264 | 0.0263 | 0.0272 |
| Page-blocks | 0.0305 | 0.0368 | 0.0318 | 0.0609 | 0.0373 | 0.0351 | 0.0309 | 0.0412 |
| Optdigits | 0.033 | 0.0533 | 0.0396 | 0.0763 | 0.0642 | 0.0658 | 0.0458 | 0.0487 |
| Bcw | 0.0371 | 0.0471 | 0.0409 | 0.0266 | 0.0343 | 0.0368 | 0.0386 | 0.0513 |
| Segment | 0.0388 | 0.055 | 0.0479 | 0.0752 | 0.0502 | 0.0505 | 0.0413 | 0.0528 |
| Splice-c4.5 | 0.0404 | 0.0692 | 0.045 | 0.0463 | 0.0411 | 0.04 | 0.0489 | 0.0613 |
| New-thyroid | 0.0423 | 0.0514 | 0.0528 | 0.0374 | 0.0398 | 0.0412 | 0.0479 | 0.0514 |
| Musk2 | 0.0438 | 0.0207 | 0.0143 | 0.0784 | 0.0399 | 0.04 | 0.0385 | 0.0494 |
| Labor | 0.0465 | 0.0737 | 0.0816 | 0.0544 | 0.0544 | 0.0561 | 0.0939 | 0.0842 |
| Wall-following | 0.0479 | 0.0118 | 0.0085 | 0.0957 | 0.0219 | 0.0206 | 0.0216 | 0.0693 |
| House-votes-84 | 0.0555 | 0.0493 | 0.0461 | 0.0976 | 0.0452 | 0.0463 | 0.0416 | 0.0649 |
| Iris | 0.0577 | 0.057 | 0.0573 | 0.0553 | 0.053 | 0.0543 | 0.056 | 0.0587 |
| Zoo | 0.0629 | 0.0663 | 0.0683 | 0.0713 | 0.0723 | 0.0757 | 0.0743 | 0.0931 |
| Spambase | 0.0662 | 0.0626 | 0.0588 | 0.0979 | 0.0626 | 0.0611 | 0.0575 | 0.0689 |
| Ionosphere | 0.0701 | 0.0869 | 0.0818 | 0.0868 | 0.0779 | 0.0746 | 0.0766 | 0.0781 |
| Nursery | 0.0744 | 0.0747 | 0.0747 | 0.0979 | 0.0979 | 0.0979 | 0.0248 | 0.0686 |
| Thyroid | 0.0752 | 0.0683 | 0.0642 | 0.1116 | 0.1061 | 0.0994 | 0.075 | 0.0855 |
| Musk1 | 0.0869 | 0.0877 | 0.0811 | 0.1325 | 0.0649 | 0.0666 | 0.0683 | 0.0763 |
| Kr-vs-kp | 0.0915 | 0.0277 | 0.0286 | 0.1267 | 0.0696 | 0.0622 | 0.0128 | 0.0772 |
| Census-income | 0.0986 | 0.0433 | 0.0433 | 0.2355 | 0.0462 | 0.0461 | 0.0494 | 0.0574 |
| Letter-recog | 0.1025 | 0.1639 | 0.1495 | 0.2563 | 0.2484 | 0.2475 | 0.0902 | 0.151 |
| Car | 0.1069 | 0.0742 | 0.0733 | 0.156 | 0.156 | 0.156 | 0.0772 | 0.081 |
| Satellite | 0.1092 | 0.1807 | 0.1175 | 0.1751 | 0.1553 | 0.1514 | 0.1085 | 0.1179 |
| Chess | 0.1318 | 0.1233 | 0.1242 | 0.1364 | 0.1338 | 0.132 | 0.1074 | 0.106 |
| Waveform-5000 | 0.1427 | 0.147 | 0.1386 | 0.1918 | 0.1556 | 0.1565 | 0.1558 | 0.1825 |
| Crx | 0.1427 | 0.1564 | 0.1396 | 0.1449 | 0.138 | 0.1402 | 0.1581 | 0.1651 |
| Sonar | 0.144 | 0.1784 | 0.1637 | 0.1519 | 0.1688 | 0.1671 | 0.1704 | 0.1683 |
| Adult | 0.1491 | 0.1274 | 0.1274 | 0.159 | 0.1306 | 0.1303 | 0.1466 | 0.1387 |
| Hungarian | 0.1592 | 0.1811 | 0.181 | 0.1585 | 0.1701 | 0.1692 | 0.1874 | 0.166 |
| Hepatitis | 0.1619 | 0.1923 | 0.17 | 0.1574 | 0.1452 | 0.149 | 0.1606 | 0.1713 |
| Lyn | 0.1669 | 0.1986 | 0.1858 | 0.1666 | 0.1801 | 0.1797 | 0.1909 | 0.2257 |
| Magic | 0.1696 | 0.1538 | 0.1537 | 0.2169 | 0.172 | 0.1716 | 0.1674 | 0.1619 |
| Cleveland | 0.171 | 0.1863 | 0.1766 | 0.1693 | 0.1764 | 0.179 | 0.1908 | 0.1909 |
| Glass3 | 0.1724 | 0.2007 | 0.1776 | 0.1871 | 0.1757 | 0.1778 | 0.1951 | 0.1846 |
| Autos | 0.1983 | 0.2154 | 0.2059 | 0.2554 | 0.23 | 0.2302 | 0.1937 | 0.2437 |
| Promoters | 0.1986 | 0.1241 | 0.1302 | 0.1387 | 0.1358 | 0.1363 | 0.1519 | 0.2325 |
| Horse-colic | 0.2107 | 0.2726 | 0.1798 | 0.2126 | 0.1622 | 0.1659 | 0.1789 | 0.2236 |
| Pid | 0.2193 | 0.2197 | 0.2198 | 0.2215 | 0.2151 | 0.2152 | 0.2536 | 0.2249 |
| Vowel | 0.2199 | 0.2413 | 0.2371 | 0.3931 | 0.3617 | 0.3646 | 0.1674 | 0.2781 |
| Cylinder-bands | 0.237 | 0.24 | 0.2277 | 0.2559 | 0.2591 | 0.2659 | 0.2702 | 0.3761 |
| Covtype-mod | 0.2413 | 0.2571 | 0.2571 | 0.3117 | 0.2912 | 0.2907 | | 0.2512 |
| Connect-4 | 0.244 | 0.2425 | 0.2425 | 0.2792 | 0.2727 | 0.2726 | 0.1875 | 0.2368 |
| Ttt | 0.2502 | 0.0247 | 0.0181 | 0.2902 | 0.2731 | 0.2714 | 0.0765 | 0.2484 |
| German | 0.2535 | 0.2575 | 0.2526 | 0.2532 | 0.257 | 0.2571 | 0.2684 | 0.2838 |
| Phoneme | 0.263 | 0.2544 | 0.2068 | 0.3035 | 0.2587 | 0.2607 | 0.1789 | 0.3484 |
| Led | 0.265 | 0.2694 | 0.2659 | 0.2632 | 0.2633 | 0.2648 | 0.2802 | 0.2702 |
| Balance-scale | 0.2682 | 0.2655 | 0.2626 | 0.2594 | 0.2594 | 0.2594 | 0.271 | 0.2661 |
| Haberman | 0.2714 | 0.2708 | 0.2709 | 0.2714 | 0.2714 | 0.2714 | 0.2709 | 0.2722 |
| Vehicle | 0.2761 | 0.2845 | 0.2723 | 0.3765 | 0.3289 | 0.3288 | 0.2742 | 0.2764 |
| Sign | 0.279 | 0.32 | 0.3204 | 0.3593 | 0.3589 | 0.3568 | 0.2038 | 0.2747 |
| Audio | 0.3288 | 0.2677 | 0.2396 | 0.3305 | 0.292 | 0.2942 | 0.3009 | 0.3361 |
| Volcanoes | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 |
| Echocardiogram | 0.3328 | 0.3355 | 0.3351 | 0.3206 | 0.3298 | 0.3302 | 0.3489 | 0.3477 |
| Contact-lenses | 0.3563 | 0.2958 | 0.3604 | 0.3438 | 0.3417 | 0.3458 | 0.3438 | 0.4292 |
| Localization | 0.359 | 0.4584 | 0.4584 | 0.4939 | 0.4902 | 0.49 | 0.2976 | 0.3564 |
| Post-operative | 0.375 | 0.425 | 0.3006 | 0.3728 | 0.3611 | 0.3528 | 0.3972 | 0.3706 |
| Bupa | 0.3843 | 0.3843 | 0.3967 | 0.3843 | 0.3843 | 0.3843 | 0.3817 | 0.3843 |
| Yeast | 0.4086 | 0.4064 | 0.4059 | 0.4115 | 0.4068 | 0.4084 | 0.421 | 0.4096 |
| Abalone | 0.4562 | 0.4623 | 0.4656 | 0.4794 | 0.4647 | 0.4643 | 0.4823 | 0.4687 |
| Poker-hand | 0.4643 | 0.4988 | 0.4988 | 0.4988 | 0.4988 | 0.4988 | | 0.329 |
| Cmc | 0.4791 | 0.447 | 0.4478 | 0.4828 | 0.4695 | 0.4654 | 0.4976 | 0.465 |
| Tae | 0.5146 | 0.5351 | 0.5334 | 0.5182 | 0.5189 | 0.5096 | 0.547 | 0.5344 |
| Lung-cancer | 0.5281 | 0.5578 | 0.5953 | 0.5203 | 0.5484 | 0.5563 | 0.6 | 0.4969 |
| Ptn | 0.5383 | 0.6444 | 0.5476 | 0.5388 | 0.5456 | 0.542 | 0.6 | 0.5872 |
| Mean | 0.1781 | 0.1817 | **0.1736** | 0.2033 | 0.1886 | 0.1885 | 0.1688 | 0.1890 |
| Mean Rank | 3.7465 | 4.5273 | **3.5000** | 5.7328 | 4.2534 | 4.2808 | 4.3356 | 5.6232 |

Table 20: Error

| | **A1DE** | **LR** | **LR-Reg** | **NB** | **WANBIA$^{CLL}$** | **WANBIA$^{MSE}$** | **RF100** | **TAN** |
|---|---|---|---|---|---|---|---|---|
| Pioneer | 0.0086 | 0.0156 | 0.0140 | 0.0102 | 0.0025 | 0.0017 | 0.0361 | 0.0129 |
| Mush | 0.0136 | 0.007 | 0.007 | 0.14 | 0.0403 | 0.0379 | 0.009 | 0.0167 |
| Shuttle | 0.0167 | 0.0111 | 0.0107 | 0.0309 | 0.0178 | 0.0169 | 0.0142 | 0.0154 |
| Syncon | 0.0299 | 0.0526 | 0.0551 | 0.0632 | 0.0554 | 0.0578 | 0.1145 | 0.0557 |
| Pendigits | 0.0556 | 0.0888 | 0.0672 | 0.1418 | 0.1256 | 0.1248 | 0.0979 | 0.0802 |
| Hypo | 0.0713 | 0.0549 | 0.0473 | 0.0775 | 0.0642 | 0.0637 | 0.0715 | 0.0885 |
| Dermatology | 0.0722 | 0.0931 | 0.0954 | 0.0713 | 0.069 | 0.0723 | 0.1303 | 0.1016 |
| Anneal | 0.0725 | 0.0609 | 0.0537 | 0.0958 | 0.0735 | 0.0741 | 0.0691 | 0.0797 |
| Optdigits | 0.0747 | 0.0999 | 0.0787 | 0.1159 | 0.099 | 0.1007 | 0.1494 | 0.0906 |
| Letter-recog | 0.0754 | 0.1001 | 0.0914 | 0.1193 | 0.1136 | 0.1135 | 0.0896 | 0.0916 |
| Thyroid | 0.0759 | 0.0754 | 0.0706 | 0.097 | 0.0867 | 0.0854 | 0.077 | 0.0805 |
| Phoneme | 0.0881 | 0.0998 | 0.0782 | 0.0951 | 0.086 | 0.0865 | 0.0731 | 0.0986 |
| Wine | 0.0917 | 0.1042 | 0.1251 | 0.0819 | 0.0656 | 0.0736 | 0.1311 | 0.1105 |
| Segment | 0.0957 | 0.1234 | 0.1017 | 0.1357 | 0.1032 | 0.1045 | 0.1061 | 0.1097 |
| Page-blocks | 0.0987 | 0.1116 | 0.0986 | 0.1431 | 0.1038 | 0.1027 | 0.0974 | 0.1165 |
| Zoo | 0.1171 | 0.1333 | 0.1279 | 0.1247 | 0.1253 | 0.1264 | 0.1279 | 0.1381 |
| New-thyroid | 0.14 | 0.1797 | 0.1741 | 0.1327 | 0.1411 | 0.1425 | 0.156 | 0.1577 |
| Splice-c4.5 | 0.1435 | 0.212 | 0.1955 | 0.1536 | 0.1462 | 0.1447 | 0.2599 | 0.176 |
| Wall-following | 0.1461 | 0.0734 | 0.0598 | 0.2081 | 0.0901 | 0.0897 | 0.1206 | 0.1762 |
| Audio | 0.1484 | 0.146 | 0.13 | 0.1486 | 0.1354 | 0.135 | 0.1361 | 0.1414 |
| Sick | 0.1551 | 0.1483 | 0.1445 | 0.1681 | 0.1458 | 0.1452 | 0.1487 | 0.1499 |
| Nursery | 0.1583 | 0.1464 | 0.1464 | 0.1771 | 0.1771 | 0.1771 | 0.101 | 0.1425 |
| Iris | 0.1628 | 0.19 | 0.2162 | 0.165 | 0.1609 | 0.1637 | 0.1813 | 0.1678 |
| Vowel | 0.169 | 0.2057 | 0.1831 | 0.2206 | 0.2116 | 0.2123 | 0.1581 | 0.1886 |
| Bcw | 0.1766 | 0.2143 | 0.1844 | 0.1586 | 0.1649 | 0.1737 | 0.1796 | 0.1996 |
| Satellite | 0.1776 | 0.2449 | 0.1685 | 0.2374 | 0.1926 | 0.1916 | 0.1682 | 0.1838 |
| Labor | 0.1792 | 0.2567 | 0.2682 | 0.1961 | 0.2082 | 0.2143 | 0.2824 | 0.2481 |
| Ptn | 0.1816 | 0.2329 | 0.1809 | 0.1824 | 0.1811 | 0.1811 | 0.1899 | 0.1829 |
| Led | 0.1995 | 0.2025 | 0.2067 | 0.1988 | 0.199 | 0.1991 | 0.2091 | 0.2034 |
| Musk2 | 0.2041 | 0.14 | 0.11 | 0.2766 | 0.1733 | 0.1786 | 0.1752 | 0.2169 |
| House-votes-84 | 0.207 | 0.2191 | 0.1926 | 0.2987 | 0.185 | 0.1873 | 0.1846 | 0.2253 |
| Car | 0.2085 | 0.1655 | 0.1628 | 0.2293 | 0.2293 | 0.2293 | 0.1782 | 0.1849 |
| Localization | 0.2091 | 0.233 | 0.233 | 0.2386 | 0.2381 | 0.2381 | 0.1939 | 0.2095 |
| Covtype-mod | 0.2183 | 0.2259 | 0.2259 | 0.2494 | 0.239 | 0.2389 | | 0.2243 |
| Autos | 0.22 | 0.244 | 0.2202 | 0.2512 | 0.2281 | 0.2284 | 0.2041 | 0.2371 |
| Poker-hand | 0.2217 | 0.2382 | 0.2382 | 0.2382 | 0.2382 | 0.2382 | | 0.2124 |
| Spambase | 0.2323 | 0.2182 | 0.2115 | 0.2949 | 0.22 | 0.22 | 0.2215 | 0.2396 |
| Yeast | 0.2333 | 0.2348 | 0.2343 | 0.2341 | 0.2338 | 0.2338 | 0.2421 | 0.2353 |
| Ionosphere | 0.2529 | 0.29 | 0.2649 | 0.2868 | 0.2533 | 0.2595 | 0.2403 | 0.2651 |
| Lyn | 0.2542 | 0.3101 | 0.2729 | 0.2585 | 0.2539 | 0.2551 | 0.2701 | 0.2886 |
| Waveform-5000 | 0.2586 | 0.2647 | 0.2671 | 0.3274 | 0.2697 | 0.2698 | 0.3036 | 0.2941 |
| Kr-vs-kp | 0.2715 | 0.1533 | 0.1547 | 0.3049 | 0.269 | 0.2673 | 0.1268 | 0.2383 |
| Musk1 | 0.2731 | 0.2926 | 0.2568 | 0.3468 | 0.2263 | 0.2292 | 0.262 | 0.2515 |
| Census-income | 0.278 | 0.1807 | 0.1807 | 0.4599 | 0.1867 | 0.1866 | 0.1928 | 0.2083 |
| Glass3 | 0.294 | 0.3453 | 0.3138 | 0.3118 | 0.3032 | 0.3061 | 0.3146 | 0.3034 |
| Vehicle | 0.3046 | 0.3235 | 0.2989 | 0.3867 | 0.3238 | 0.3239 | 0.3016 | 0.3045 |
| Chess | 0.3081 | 0.3121 | 0.3017 | 0.3143 | 0.3113 | 0.3113 | 0.2771 | 0.2787 |
| Balance-scale | 0.3229 | 0.3128 | 0.3672 | 0.3276 | 0.3276 | 0.3276 | 0.3181 | 0.3242 |
| Adult | 0.3247 | 0.2967 | 0.2967 | 0.3405 | 0.3 | 0.2999 | 0.3274 | 0.3091 |
| Volcanoes | 0.326 | 0.326 | 0.326 | 0.326 | 0.326 | 0.326 | 0.326 | 0.326 |
| Crx | 0.3353 | 0.3483 | 0.3319 | 0.3414 | 0.3194 | 0.3213 | 0.3437 | 0.354 |
| Sonar | 0.3367 | 0.4163 | 0.3586 | 0.3507 | 0.3365 | 0.3364 | 0.3518 | 0.3366 |
| Connect-4 | 0.3382 | 0.3361 | 0.3361 | 0.3592 | 0.3559 | 0.3559 | 0.3057 | 0.3322 |
| Magic | 0.3491 | 0.337 | 0.3372 | 0.3916 | 0.3568 | 0.3567 | 0.3571 | 0.3425 |
| Hungarian | 0.3503 | 0.3781 | 0.3678 | 0.3659 | 0.3428 | 0.3436 | 0.369 | 0.3441 |
| Sign | 0.3508 | 0.3731 | 0.3734 | 0.3968 | 0.3899 | 0.3897 | 0.3104 | 0.3499 |
| Cleveland | 0.354 | 0.3729 | 0.3606 | 0.3642 | 0.353 | 0.355 | 0.3696 | 0.372 |
| Hepatitis | 0.3544 | 0.4358 | 0.3725 | 0.3589 | 0.3353 | 0.3368 | 0.3375 | 0.359 |
| Pid | 0.3907 | 0.3888 | 0.3896 | 0.3949 | 0.3884 | 0.3888 | 0.4247 | 0.3911 |
| Promoters | 0.3948 | 0.3271 | 0.3281 | 0.333 | 0.3277 | 0.3253 | 0.3983 | 0.444 |
| Contact-lenses | 0.395 | 0.4356 | 0.431 | 0.3846 | 0.3845 | 0.3853 | 0.4098 | 0.4477 |
| Ttt | 0.4037 | 0.1385 | 0.1293 | 0.4336 | 0.4262 | 0.4254 | 0.2916 | 0.4098 |
| Horse-colic | 0.4115 | 0.5142 | 0.3703 | 0.4227 | 0.3577 | 0.36 | 0.3762 | 0.4219 |
| German | 0.4168 | 0.4234 | 0.4145 | 0.4202 | 0.4151 | 0.4156 | 0.4211 | 0.4469 |
| Abalone | 0.4198 | 0.4208 | 0.4368 | 0.4641 | 0.4206 | 0.4204 | 0.4539 | 0.4268 |
| Haberman | 0.4212 | 0.4213 | 0.4248 | 0.4212 | 0.4212 | 0.4212 | 0.4214 | 0.4213 |
| Post-operative | 0.4281 | 0.4777 | 0.4085 | 0.4233 | 0.4191 | 0.4174 | 0.4399 | 0.4204 |
| Cmc | 0.4349 | 0.4288 | 0.4289 | 0.4463 | 0.4312 | 0.4309 | 0.4739 | 0.4358 |
| Cylinder-bands | 0.4451 | 0.4831 | 0.4161 | 0.4661 | 0.4587 | 0.4621 | 0.4157 | 0.4794 |
| Echocardiogram | 0.4506 | 0.467 | 0.4553 | 0.4491 | 0.4459 | 0.4461 | 0.4574 | 0.4627 |
| Bupa | 0.4863 | 0.4863 | 0.4878 | 0.4863 | 0.4863 | 0.4863 | 0.4862 | 0.4861 |
| Tae | 0.5093 | 0.553 | 0.5023 | 0.5135 | 0.5075 | 0.4979 | 0.4939 | 0.4857 |
| Lung-cancer | 0.5698 | 0.5945 | 0.5689 | 0.564 | 0.5707 | 0.5711 | 0.4732 | 0.5033 |
| Mean | **0.2461** | 0.2544 | 0.2403 | 0.2705 | 0.2462 | 0.2468 | 0.2469 | 0.2528 |
| Mean Rank | **3.6712** | 5.1301 | 3.7397 | 6.0684 | 3.8082 | 4.0342 | 4.4657 | 5.0821 |

Table 21: RMSE

| | A1DE | LR | LR-Reg | NB | WANBIA$^{CLL}$ | WANBIA$^{MSE}$ | RF100 | TAN |
|---|---|---|---|---|---|---|---|---|
| Mush | 0.0002 | 0 | 0 | 0.023 | 0.001 | 0.001 | 0 | 0.0001 |
| Pioneer | 0.0003 | 0.0024 | 0.0028 | 0.0011 | 0 | 0 | 0 | 0.001 |
| Shuttle | 0.0007 | 0.0002 | 0.0003 | 0.003 | 0.0011 | 0.0009 | 0.0006 | 0.0006 |
| Syncon | 0.002 | 0.005 | 0.0054 | 0.0104 | 0.0068 | 0.007 | 0.008 | 0.0055 |
| Hypo | 0.0084 | 0.0029 | 0.0026 | 0.0101 | 0.0061 | 0.0061 | 0.0083 | 0.012 |
| Dermatology | 0.0104 | 0.0135 | 0.0168 | 0.0108 | 0.0071 | 0.0082 | 0.019 | 0.013 |
| Wine | 0.012 | 0.0115 | 0.0132 | 0.0118 | 0.003 | 0.0039 | 0.01 | 0.0102 |
| Anneal | 0.0124 | 0.005 | 0.0042 | 0.0256 | 0.0135 | 0.012 | 0.006 | 0.0137 |
| Pendigits | 0.0133 | 0.0167 | 0.0162 | 0.1081 | 0.0939 | 0.0915 | 0.0216 | 0.0296 |
| Labor | 0.0168 | 0.0318 | 0.039 | 0.0167 | 0.0175 | 0.0167 | 0.0409 | 0.0201 |
| Page-blocks | 0.0231 | 0.0228 | 0.0244 | 0.0525 | 0.0327 | 0.0304 | 0.0217 | 0.0336 |
| Optdigits | 0.0233 | 0.0237 | 0.0232 | 0.0666 | 0.0513 | 0.0503 | 0.0294 | 0.0348 |
| Sick | 0.0237 | 0.0224 | 0.0219 | 0.0284 | 0.0227 | 0.0231 | 0.0194 | 0.023 |
| Segment | 0.0253 | 0.0289 | 0.0284 | 0.0633 | 0.0396 | 0.0391 | 0.0253 | 0.0334 |
| Musk2 | 0.0265 | 0.0101 | 0.0065 | 0.0718 | 0.0299 | 0.0274 | 0.028 | 0.0386 |
| Bcw | 0.0274 | 0.0284 | 0.0301 | 0.0249 | 0.0269 | 0.0275 | 0.0301 | 0.0254 |
| Wall-following | 0.0285 | 0.0059 | 0.004 | 0.0854 | 0.0165 | 0.0156 | 0.0122 | 0.0499 |
| New-thyroid | 0.0303 | 0.0298 | 0.0306 | 0.0295 | 0.0273 | 0.0283 | 0.0285 | 0.0268 |
| Splice-c4.5 | 0.0307 | 0.0362 | 0.0326 | 0.0382 | 0.0316 | 0.0294 | 0.0272 | 0.038 |
| Zoo | 0.0334 | 0.0332 | 0.0321 | 0.0394 | 0.0363 | 0.0384 | 0.0356 | 0.0468 |
| House-votes-84 | 0.0466 | 0.0271 | 0.0262 | 0.0913 | 0.0364 | 0.0358 | 0.0327 | 0.0444 |
| Iris | 0.0466 | 0.0401 | 0.0442 | 0.0503 | 0.0402 | 0.0412 | 0.0398 | 0.0464 |
| Musk1 | 0.0578 | 0.0463 | 0.0431 | 0.1165 | 0.0394 | 0.0399 | 0.0328 | 0.0543 |
| Ionosphere | 0.0579 | 0.057 | 0.056 | 0.0807 | 0.0592 | 0.0542 | 0.0624 | 0.0647 |
| Thyroid | 0.0595 | 0.0434 | 0.0453 | 0.0979 | 0.0944 | 0.0851 | 0.0516 | 0.0634 |
| Spambase | 0.0601 | 0.0452 | 0.0482 | 0.0949 | 0.0551 | 0.0532 | 0.0432 | 0.0588 |
| Car | 0.0605 | 0.0523 | 0.0519 | 0.1076 | 0.1076 | 0.1076 | 0.0389 | 0.0474 |
| Nursery | 0.0656 | 0.0684 | 0.0682 | 0.0904 | 0.0904 | 0.0904 | 0.0086 | 0.0543 |
| Letter-recog | 0.0684 | 0.0967 | 0.1013 | 0.2196 | 0.2134 | 0.2122 | 0.049 | 0.1041 |
| Kr-vs-kp | 0.0716 | 0.0192 | 0.0197 | 0.1067 | 0.0567 | 0.0519 | 0.0063 | 0.0619 |
| Satellite | 0.0831 | 0.0855 | 0.09 | 0.1684 | 0.1428 | 0.1386 | 0.0874 | 0.09 |
| Census-income | 0.0862 | 0.041 | 0.041 | 0.2319 | 0.0454 | 0.0453 | 0.0416 | 0.052 |
| Promoters | 0.0872 | 0.0585 | 0.0613 | 0.0683 | 0.0604 | 0.0599 | 0.0552 | 0.0773 |
| Chess | 0.0943 | 0.0772 | 0.0813 | 0.0989 | 0.0974 | 0.0955 | 0.0548 | 0.0551 |
| Vowel | 0.0985 | 0.1 | 0.1121 | 0.2287 | 0.222 | 0.2224 | 0.0756 | 0.1069 |
| Autos | 0.1164 | 0.1087 | 0.1068 | 0.1791 | 0.1427 | 0.1384 | 0.1111 | 0.1464 |
| Waveform-5000 | 0.1176 | 0.1112 | 0.1134 | 0.1828 | 0.1404 | 0.1403 | 0.1114 | 0.1212 |
| Sonar | 0.1179 | 0.101 | 0.1016 | 0.1314 | 0.1137 | 0.1094 | 0.1045 | 0.1044 |
| Crx | 0.1206 | 0.1103 | 0.1079 | 0.1253 | 0.1108 | 0.1106 | 0.117 | 0.1185 |
| Lyn | 0.1234 | 0.1155 | 0.1183 | 0.1257 | 0.1169 | 0.1132 | 0.1288 | 0.1232 |
| Hepatitis | 0.1242 | 0.1007 | 0.1012 | 0.1294 | 0.0958 | 0.095 | 0.1071 | 0.1112 |
| Glass3 | 0.1378 | 0.1289 | 0.1226 | 0.1597 | 0.1424 | 0.1425 | 0.1348 | 0.1269 |
| Adult | 0.1401 | 0.1207 | 0.1208 | 0.1544 | 0.127 | 0.1267 | 0.1109 | 0.1263 |
| Hungarian | 0.1426 | 0.1291 | 0.1354 | 0.1487 | 0.1373 | 0.1361 | 0.1346 | 0.1166 |
| Phoneme | 0.1465 | 0.1264 | 0.1264 | 0.1965 | 0.1758 | 0.1763 | 0.1102 | 0.1877 |
| Cleveland | 0.1505 | 0.1357 | 0.1438 | 0.1548 | 0.1423 | 0.1406 | 0.1304 | 0.1416 |
| Cylinder-bands | 0.1522 | 0.1464 | 0.1464 | 0.1814 | 0.142 | 0.1456 | 0.208 | 0.2912 |
| Magic | 0.1605 | 0.1446 | 0.1443 | 0.2115 | 0.1656 | 0.1651 | 0.1244 | 0.147 |
| Horse-colic | 0.1619 | 0.1403 | 0.1344 | 0.182 | 0.1275 | 0.1293 | 0.1345 | 0.1452 |
| Balance-scale | 0.1721 | 0.17 | 0.1665 | 0.1633 | 0.1633 | 0.1633 | 0.1731 | 0.1707 |
| Pid | 0.1979 | 0.1895 | 0.1872 | 0.2047 | 0.1873 | 0.1886 | 0.1802 | 0.1816 |
| Ttt | 0.1996 | 0.0171 | 0.0162 | 0.2493 | 0.2354 | 0.2349 | 0.027 | 0.1701 |
| Vehicle | 0.1998 | 0.1765 | 0.1821 | 0.3066 | 0.2463 | 0.2446 | 0.1827 | 0.196 |
| German | 0.1998 | 0.1932 | 0.1953 | 0.2101 | 0.2058 | 0.2053 | 0.197 | 0.1917 |
| Contact-lenses | 0.209 | 0.158 | 0.2094 | 0.2069 | 0.2015 | 0.2042 | 0.1748 | 0.3408 |
| Haberman | 0.2106 | 0.2124 | 0.2206 | 0.2106 | 0.2106 | 0.2106 | 0.2107 | 0.2107 |
| Audio | 0.219 | 0.1478 | 0.1424 | 0.2185 | 0.1758 | 0.1744 | 0.173 | 0.1662 |
| Covtype-mod | 0.2208 | 0.2474 | 0.2474 | 0.3034 | 0.2867 | 0.2858 | | 0.2299 |
| Connect-4 | 0.225 | 0.2346 | 0.2346 | 0.2643 | 0.2628 | 0.2627 | 0.1427 | 0.2226 |
| Led | 0.2278 | 0.2269 | 0.2245 | 0.2262 | 0.2264 | 0.2265 | 0.2278 | 0.2257 |
| Echocardiogram | 0.2447 | 0.2372 | 0.239 | 0.2578 | 0.235 | 0.2329 | 0.2256 | 0.2356 |
| Sign | 0.257 | 0.2927 | 0.2924 | 0.3432 | 0.3411 | 0.3388 | 0.154 | 0.2505 |
| Post-operative | 0.2915 | 0.2715 | 0.2848 | 0.2805 | 0.2792 | 0.2766 | 0.3007 | 0.2685 |
| Localization | 0.3179 | 0.4368 | 0.4368 | 0.4717 | 0.4693 | 0.4698 | 0.2047 | 0.3097 |
| Tae | 0.3305 | 0.3306 | 0.3311 | 0.3649 | 0.3636 | 0.3583 | 0.3315 | 0.3385 |
| Volcanoes | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 | 0.3309 |
| Bupa | 0.3396 | 0.3396 | 0.3146 | 0.3396 | 0.3396 | 0.3396 | 0.3451 | 0.3396 |
| Abalone | 0.3425 | 0.3457 | 0.3406 | 0.4201 | 0.3795 | 0.3785 | 0.3257 | 0.3361 |
| Lung-cancer | 0.3473 | 0.3714 | 0.384 | 0.352 | 0.3495 | 0.3525 | 0.3804 | 0.2834 |
| Yeast | 0.3672 | 0.3651 | 0.3655 | 0.3745 | 0.373 | 0.3722 | 0.336 | 0.3495 |
| Ptn | 0.3826 | 0.3667 | 0.3681 | 0.384 | 0.386 | 0.3791 | 0.3876 | 0.3708 |
| Cmc | 0.3907 | 0.3689 | 0.3677 | 0.4237 | 0.3941 | 0.3908 | 0.3383 | 0.3425 |
| Poker-hand | 0.4423 | 0.4988 | 0.4988 | 0.4988 | 0.4988 | 0.4988 | | 0.2356 |
| Mean | 0.1366 | 0.1293 | 0.1305 | 0.1677 | 0.1486 | 0.1477 | **0.1151** | 0.1334 |
| Mean Rank | 4.7397 | 3.3561 | 3.5958 | 7.0136 | 5.0273 | 4.5342 | **3.3424** | 4.3901 |

Table 22: Bias

| | A1DE | LR | LR-Reg | NB | WANBIA$^{CLL}$ | WANBIA$^{MSE}$ | RF100 | TAN |
|---|---|---|---|---|---|---|---|---|
| Volcanoes | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Mush | 0.0001 | 0 | 0 | 0.0031 | 0.0002 | 0 | 0 | 0.0003 |
| Shuttle | 0.0005 | 0.0002 | 0.0002 | 0.001 | 0.0003 | 0.0003 | 0.0003 | 0.0004 |
| Syncon | 0.0013 | 0.0051 | 0.0054 | 0.0029 | 0.0047 | 0.0048 | 0.0047 | 0.006 |
| Pioneer | 0.0022 | 0.0055 | 0.002 | 0.0024 | 0.0002 | 0.0001 | 0.0007 | 0.0046 |
| Sick | 0.0026 | 0.0045 | 0.0037 | 0.0028 | 0.0036 | 0.0032 | 0.0069 | 0.0041 |
| Hypo | 0.0031 | 0.0033 | 0.0023 | 0.0045 | 0.0021 | 0.0021 | 0.0039 | 0.0067 |
| Wine | 0.0054 | 0.007 | 0.0068 | 0.0023 | 0.0049 | 0.0073 | 0.0111 | 0.0123 |
| Spambase | 0.0061 | 0.0174 | 0.0106 | 0.003 | 0.0075 | 0.0079 | 0.0143 | 0.0101 |
| Anneal | 0.0063 | 0.0067 | 0.0051 | 0.0105 | 0.0064 | 0.007 | 0.0062 | 0.0129 |
| Pendigits | 0.0063 | 0.0247 | 0.0126 | 0.0097 | 0.0099 | 0.0103 | 0.0124 | 0.0114 |
| Page-blocks | 0.0074 | 0.014 | 0.0073 | 0.0083 | 0.0046 | 0.0047 | 0.0092 | 0.0076 |
| Nursery | 0.0089 | 0.0063 | 0.0065 | 0.0074 | 0.0074 | 0.0074 | 0.0162 | 0.0143 |
| House-votes-84 | 0.0089 | 0.0222 | 0.0199 | 0.0063 | 0.0088 | 0.0106 | 0.0089 | 0.0206 |
| Adult | 0.009 | 0.0066 | 0.0066 | 0.0045 | 0.0036 | 0.0036 | 0.0357 | 0.0124 |
| Magic | 0.0091 | 0.0092 | 0.0094 | 0.0054 | 0.0064 | 0.0065 | 0.043 | 0.0149 |
| Optdigits | 0.0096 | 0.0297 | 0.0164 | 0.0097 | 0.0128 | 0.0155 | 0.0165 | 0.0139 |
| Bcw | 0.0097 | 0.0187 | 0.0109 | 0.0017 | 0.0074 | 0.0093 | 0.0085 | 0.0259 |
| Splice-c4.5 | 0.0097 | 0.0331 | 0.0125 | 0.0081 | 0.0095 | 0.0106 | 0.0217 | 0.0234 |
| Dermatology | 0.0109 | 0.0153 | 0.0142 | 0.0093 | 0.0119 | 0.0128 | 0.0178 | 0.0305 |
| Iris | 0.011 | 0.0169 | 0.0131 | 0.0051 | 0.0128 | 0.0131 | 0.0162 | 0.0123 |
| New-thyroid | 0.012 | 0.0216 | 0.0222 | 0.0079 | 0.0124 | 0.0129 | 0.0194 | 0.0246 |
| Ionosphere | 0.0122 | 0.0299 | 0.0258 | 0.0061 | 0.0187 | 0.0205 | 0.0142 | 0.0134 |
| Census-income | 0.0124 | 0.0023 | 0.0023 | 0.0036 | 0.0008 | 0.0008 | 0.0078 | 0.0055 |
| Segment | 0.0135 | 0.0261 | 0.0195 | 0.0119 | 0.0107 | 0.0115 | 0.016 | 0.0194 |
| Thyroid | 0.0157 | 0.0249 | 0.0189 | 0.0137 | 0.0117 | 0.0143 | 0.0234 | 0.0221 |
| Hungarian | 0.0166 | 0.052 | 0.0455 | 0.0098 | 0.0327 | 0.0331 | 0.0528 | 0.0494 |
| Musk2 | 0.0173 | 0.0106 | 0.0078 | 0.0066 | 0.0099 | 0.0126 | 0.0105 | 0.0108 |
| Connect-4 | 0.0189 | 0.0079 | 0.0079 | 0.0149 | 0.0098 | 0.0099 | 0.0448 | 0.0142 |
| Wall-following | 0.0194 | 0.0059 | 0.0045 | 0.0103 | 0.0054 | 0.005 | 0.0094 | 0.0194 |
| Kr-vs-kp | 0.0199 | 0.0085 | 0.009 | 0.02 | 0.0129 | 0.0103 | 0.0065 | 0.0153 |
| Covtype-mod | 0.0205 | 0.0097 | 0.0097 | 0.0082 | 0.0045 | 0.0049 | | 0.0213 |
| Cleveland | 0.0205 | 0.0506 | 0.0327 | 0.0145 | 0.0341 | 0.0384 | 0.0603 | 0.0493 |
| Pid | 0.0214 | 0.0302 | 0.0326 | 0.0168 | 0.0278 | 0.0266 | 0.0734 | 0.0432 |
| Poker-hand | 0.022 | 0 | 0 | 0 | 0 | 0 | | 0.0935 |
| Sign | 0.022 | 0.0273 | 0.028 | 0.0161 | 0.0178 | 0.0179 | 0.0498 | 0.0242 |
| Crx | 0.0221 | 0.0462 | 0.0317 | 0.0196 | 0.0272 | 0.0297 | 0.0411 | 0.0467 |
| Waveform-5000 | 0.0251 | 0.0358 | 0.0253 | 0.009 | 0.0152 | 0.0162 | 0.0443 | 0.0613 |
| Satellite | 0.0261 | 0.0952 | 0.0275 | 0.0067 | 0.0124 | 0.0129 | 0.0211 | 0.0279 |
| Sonar | 0.0261 | 0.0773 | 0.0621 | 0.0206 | 0.055 | 0.0577 | 0.066 | 0.0639 |
| Musk1 | 0.0291 | 0.0414 | 0.0379 | 0.016 | 0.0255 | 0.0267 | 0.0355 | 0.0219 |
| Zoo | 0.0295 | 0.0331 | 0.0362 | 0.0319 | 0.036 | 0.0374 | 0.0387 | 0.0463 |
| Labor | 0.0297 | 0.0419 | 0.0426 | 0.0377 | 0.0369 | 0.0394 | 0.053 | 0.0641 |
| Letter-recog | 0.0341 | 0.0672 | 0.0482 | 0.0367 | 0.035 | 0.0353 | 0.0413 | 0.0469 |
| Glass3 | 0.0346 | 0.0718 | 0.055 | 0.0275 | 0.0333 | 0.0353 | 0.0603 | 0.0576 |
| Led | 0.0372 | 0.0425 | 0.0414 | 0.037 | 0.0369 | 0.0383 | 0.0523 | 0.0445 |
| Chess | 0.0375 | 0.0461 | 0.043 | 0.0374 | 0.0363 | 0.0366 | 0.0526 | 0.0509 |
| Hepatitis | 0.0378 | 0.0915 | 0.0688 | 0.028 | 0.0493 | 0.0541 | 0.0535 | 0.0601 |
| Localization | 0.041 | 0.0217 | 0.0216 | 0.0222 | 0.0209 | 0.0202 | 0.0929 | 0.0467 |
| Yeast | 0.0413 | 0.0413 | 0.0404 | 0.037 | 0.0338 | 0.0362 | 0.0849 | 0.0602 |
| Lyn | 0.0435 | 0.0831 | 0.0675 | 0.0408 | 0.0632 | 0.0665 | 0.0621 | 0.1025 |
| Bupa | 0.0448 | 0.0448 | 0.0821 | 0.0448 | 0.0448 | 0.0448 | 0.0366 | 0.0448 |
| Car | 0.0464 | 0.0219 | 0.0214 | 0.0484 | 0.0484 | 0.0484 | 0.0383 | 0.0336 |
| Horse-colic | 0.0488 | 0.1322 | 0.0453 | 0.0307 | 0.0347 | 0.0366 | 0.0445 | 0.0785 |
| Ttt | 0.0506 | 0.0075 | 0.0018 | 0.0409 | 0.0377 | 0.0365 | 0.0495 | 0.0783 |
| German | 0.0537 | 0.0643 | 0.0573 | 0.0431 | 0.0511 | 0.0517 | 0.0714 | 0.0921 |
| Haberman | 0.0608 | 0.0584 | 0.0503 | 0.0608 | 0.0608 | 0.0608 | 0.0602 | 0.0615 |
| Vehicle | 0.0763 | 0.1079 | 0.0901 | 0.0699 | 0.0826 | 0.0842 | 0.0915 | 0.0803 |
| Autos | 0.0819 | 0.1067 | 0.0991 | 0.0763 | 0.0873 | 0.0918 | 0.0825 | 0.0973 |
| Post-operative | 0.0835 | 0.1535 | 0.0158 | 0.0923 | 0.0819 | 0.0761 | 0.0965 | 0.1021 |
| Cylinder-bands | 0.0848 | 0.0936 | 0.0813 | 0.0745 | 0.117 | 0.1203 | 0.0622 | 0.0849 |
| Echocardiogram | 0.0881 | 0.0983 | 0.0961 | 0.0628 | 0.0948 | 0.0973 | 0.1233 | 0.1121 |
| Cmc | 0.0885 | 0.0781 | 0.0801 | 0.0591 | 0.0754 | 0.0747 | 0.1593 | 0.1225 |
| Balance-scale | 0.0961 | 0.0955 | 0.0962 | 0.0962 | 0.0962 | 0.0962 | 0.0979 | 0.0954 |
| Audio | 0.1097 | 0.1199 | 0.0972 | 0.112 | 0.1162 | 0.1199 | 0.1279 | 0.1699 |
| Promoters | 0.1113 | 0.0656 | 0.0689 | 0.0704 | 0.0755 | 0.0764 | 0.0967 | 0.1552 |
| Abalone | 0.1136 | 0.1166 | 0.125 | 0.0594 | 0.0852 | 0.0858 | 0.1566 | 0.1327 |
| Phoneme | 0.1165 | 0.128 | 0.0804 | 0.107 | 0.0829 | 0.0844 | 0.0687 | 0.1606 |
| Vowel | 0.1214 | 0.1413 | 0.125 | 0.1643 | 0.1397 | 0.1422 | 0.0918 | 0.1712 |
| Contact-lenses | 0.1473 | 0.1378 | 0.151 | 0.1368 | 0.1401 | 0.1417 | 0.169 | 0.0884 |
| Ptn | 0.1557 | 0.2777 | 0.1796 | 0.1548 | 0.1595 | 0.1629 | 0.2124 | 0.2164 |
| Lung-cancer | 0.1808 | 0.1864 | 0.2113 | 0.1683 | 0.1989 | 0.2037 | 0.2196 | 0.2135 |
| Tae | 0.184 | 0.2045 | 0.2024 | 0.1533 | 0.1552 | 0.1513 | 0.2156 | 0.196 |
| Mean | 0.0415 | 0.0525 | 0.0430 | **0.0357** | 0.0400 | 0.0409 | 0.0537 | 0.0556 |
| Mean Rank | 3.8150 | 5.7054 | 4.6712 | **2.8082** | 3.0821 | 3.9520 | 5.6917 | 6.2739 |

Table 23: Variance

| | A1DE | LR | LR-Reg | NB | WANBIA$^{CLL}$ | WANBIA$^{MSE}$ | RF100 | TAN |
|---|---|---|---|---|---|---|---|---|
| Contact-lenses | 0.075 | 0.117 | 13.765 | 0.067 | 0.063 | 0.074 | 0.621 | 0.071 |
| Haberman | 0.098 | 0.114 | 2.67 | 0.088 | 0.08 | 0.077 | 0.655 | 0.096 |
| Tae | 0.098 | 0.895 | 39.861 | 0.086 | 0.111 | 0.131 | 0.965 | 0.097 |
| Echocardiogram | 0.098 | 0.198 | 7.756 | 0.086 | 0.108 | 0.116 | 0.733 | 0.092 |
| New-thyroid | 0.099 | 0.2 | 13.392 | 0.093 | 0.121 | 0.132 | 0.679 | 0.1 |
| Post-operative | 0.099 | 0.294 | 14.513 | 0.083 | 0.111 | 0.106 | 0.762 | 0.101 |
| Bupa | 0.099 | 0.112 | 3.486 | 0.097 | 0.084 | 0.087 | 0.693 | 0.101 |
| Iris | 0.099 | 0.166 | 6.629 | 0.088 | 0.115 | 0.123 | 0.661 | 0.099 |
| Wine | 0.1 | 0.231 | 10.128 | 0.096 | 0.189 | 0.236 | 0.735 | 0.108 |
| Labor | 0.1 | 0.166 | 16.176 | 0.085 | 0.132 | 0.154 | 0.663 | 0.105 |
| Glass3 | 0.1 | 0.556 | 48.909 | 0.089 | 0.159 | 0.178 | 1.018 | 0.108 |
| Zoo | 0.104 | 0.368 | 17.779 | 0.088 | 0.193 | 0.268 | 0.737 | 0.124 |
| Cleveland | 0.106 | 0.662 | 41.424 | 0.105 | 0.2 | 0.234 | 1.258 | 0.119 |
| Hungarian | 0.107 | 0.94 | 67.985 | 0.1 | 0.269 | 0.319 | 1.233 | 0.118 |
| Balance-scale | 0.108 | 0.232 | 10.895 | 0.1 | 0.091 | 0.086 | 1.072 | 0.115 |
| Hepatitis | 0.111 | 0.335 | 17.759 | 0.095 | 0.2 | 0.231 | 0.843 | 0.114 |
| Led | 0.113 | 5.203 | 325.834 | 0.105 | 0.341 | 0.444 | 2.831 | 0.13 |
| House-votes-84 | 0.115 | 0.341 | 16.883 | 0.103 | 0.246 | 0.329 | 1.039 | 0.129 |
| Pid | 0.116 | 0.253 | 14.32 | 0.107 | 0.225 | 0.23 | 2.152 | 0.118 |
| Volcanoes | 0.12 | 2.48 | 109.075 | 0.114 | 0.111 | 0.116 | 1.045 | 0.123 |
| Yeast | 0.121 | 33.068 | 1091.32 | 0.111 | 0.598 | 0.714 | 3.914 | 0.14 |
| Lyn | 0.121 | 0.407 | 24.48 | 0.094 | 0.262 | 0.345 | 0.883 | 0.118 |
| Ttt | 0.121 | 0.577 | 62.348 | 0.113 | 0.202 | 0.203 | 2.26 | 0.127 |
| Horse-colic | 0.124 | 1.581 | 78.819 | 0.108 | 0.376 | 0.385 | 1.373 | 0.147 |
| Crx | 0.125 | 4.24 | 171.602 | 0.108 | 0.668 | 0.637 | 2.097 | 0.145 |
| Car | 0.128 | 6.549 | 469.286 | 0.117 | 0.123 | 0.127 | 2.556 | 0.137 |
| Cmc | 0.129 | 6.639 | 243.93 | 0.117 | 0.535 | 0.579 | 5.766 | 0.139 |
| Ptn | 0.131 | 181.979 | 3127.046 | 0.094 | 0.965 | 1.278 | 3.086 | 0.144 |
| Bcw | 0.132 | 0.423 | 20.418 | 0.106 | 0.312 | 0.398 | 1.059 | 0.138 |
| Vowel | 0.137 | 18.835 | 1901.16 | 0.114 | 0.921 | 1.313 | 3.79 | 0.164 |
| Vehicle | 0.14 | 37.456 | 1051.993 | 0.115 | 0.729 | 0.962 | 3.98 | 0.167 |
| German | 0.145 | 3.33 | 326.224 | 0.12 | 0.556 | 0.559 | 3.468 | 0.168 |
| Autos | 0.156 | 1.821 | 212.885 | 0.096 | 0.675 | 0.93 | 1.22 | 0.175 |
| Ionosphere | 0.16 | 0.611 | 36.326 | 0.113 | 0.363 | 1.045 | 1.151 | 0.218 |
| Chess | 0.161 | 5.658 | 308.214 | 0.123 | 0.65 | 0.756 | 3.069 | 0.212 |
| Sonar | 0.162 | 1.51 | 150.955 | 0.12 | 0.356 | 0.429 | 1.794 | 0.227 |
| Abalone | 0.168 | 16.021 | 883.628 | 0.146 | 1.953 | 1.972 | 14.309 | 0.195 |
| Dermatology | 0.183 | 1.792 | 168.475 | 0.111 | 1.84 | 3.228 | 1.403 | 0.243 |
| Promoters | 0.192 | 0.616 | 24.846 | 0.106 | 0.395 | 0.411 | 0.86 | 0.31 |
| Lung-cancer | 0.192 | 1.035 | 65.012 | 0.098 | 0.311 | 0.246 | 0.772 | 0.214 |
| Page-blocks | 0.22 | 77.192 | 3062.031 | 0.166 | 3.405 | 3.923 | 6.038 | 0.233 |
| Anneal | 0.235 | 4.306 | 555.919 | 0.124 | 7.262 | 10.031 | 2.345 | 0.294 |
| Segment | 0.256 | 11.027 | 1455.738 | 0.136 | 4.472 | 5.516 | 4.545 | 0.347 |
| Nursery | 0.268 | 43.87 | 2885.603 | 0.237 | 0.476 | 0.507 | 17.253 | 0.307 |
| Sign | 0.271 | 97.373 | 3309.138 | 0.257 | 2.234 | 3.712 | 29.188 | 0.308 |
| Sick | 0.274 | 18.226 | 1497.482 | 0.195 | 13.786 | 15.394 | 7.901 | 0.412 |
| Hypo | 0.293 | 7.452 | 809.711 | 0.193 | 15.659 | 21.443 | 5.492 | 0.415 |
| Kr-vs-kp | 0.303 | 27.203 | 1483.335 | 0.195 | 6.525 | 8.376 | 9.458 | 0.533 |
| Mush | 0.349 | 4.495 | 587.965 | 0.235 | 3.97 | 7.46 | 5.825 | 0.506 |
| Audio | 0.379 | 31.293 | 6576.503 | 0.125 | 4.846 | 8.245 | 3.003 | 0.616 |
| Magic | 0.408 | 84.379 | 4453.728 | 0.32 | 4.507 | 4.668 | 56.51 | 0.48 |
| Pendigits | 0.462 | 37.11 | 4665.818 | 0.248 | 13.162 | 19.172 | 25.167 | 0.735 |
| Syncon | 0.466 | 6.798 | 816.926 | 0.135 | 5.481 | 6.478 | 2.129 | 0.769 |
| Waveform-5000 | 0.474 | 107.506 | 7163.793 | 0.237 | 5.051 | 6.559 | 19.181 | 0.917 |
| Thyroid | 0.555 | 1839.199 | 42695.323 | 0.273 | 187.224 | 272.806 | 35.287 | 0.915 |
| Phoneme | 0.565 | 2430.18 | 78868.508 | 0.15 | 12.683 | 15.977 | 36.345 | 0.736 |
| Cylinder-bands | 0.623 | 8.705 | 1006.751 | 0.119 | 1.788 | 1.828 | 3.009 | 0.586 |
| Splice-c4.5 | 0.664 | 7.369 | 963.412 | 0.229 | 9.889 | 14.683 | 11.431 | 1.376 |
| Spambase | 0.665 | 106.084 | 6027.538 | 0.256 | 12.773 | 17.797 | 17.634 | 1.417 |
| Musk1 | 0.707 | 3.631 | 837.027 | 0.188 | 11.641 | 23.526 | 4.597 | 1.508 |
| Letter-recog | 0.788 | 5570.466 | 892.459 | 0.363 | 73.676 | 96.965 | 87.782 | 1.336 |
| Wall-following | 0.876 | 13.34 | 2915.697 | 0.198 | 4.933 | 6.254 | 12.4 | 1.276 |
| Shuttle | 0.993 | 61.266 | 5237.291 | 0.756 | 37.254 | 47.905 | 63.185 | 1.291 |
| Satellite | 1.02 | 1534.966 | 9646.135 | 0.255 | 16.169 | 26.975 | 18.788 | 2.135 |
| Adult | 1.025 | 720.394 | 31035.38 | 0.74 | 33.376 | 41.734 | 199.569 | 1.465 |
| Optdigits | 1.375 | 50.882 | 6635.504 | 0.303 | 116.538 | 152.671 | 28.835 | 3.203 |
| Localization | 1.977 | 6113.835 | 226842.296 | 1.834 | 92.014 | 98.267 | 938.448 | 2.259 |
| Connect-4 | 4.695 | 2218.933 | 95760.528 | 1.748 | 91.768 | 108.129 | 691.22 | 10.542 |
| Covtype-mod | 10.856 | 25589.054 | 519389.101 | 7.295 | 683.568 | 843.389 | | 15.292 |
| Poker-hand | 15.15 | 32887.718 | 751710.215 | 12.815 | 59.807 | 66.984 | | 18.964 |
| Musk2 | 19.881 | 56.031 | 19571.881 | 0.583 | 54.759 | 198.24 | 27.884 | 36.344 |
| Census-income | 20.808 | 11189.489 | 438360.993 | 7.233 | 1082.365 | 1129.399 | 2307.386 | 50.237 |
| Pioneer | 309.15 | 835.65 | 8835.12 | 0.328 | 658.631 | 564.602 | 296.181 | 671.052 |
| Mean | 5.50 | 1262.10 | 14865.08 | **0.58** | 45.85 | 53.01 | 71.17 | 11.43 |
| Mean Rank | 2.2941 | 6.4109 | 7.9726 | **1.1369** | 4.0958 | 5.1095 | 5.8356 | 3.1438 |

Table 24: Train time

| | A1DE | LR | LR-Reg | NB | WANBIA$^{CLL}$ | WANBIA$^{MSE}$ | RF100 | TAN |
|---|---|---|---|---|---|---|---|---|
| Contact-lenses | 0.006 | 0.0015 | 0.003 | 0.001 | 0.003 | 0.002 | 0.005 | 0.005 |
| Post-operative | 0.017 | 0.0075 | 0.009 | 0.005 | 0.004 | 0.009 | 0.011 | 0.015 |
| Labor | 0.018 | 0.0045 | 0.006 | 0.007 | 0.005 | 0.005 | 0.006 | 0.01 |
| Iris | 0.02 | 0.0045 | 0.015 | 0.01 | 0.007 | 0.006 | 0.022 | 0.016 |
| Tae | 0.021 | 0.0075 | 0.0105 | 0.008 | 0.009 | 0.006 | 0.021 | 0.021 |
| New-thyroid | 0.023 | 0.003 | 0.0135 | 0.006 | 0.007 | 0.009 | 0.023 | 0.018 |
| Echocardiogram | 0.024 | 0.0045 | 0.015 | 0.006 | 0.006 | 0.004 | 0.022 | 0.013 |
| Bupa | 0.028 | 0.0075 | 0.024 | 0.007 | 0.01 | 0.009 | 0.018 | 0.018 |
| Hepatitis | 0.029 | 0.006 | 0.012 | 0.009 | 0.009 | 0.007 | 0.022 | 0.019 |
| Glass3 | 0.029 | 0.009 | 0.021 | 0.009 | 0.008 | 0.006 | 0.039 | 0.023 |
| Haberman | 0.029 | 0.006 | 0.0225 | 0.008 | 0.008 | 0.011 | 0.013 | 0.019 |
| Wine | 0.031 | 0.006 | 0.018 | 0.008 | 0.006 | 0.008 | 0.019 | 0.018 |
| Zoo | 0.033 | 0.0135 | 0.0105 | 0.005 | 0.004 | 0.006 | 0.016 | 0.014 |
| Lyn | 0.033 | 0.0105 | 0.015 | 0.009 | 0.007 | 0.008 | 0.022 | 0.018 |
| Cleveland | 0.036 | 0.012 | 0.021 | 0.012 | 0.006 | 0.008 | 0.05 | 0.023 |
| Balance-scale | 0.036 | 0.012 | 0.045 | 0.01 | 0.012 | 0.018 | 0.087 | 0.022 |
| Pid | 0.039 | 0.009 | 0.0285 | 0.013 | 0.009 | 0.011 | 0.145 | 0.023 |
| Hungarian | 0.039 | 0.006 | 0.033 | 0.009 | 0.008 | 0.009 | 0.053 | 0.022 |
| Lung-cancer | 0.041 | 0 | 0.0015 | 0.002 | 0.003 | 0.005 | 0.005 | 0.009 |
| Bcw | 0.045 | 0.0105 | 0.0225 | 0.013 | 0.012 | 0.007 | 0.06 | 0.025 |
| Volcanoes | 0.048 | 0.018 | 0.0495 | 0.023 | 0.02 | 0.02 | 0.08 | 0.033 |
| Tt | 0.051 | 0.015 | 0.0675 | 0.014 | 0.014 | 0.012 | 0.183 | 0.027 |
| House-votes-84 | 0.052 | 0.012 | 0.015 | 0.009 | 0.011 | 0.01 | 0.055 | 0.021 |
| Autos | 0.053 | 0.0105 | 0.018 | 0.015 | 0.009 | 0.006 | 0.036 | 0.018 |
| Car | 0.053 | 0.024 | 0.057 | 0.019 | 0.022 | 0.022 | 0.264 | 0.033 |
| Promoters | 0.057 | 0.0045 | 0.012 | 0.008 | 0.005 | 0.003 | 0.013 | 0.013 |
| Horse-colic | 0.057 | 0.012 | 0.018 | 0.009 | 0.012 | 0.012 | 0.058 | 0.024 |
| Crx | 0.058 | 0.0165 | 0.0375 | 0.014 | 0.009 | 0.01 | 0.12 | 0.026 |
| Ptn | 0.06 | 0.0165 | 0.201 | 0.02 | 0.011 | 0.012 | 0.135 | 0.035 |
| Cmc | 0.061 | 0.0135 | 0.0405 | 0.017 | 0.012 | 0.01 | 0.471 | 0.035 |
| Led | 0.062 | 0.018 | 0.054 | 0.017 | 0.014 | 0.016 | 0.29 | 0.036 |
| Vowel | 0.064 | 0.027 | 0.1125 | 0.022 | 0.019 | 0.022 | 0.298 | 0.045 |
| Yeast | 0.068 | 0.03 | 0.081 | 0.027 | 0.02 | 0.02 | 0.391 | 0.042 |
| Ionosphere | 0.073 | 0.0105 | 0.015 | 0.007 | 0.01 | 0.005 | 0.046 | 0.021 |
| Dermatology | 0.075 | 0.0105 | 0.036 | 0.017 | 0.01 | 0.01 | 0.059 | 0.026 |
| Vehicle | 0.075 | 0.009 | 0.1965 | 0.02 | 0.016 | 0.011 | 0.21 | 0.036 |
| Sonar | 0.079 | 0.003 | 0.0255 | 0.009 | 0.007 | 0.008 | 0.044 | 0.026 |
| German | 0.085 | 0.0105 | 0.093 | 0.018 | 0.011 | 0.015 | 0.218 | 0.029 |
| Abalone | 0.091 | 0.0345 | 0.1155 | 0.026 | 0.026 | 0.023 | 1.793 | 0.046 |
| Chess | 0.101 | 0.0135 | 0.045 | 0.014 | 0.011 | 0.011 | 0.165 | 0.026 |
| Cylinder-bands | 0.103 | 0.0105 | 0.033 | 0.017 | 0.012 | 0.011 | 0.095 | 0.028 |
| Page-blocks | 0.165 | 0.066 | 0.2535 | 0.046 | 0.041 | 0.04 | 0.868 | 0.081 |
| Segment | 0.192 | 0.039 | 0.1545 | 0.035 | 0.034 | 0.034 | 0.446 | 0.07 |
| Anneal | 0.193 | 0.0225 | 0.0645 | 0.024 | 0.023 | 0.021 | 0.147 | 0.047 |
| Sign | 0.256 | 0.105 | 0.2835 | 0.07 | 0.065 | 0.071 | 5.943 | 0.114 |
| Nursery | 0.291 | 0.138 | 0.4845 | 0.099 | 0.094 | 0.095 | 3.54 | 0.159 |
| Sick | 0.307 | 0.048 | 0.21 | 0.038 | 0.036 | 0.036 | 0.943 | 0.072 |
| Audio | 0.343 | 0.024 | 0.114 | 0.024 | 0.023 | 0.024 | 0.087 | 0.068 |
| Syncon | 0.351 | 0.0195 | 0.0855 | 0.023 | 0.02 | 0.02 | 0.092 | 0.046 |
| Kr-vs-kp | 0.359 | 0.042 | 0.1365 | 0.036 | 0.037 | 0.037 | 1.02 | 0.066 |
| Hypo | 0.395 | 0.0615 | 0.1905 | 0.05 | 0.045 | 0.05 | 0.603 | 0.099 |
| Magic | 0.409 | 0.1455 | 0.4125 | 0.096 | 0.092 | 0.088 | 10.413 | 0.161 |
| Mush | 0.439 | 0.093 | 0.261 | 0.063 | 0.061 | 0.065 | 0.788 | 0.109 |
| Phoneme | 0.465 | 0.3885 | 1.2345 | 0.259 | 0.24 | 0.245 | 3.489 | 0.479 |
| Wall-following | 0.716 | 0.081 | 0.2115 | 0.061 | 0.063 | 0.064 | 1.179 | 0.13 |
| Pendigits | 0.808 | 0.231 | 0.6675 | 0.178 | 0.173 | 0.173 | 4.234 | 0.352 |
| Waveform-5000 | 0.848 | 0.078 | 0.291 | 0.073 | 0.064 | 0.064 | 1.834 | 0.133 |
| Musk1 | 0.929 | 0.0165 | 0.0675 | 0.023 | 0.021 | 0.021 | 0.104 | 0.049 |
| Splice-c4.5 | 1.088 | 0.066 | 0.192 | 0.058 | 0.057 | 0.059 | 0.924 | 0.122 |
| Spambase | 1.223 | 0.0825 | 0.2685 | 0.073 | 0.068 | 0.066 | 1.727 | 0.123 |
| Adult | 1.458 | 0.4635 | 1.3635 | 0.318 | 0.295 | 0.301 | 22.379 | 0.522 |
| Shuttle | 1.709 | 0.831 | 2.4285 | 0.561 | 0.55 | 0.558 | 7.409 | 0.965 |
| Satellite | 1.989 | 0.114 | 0.39 | 0.123 | 0.122 | 0.122 | 1.901 | 0.273 |
| Thyroid | 2.261 | 0.372 | 0.9375 | 0.359 | 0.361 | 0.357 | 5.804 | 0.853 |
| Letter-recog | 2.664 | 0.906 | 0.918 | 0.697 | 0.702 | 0.699 | 11.936 | 1.484 |
| Localization | 3.515 | 3.1875 | 8.5485 | 1.947 | 1.879 | 1.895 | 93.887 | 2.782 |
| Optdigits | 5.117 | 0.2145 | 0.657 | 0.24 | 0.232 | 0.235 | 2.463 | 0.598 |
| Connect-4 | 12.134 | 1.2315 | 2.4525 | 0.972 | 0.965 | 0.978 | 47.681 | 1.849 |
| Pioneer | 23.738 | 2.8455 | 7.3455 | 1.649 | 1.453 | 1.462 | 7.09 | 16.145 |
| Covtype-mod | 25.162 | 10.251 | 10.251 | 7.245 | 7.112 | 7.196 | | 12.344 |
| Musk2 | 33.621 | 0.2325 | 0.714 | 0.24 | 0.245 | 0.245 | 1.193 | 0.549 |
| Poker-hand | 39.21 | 21.9015 | 21.9015 | 15.605 | 15.603 | 15.118 | | 24.526 |
| Census-income | 50.673 | 4.839 | 10.836 | 3.92 | 3.933 | 3.845 | 131.887 | 6.667 |
| Mean | 2.945 | 0.679 | 1.041 | 0.489 | **0.481** | 0.475 | 5.319 | 1.000 |
| Mean Rank | 7.1438 | 2.9794 | 5.9383 | 2.9315 | **2.1027** | 2.2191 | 7.2465 | 5.4383 |

Table 25: Test time

B. Cestnik, I. Kononenko, and I. Bratko. Assistant 86: A knowledge-elicitation tool for sophisticated users. In *Proceedings of the Second European Working Session on Learning*, pages 31–45. Wulmslow, UK: Sigma Press, 1987.

B. Damien and G. I. Webb. The need for low bias algorithms in classification learning from small data sets. In *Proceedings of the Sixth European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD) 2002*, pages 62–73, Berlin, 2002. Springer-Verlag.

M. H. DeGroot and S. E. Fienbert. The comparison and evaluation of forecasters. *Statistician*, 32 (1):12–22, 1982.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

P. Domingos and M. J. Pazzani. Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 105–112. Morgan Kaufmann, 1996.

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley and Sons, 2006.

U. M. Fayyad and B. I. Keki. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8(1):87–102, 1992.

J. T. A. S. Ferreira, D. G. T. Denison, and D. J. Hand. Weighted naive Bayes modelling for data mining, 2001.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

J. Friedman. Flexible metric nearest neighbor classification. Technical report, Department of Statistics, Stanford University, 1994.

M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *The American Statistical Association*, 32(200):675–701, 1937.

M. Friedman. A comparison of alternative tests of significance for the problem of *m* rankings. *The American Statistical Association*, 11(1):86–92, 1940.

N. Friedman and M. Goldszmidt. Building classifiers using Bayesian networks. In *AAAI-96*, pages 1277–1284, 1996.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29 (2):131–163, 1997.

R. Greiner, W. Zhou, X. Su, and B. Shen. Structural extensions to logistic regression: Discriminative parameter learning of belief net classifiers. *Journal of Machine Learning Research*, 2004.

D. Grossman and P. Domingos. Learning Bayesian network classifiers by maximizing conditional likelihood. In *International Conference on Machine Learning*, 2004.

I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh. *Feature Extraction, Foundation and Applications*. Springer, 2004.

M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 359–366. Morgan Kaufmann, 2000.

M. A. Hall. A decision tree-based attribute weighting filter for naive Bayes. *Knowledge-Based Systems*, 20:120–126, March 2007.

D. J. Hand and K. Yu. Idiot's Bayes - not so stupid after all? *International Statistical Review*, 69 (3):385–398, 2001.

J. Hilden and B. Bjerregaard. Computer-aided diagnosis and the atypical case. In *In Decision Making and Medical Care: Can Information Science Help*, pages 365–378. North-Holland Publishing Company, 1976.

R. Kohavi and D. Wolpert. Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 275–283. San Francisco: Morgan Kaufmann, 1996.

I. Kononenko. Comparison of inductive and naive Bayesian learning approaches to automatic knowledge acquisition. *Current Trends in Knowledge Acquisition*, 1990.

P. Langley and S. Sage. Induction of selective Bayesian classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 399–406. Morgan Kaufmann, 1994.

P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 223–228. AAAI Press and MIT Press, 1992.

D. D. Lewis. Naive Bayes at forty: The independence assumption in information retrieval. In *ECML-98: Proceedings of the Tenth European Conference on Machine Learning*, pages 4–15, Berlin, April 1998. Springer.

A. Murphy and R. Winkler. Reliability of subjective probability forecasts of precipitation and temperature. *Applied Statistics*, 26(1):41–47, 1977.

F. Pernkopf and J. A. Bilmes. Efficient heuristics for discriminative structure learning of Bayesian network classifiers. *Journal of Machine Learning Research*, 2010.

C. A. Ratanamahatana and D. Gunopulos. Feature selection for the naive Bayesian classifier using decision trees. *Applied Artificial Intelligence*, pages 475–487, 2003.

T. Roos, H. Wettig, P. Grünwald, P. Myllymäki, and H. Tirri. On discriminative Bayesian network classifiers and logistic regression. *Machine Learning*, 59(3):267–296, 2005.

G. I. Webb. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, 40(2):159–196, 2000.

G. I. Webb, J. Boughton, and Z. Wang. Not so naive Bayes: Averaged one-dependence estimators. *Machine Learning*, 58(1):5–24, 2005.

G. I. Webb, J. Boughton, F. Zheng, K. M. Ting, and H. Salem. Learning by extrapolation from marginal to full-multivariate probability distributions: decreasingly naive Bayesian classification. *Machine Learning*, pages 1–40, 2011.

J. Wu and Z. Cai. Attribute weighting via differential evolution algorithm for attribute weighted naive Bayes (wnb). *Journal of Computational Information Systems*, 7(5):1672–1679, 2011.

B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *ACM SIGKDD, International Conference on Knowledge Discovery and Data Mining*, 2002.

H. Zhang and S. Sheng. Learning weighted naive Bayes with accurate ranking. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM -04)*, pages 567–570, 2004.

F. Zheng, G. I. Webb, P. Suraweera, and L. Zhu. Subsumption resolution: An efficient and effective technique for semi-naive Bayesian learning. *Machine Learning*, 87(1):93–125, 2012.

Z. Zheng and G.I. Webb. Lazy learning of Bayesian rules. *Machine Learning*, 41(1):53–84, 2000.

Z. Zheng, G. I. Webb, and K.M. Ting. Lazy Bayesian Rules: A lazy semi-naive Bayesian learning technique competitive to boosting decision trees. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML-99)*, pages 493–502. Morgan Kaufmann, 1999.

C Zhu, R. H. Byrd, and J. Nocedal. L-bfgs-b: Algorithm 778: L-bfgs-b, fortran routines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software*, 23(4):550–560, 1997.

Journal of Machine Learning Research 14 (2013) 1989-2028 Submitted 12/11; Revised 8/12; Published 7/13

# Machine Learning with Operational Costs

**Theja Tulabandhula** THEJA@MIT.EDU
*Department of Electrical Engineering and Computer Science*
*Massachusetts Institute of Technology*
*Cambridge, MA 02139, USA*

**Cynthia Rudin** RUDIN@MIT.EDU
*MIT Sloan School of Management and Operations Research Center*
*Massachusetts Institute of Technology*
*Cambridge, MA 02139, USA*

**Editor:** John-Shawe Taylor

## Abstract

This work proposes a way to align statistical modeling with decision making. We provide a method that propagates the uncertainty in predictive modeling to the uncertainty in operational cost, where operational cost is the amount spent by the practitioner in solving the problem. The method allows us to explore the range of operational costs associated with the set of reasonable statistical models, so as to provide a useful way for practitioners to understand uncertainty. To do this, the operational cost is cast as a regularization term in a learning algorithm's objective function, allowing either an optimistic or pessimistic view of possible costs, depending on the regularization parameter. From another perspective, if we have prior knowledge about the operational cost, for instance that it should be low, this knowledge can help to restrict the hypothesis space, and can help with generalization. We provide a theoretical generalization bound for this scenario. We also show that learning with operational costs is related to robust optimization.

**Keywords:** statistical learning theory, optimization, covering numbers, decision theory

## 1. Introduction

Machine learning algorithms are used to produce predictions, and these predictions are often used to make a policy or plan of action afterwards, where there is a cost to implement the policy. In this work, we would like to understand how the uncertainty in predictive modeling can translate into the uncertainty in the cost for implementing the policy. This would help us answer questions like:

Q1. "What is a reasonable amount to allocate for this task so we can react best to whatever nature brings?"

Q2. "Can we produce a reasonable probabilistic model, supported by data, where we might expect to pay a specific amount?"

Q3. "Can our intuition about how much it will cost to solve a problem help us produce a better probabilistic model?"

The three questions above cannot be answered by standard decision theory, where the goal is to produce a single policy that minimizes expected cost. These questions also cannot be answered by

©2013 Theja Tulabandhula and Cynthia Rudin.

robust optimization, where the goal is to produce a single policy that is robust to the uncertainty in nature. Those paradigms produce a single policy decision that takes uncertainty into account, and the chosen policy might not be a best response policy to any realistic situation. In contrast, our goal is to understand the uncertainty and how to react to it, using policies that would be best responses to individual situations.

There are many applications in which this method can be used. For example, in scheduling staff for a medical clinic, predictions based on a statistical model of the number of patients might be used to understand the possible policies and costs for staffing. In traffic flow problems, predictions based on a model of the forecasted traffic might be useful for determining load balancing policies on the network and their associated costs. In online advertising, predictions based on models for the payoff and ad-click rate might be used to understand policies for when the ad should be displayed and the associated revenue.

In order to propagate the uncertainty in modeling to the uncertainty in costs, we introduce what we call the *simultaneous process*, where we explore the range of predictive models and corresponding policy decisions at the same time. The simultaneous process was named to contrast with a more traditional *sequential process*, where first, data are input into a statistical algorithm to produce a predictive model, which makes recommendations for the future, and second, the user develops a plan of action and projected cost for implementing the policy. The sequential process is commonly used in practice, even though there may actually be a whole class of models that could be relevant for the policy decision problem. The sequential process essentially assumes that the probabilistic model is "correct enough" to make a decision that is "close enough."

In the simultaneous process, the machine learning algorithm contains a regularization term encoding the policy and its associated cost, with an adjustable regularization parameter. If there is some uncertainty about how much it will cost to solve the problem, the regularization parameter can be swept through an interval to find a range of possible costs, from optimistic to pessimistic. The method then produces the most likely scenario for each value of the cost. This way, by looking at the full range of the regularization parameter, we sweep out costs for all of the reasonable probabilistic models. This range can be used to determine how much might be reasonably allocated to solve the problem.

Having the full range of costs for reasonable models can directly answer the question in the first paragraph regarding allocation, "What is a reasonable amount to allocate for this task so we can react best to whatever nature brings?" One might choose to allocate the maximum cost for the set of reasonable predictive models for instance. The second question above is "Can we produce a reasonable probabilistic model, supported by data, where we might expect to pay a specific amount?" This is an important question, since business managers often like to know if there is some scenario/decision pair that is supported by the data, but for which the operational cost is low (or high); the simultaneous process would be able to find such scenarios directly. To do this, we would look at the setting of the regularization parameter that resulted in the desired value of the cost, and then look at the solution of the simultaneous formulation, which gives the model and its corresponding policy decision.

Let us consider the third question above, which is "Can our intuition about how much it will cost to solve a problem help us produce a better probabilistic model?" The regularization parameter can be interpreted to regulate the strength of our belief in the operational cost. If we have a strong belief in the cost to solve the problem, and if that belief is correct, this will guide the choice of regularization parameter, and will help with prediction. In many real scenarios, a practitioner or

domain expert might truly have a prior belief on the cost to complete a task. Arguably, a manager having this more grounded type of prior belief is much more natural than, for instance, the manager having a prior belief on the $\ell_2$ norm of the coefficients of a linear model, or the number of nonzero coefficients in the model. Being able to encode this type of prior belief on cost could potentially be helpful for prediction: as with other types of prior beliefs, it can help to restrict the hypothesis space and can assist with generalization. In this work, we show that the restricted hypothesis spaces resulting from our method can often be bounded by an intersection of an an $\ell_q$ ball with a halfspace - and this is true for many different types of decision problems. We analyze the complexity of this type of hypothesis space with a technique based on Maurey's Lemma (Barron, 1993; Zhang, 2002) that leads eventually to a counting problem, where we calculate the number of integer points within a polyhedron in order to obtain a covering number bound.

The operational cost regularization term can be the optimal value of a complicated optimization problem, like a scheduling problem. This means we will need to solve an optimization problem each time we evaluate the learning algorithm's objective. However, the practitioner must be able to solve that problem anyway in order to develop a plan of action; it is the same problem they need to solve in the traditional sequential process, or using standard decision theory. Since the decision problem is solved only on data from the present, whose labels are not yet known, solving the decision problem may not be difficult, especially if the number of unlabeled examples is small. In that case, the method can still scale up to huge historical data sets, since the historical data factors into the training error term but not the new regularization term, and both terms can be computed. An example is to compute a schedule for a day, based on factors of the various meetings on the schedule that day. We can use a very large amount of past meeting-length data for the training error term, but then we use only the small set of possible meetings coming up that day to pass into the scheduling problem. In that case, both the training error term and the regularization term are able to be computed, and the objective can be minimized.

The simultaneous process is a type of decision theory. To give some background, there are two types of relevant decision theories: normative (which assumes full information, rationality and infinite computational power) and descriptive (models realistic human behavior). Normative decision theories that address decision making under uncertainty can be classified into those based on ignorance (using no probabilistic information) and those based on risk (using probabilistic information). The former include maximax, maximin (Wald), minimax regret (Savage), criterion of realism (Hurwicz), equally likely (Laplace) approaches. The latter include utility based expected value and bayesian approaches (Savage). Info-gap, Dempster-Shafer, fuzzy logic, and possibility theories offer non-probabilistic alternatives to probability in Bayesian/expected value theories (French, 1986; Hansson, 1994).

The simultaneous process does not fit into any of the decision theories listed above. For instance, a core idea in the Bayesian approach is to choose a single policy that maximizes expected utility, or minimizes expected cost. Our goal is not to find a single policy that is useful on average. In contrast, our goal is to trace out a path of models, their specific (not average) optimal-response policies, and their costs. The policy from the Bayesian approach may not correspond to the best decision for any particular single model, whereas that is something we want in our case. We trace out this path by changing our prior belief on the operational cost (that is, by changing the strength of our regularization term). In Bayesian decision theory, the prior is over possible probabilistic models, rather than on possible costs as in this paper. Constructing this prior over possible probabilistic models can be challenging, and the prior often ends up being chosen arbitrarily, or as a matter of

convenience. In contrast, we assume only an unknown probability measure over the data, and the data itself defines the possible probabilistic models for which we compute policies.

Maximax (optimistic) and maximin (pessimistic) decision approaches contrast with the Bayesian framework and do not assume a distribution on the possible probabilistic models. In Section 4 we will discuss how these approaches are related to the simultaneous process. They overlap with the simultaneous process but not completely. Robust optimization is a maximin approach to decision making, and the simultaneous process also differs in principle from robust optimization. In robust optimization, one would generally need to allocate much more than is necessary for any single realistic situation, in order to produce a policy that is robust to almost all situations. However, this is not always true; in fact, we show in this work that in some circumstances, while sweeping through the regularization parameter, one of the results produced by the simultaneous process is the same as the one coming from robust optimization.

We introduce the sequential and simultaneous processes in Section 2. In Section 3, we give several examples of algorithms that incorporate these operational costs. In doing so, we provide answers for the first two questions Q1 and Q2 above, with respect to specific problems.

Our first example application is a staffing problem at a medical clinic, where the decision problem is to staff a set of stations that patients must complete in a certain order. The time required for patients to complete each station is random and estimated from past data. The second example is a real-estate purchasing problem, where the policy decision is to purchase a subset of available properties. The values of the properties need to be estimated from comparable sales. The third example is a call center staffing problem, where we need to create a staffing policy based on historical call arrival and service time information. A fourth example is the "Machine Learning and Traveling Repairman Problem" (ML&TRP) where the policy decision is a route for a repair crew. As mentioned above, there is a large subset of problems that can be formulated using the simultaneous process that have a special property: they are equivalent to robust optimization (RO) problems. Section 4 discusses this relationship and provides, under specific conditions, the equivalence of the simultaneous process with RO. Robust optimization, when used for decision-making, does not usually include machine learning, nor any other type of statistical model, so we discuss how a statistical model can be incorporated within an uncertainty set for an RO. Specifically, we discuss how different loss functions from machine learning correspond to different uncertainty sets. We also discuss the overlap between RO and the optimistic and pessimistic versions of the simultaneous process.

We consider the implications of the simultaneous process on statistical learning theory in Section 5. In particular, we aim to understand how operational costs affect prediction (generalization) ability. This helps answer the third question Q3, about how intuition about operational cost can help produce a better probabilistic model.

We show first that the hypothesis spaces for most of the applications in Section 3 can be bounded in a specific way - by an intersection of a ball and a halfspace - and this is true regardless of how complicated the constraints of the optimization problem are, and how different the operational costs are from each other in the different applications. Second, we bound the complexity of this type of hypothesis space using a technique based on Maurey's Lemma (Barron, 1993; Zhang, 2002) that leads eventually to a counting problem, where we calculate the number of integer points within a polyhedron in order to obtain a generalization bound. Our results show that it is possible to make use of much more general structure in estimation problems, compared to the standard (norm-constrained) structures like sparsity and smoothness; further, this additional structure can benefit generalization

ability. A shorter version of this work has been previously published (see Tulabandhula and Rudin, 2012).

## 2. The Sequential and Simultaneous Processes

We have a training set of (random) labeled instances, $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$ that we will use to learn a function $f^* : \mathcal{X} \to \mathcal{Y}$. Commonly in machine learning this is done by choosing $f$ to be the solution of a minimization problem:

$$f^* \in \text{argmin}_{f \in \mathcal{F}^{unc}} \left( \sum_{i=1}^{n} l(f(x_i), y_i) + C_2 R(f) \right), \tag{1}$$

for some loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$, regularizer $R : \mathcal{F}^{unc} \to \mathbb{R}$, constant $C_2$ and function class $\mathcal{F}^{unc}$. Here, $\mathcal{Y} \subset \mathbb{R}$. Typical loss functions used in machine learning are the 0-1 loss, ramp loss, hinge loss, logistic loss and the exponential loss. Function class $\mathcal{F}^{unc}$ is commonly the class of all linear functionals, where an element $f \in \mathcal{F}^{unc}$ is of the form $\beta^T x$, where $\mathcal{X} \subset \mathbb{R}^p$, $\beta \in \mathbb{R}^p$. We have used '*unc*' in the superscript for $\mathcal{F}^{unc}$ to refer to the word "unconstrained," since it contains all linear functionals. Typical regularizers $R$ are the $\ell_1$ and $\ell_2$ norms of $\beta$. Note that nonlinearities can be incorporated into $\mathcal{F}^{unc}$ by allowing nonlinear features, so that we now would have $f(x) = \sum_{j=1}^{p} \beta_j h_j(x)$, where $\{h_j\}_j$ is the set of features, which can be arbitrary nonlinear functions of $x$; for simplicity in notation, we will equate $h_j(x) = x_j$ and have $\mathcal{X} \subset \mathbb{R}^p$.

Consider an organization making policy decisions. Given a new collection of unlabeled instances $\{\tilde{x}_i\}_{i=1}^m$, the organization wants to create a policy $\pi^*$ that minimizes a certain operational cost $\text{OpCost}(\pi, f^*, \{\tilde{x}_i\}_i)$. Of course, if the organization knew the true labels for the $\{\tilde{x}_i\}_i$'s beforehand, it would choose a policy to optimize the operational cost based directly on these labels, and would not need $f^*$. Since the labels are not known, the operational costs are calculated using the model's predictions, the $f^*(\tilde{x}_i)$'s. The difference between the traditional sequential process and the new simultaneous process is whether $f^*$ is chosen with or without knowledge of the operational cost.

As an example, consider $\{\tilde{x}_i\}_i$ as representing machines in a factory waiting to be repaired, where the first feature $\tilde{x}_{i,1}$ is the age of the machine, the second feature $\tilde{x}_{i,2}$ is the condition at its last inspection, etc. The value $f^*(\tilde{x}_i)$ is the predicted probability of failure for $\tilde{x}_i$. Policy $\pi^*$ is the order in which the machines $\{\tilde{x}_i\}_i$ are repaired, which is chosen based on how likely they are to fail, that is, $\{f^*(\tilde{x}_i)\}_i$, and on the costs of the various types of repairs needed. The traditional sequential process picks a model $f^*$, based on past failure data without the knowledge of operational cost, and afterwards computes $\pi^*$ based on an optimization problem involving the $\{f^*(\tilde{x}_i)\}_i$'s and the operational cost. The new simultaneous process picks $f^*$ and $\pi^*$ at the same time, based on optimism or pessimism on the operational cost of $\pi^*$.

Formally, the **sequential process** computes the policy according to two steps, as follows.

**Step 1:** Create function $f^*$ based on $\{(x_i, y_i)\}_i$ according to (1). That is

$$f^* \in \text{argmin}_{f \in \mathcal{F}^{unc}} \left( \sum_{i=1}^{n} l(f(x_i), y_i) + C_2 R(f) \right).$$

**Step 2:** Choose policy $\pi^*$ to minimize the operational cost,

$$\pi^* \in \text{argmin}_{\pi \in \Pi} \text{OpCost}(\pi, f^*, \{\tilde{x}_i\}_i).$$

The operational cost $\text{OpCost}(\pi, f^*, \{\tilde{x}_i\}_i)$ is the amount the organization will spend if policy $\pi$ is chosen in response to the values of $\{f^*(\tilde{x}_i)\}_i$.

To define the **simultaneous process**, we combine Steps 1 and 2 of the sequential process. We can choose an **optimistic bias**, where we prefer (all else being equal) a model providing lower costs, or we can choose a **pessimistic bias** that prefers higher costs, where the degree of optimism or pessimism is controlled by a parameter $C_1$. in other words, the optimistic bias lowers costs when there is uncertainty, whereas the pessimistic bias raises them. The new steps are as follows.

**Step 1:** Choose a model $f^\circ$ obeying one of the following:

$$\text{Optimistic Bias: } f^\circ \in \operatorname*{argmin}_{f \in \mathcal{F}^{unc}} \left[ \sum_{i=1}^{n} l\left(f(x_i), y_i\right) + C_2 R(f) + C_1 \min_{\pi \in \Pi} \text{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right) \right], \tag{2}$$

$$\text{Pessimistic Bias: } f^\circ \in \operatorname*{argmin}_{f \in \mathcal{F}^{unc}} \left[ \sum_{i=1}^{n} l\left(f(x_i), y_i\right) + C_2 R(f) - C_1 \min_{\pi \in \Pi} \text{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right) \right]. \tag{3}$$

**Step 2:** Compute the policy:

$$\pi^\circ \in \operatorname*{argmin}_{\pi \in \Pi} \text{OpCost}\left(\pi, f^\circ, \{\tilde{x}_i\}_i\right).$$

When $C_1 = 0$, the simultaneous process becomes the sequential process; the sequential process is a special case of the simultaneous process.

The optimization problem in the simultaneous process can be computationally difficult, particularly if the subproblem to minimize OpCost involves discrete optimization. However, if the number of unlabeled instances is small, or if the policy decision can be broken into several smaller subproblems, then even if the training set is large, one can solve Step 1 using different types of mathematical programming solvers, including MINLP solvers (Bonami et al., 2008), Nelder-Mead (Nelder and Mead, 1965) and Alternating Minimization schemes (Tulabandhula et al., 2011). One needs to be able to solve instances of that optimization problem in any case for Step 2 of the sequential process. The simultaneous process is more intensive than the sequential process in that it requires repeated solutions of that optimization problem, rather than a single solution.

The regularization term $R(f)$ can be for example, an $\ell_1$ or $\ell_2$ regularization term to encourage a sparse or smooth solution.

As the $C_1$ coefficient swings between large values for optimistic and pessimistic cases, the algorithm finds the best solution (having the lowest loss with respect to the data) for each possible cost. Once the regularization coefficient is too large, the algorithm will sacrifice empirical error in favor of lower costs, and will thus obtain solutions that are not reasonable. When that happens, we know we have already mapped out the full range of costs for reasonable solutions. This range can be used for pre-allocation decisions.

By sweeping over a range of $C_1$, we obtain a range of costs that we might incur. Based on this range, we can choose to allocate a reasonable amount of resources so that we can react best to whatever nature brings. This helps answer question Q1 in Section 1. In addition, we can pick a value of $C_1$ such that the resulting operational cost is a specific amount. In this case, we checking

whether a probabilistic model exists, corresponding to that cost, that is reasonably supported by data. This can answer question Q2 in Section 1.

It is possible for the set of feasible policies $\Pi$ to depend on recommendations $\{f(\tilde{x}_1),...,f(\tilde{x}_m)\}$, so that $\Pi = \Pi(f,\{\tilde{x}_i\}_i)$ in general. We will revisit this possibility in Section 4. It is also possible for the optimization over $\pi \in \Pi$ to be trivial, or the optimization problem could have a closed form solution. Our notation does accommodate this, and is more general.

One should not view the operational cost as a utility function that needs to be estimated, as in reinforcement learning, where we do not know the cost. Here one knows precisely what the cost will be under each possible outcome. Unlike in reinforcement learning, we have a complicated one shot decision problem at hand and have training data as well as future/unlabeled examples on which the predictive model makes prediction on.

The use of unlabeled data $\{\tilde{x}_i\}_i$ has been explored widely in the machine learning literature under semi-supervised, transductive, and unsupervised learning. In particular, we point out that the simultaneous process is not a semi-supervised learning method (see Chapelle et al., 2006), since it does not use the unlabeled data to provide information about the underlying distribution. A small unlabeled sample is not very useful for semi-supervised learning, but could be very useful for constructing a low-cost policy. The simultaneous process also has a resemblance to transductive learning (see Zhu, 2007), whose goal is to produce the output labels on the set of unlabeled examples; in this case, we produce a function (namely the operational cost) applied to those output labels. The simultaneous process, for a fixed choice of $C_1$, can also be considered as a multi-objective machine learning method, since it involves an optimization problem having two terms with competing goals (see Jin, 2006).

### 2.1 The Simultaneous Process in the Context of Structural Risk Minimization

In the framework of statistical learning theory (e.g., Vapnik, 1998; Pollard, 1984; Anthony and Bartlett, 1999; Zhang, 2002), prediction ability of a class of models is guaranteed when the class has low "complexity," where complexity is defined via covering numbers, VC (Vapnik-Chervonenkis) dimension, Rademacher complexity, gaussian complexity, etc. Limiting the complexity of the hypothesis space imposes a bias, and the classical image associated with the bias-variance tradeoff is provided in Figure 1(a). The set of good models is indicated on the axis of the figure. Models that are not good are either overfitted (explaining too much of the variance of the data, having a high complexity), or underfitted (having too strong of a bias and a high empirical error). By understanding complexity, we can find a model class where both the training error and the complexity are kept low. An example of increasingly complex model classes is the set of nested classes of polynomials, starting with constants, then linear functions, second order polynomials and so on.

In predictive modeling problems, there is often no one right statistical model when dealing with finite data sets, in fact there may be a whole class of good models. In addition, it is possible that a small change in the choice of predictive model could lead to a large change in the cost required to implement the policy recommended by the model. This occurs, for instance, when costs are based on objects (e.g., products) that come in discrete amounts. Figure 1(b) illustrates this possibility, by showing that there may be a variety of costs amongst the class of good models. The simultaneous process can find the range of costs for the set of good models, which can be used for allocation of costs, as discussed in the first question Q1 in the introduction.

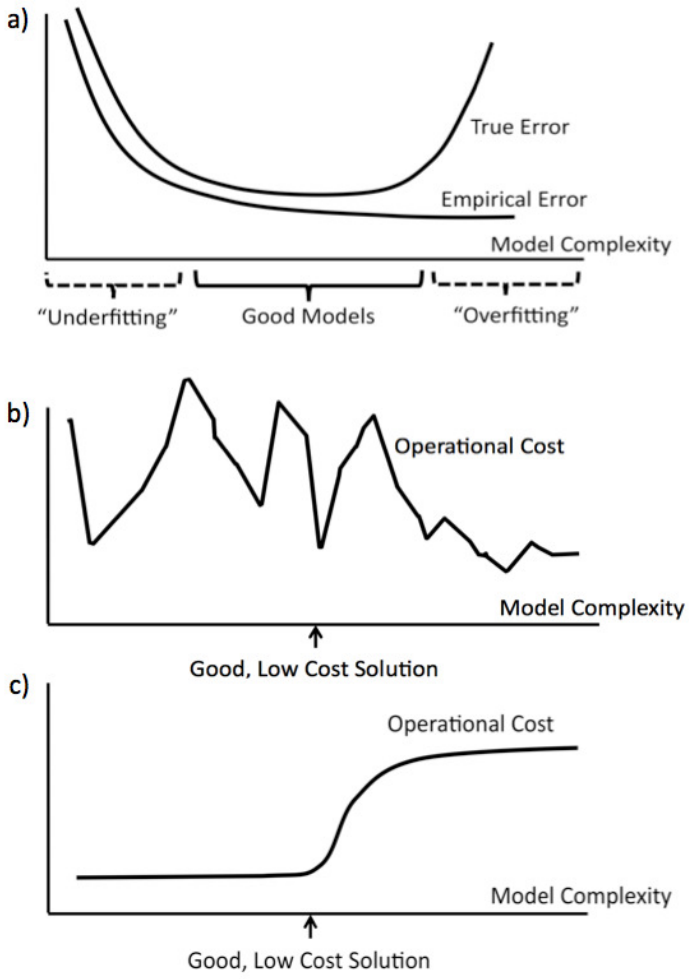



Figure 1: In all three plots, the x-axis represents model classes with increasing complexity. a) Relationship between training error and test error as a function of model complexity. b) A possible operational cost as a function of model complexity. c) Another possible operational cost.

Recall that question Q3 asked if our intuition about how much it will cost to solve a problem can help us produce a better probabilistic model. Figure 1 can be used to illustrate how this question can be answered. Assume we have a strong prior belief that the operational cost will not be above a certain fixed amount.

Accordingly, we will choose only amongst the class of low cost models. This can significantly limit the complexity of the hypothesis space, because the set of low-cost good models might be much smaller than the full space of good models. Consider, for example, the cost displayed in Figure 1(c), where only models on the left part of the plot would be considered, since they are low cost models. Because the hypothesis space is smaller, we may be able to produce a tighter

bound on the complexity of the hypothesis space, thereby obtaining a better prediction guarantee for the simultaneous process than for the sequential process. In Section 5 we develop results of this type. These results indicate that in some cases, the operational cost can be an important quantity for generalization.

## 3. Conceptual Demonstrations

We provide four examples. In the first, we estimate manpower requirements for a scheduling task. In the second, we estimate real estate prices for a purchasing decision. In the third, we estimate call arrival rates for a call center staffing problem. In the fourth, we estimate failure probabilities for manholes (access points to an underground electrical grid). The first two are small scale reproducible examples, designed to demonstrate new types of constraints due to operational costs. In the first example, the operational cost subproblem involves scheduling. In the second, it is a knapsack problem, and in the third, it is another multidimensional knapsack variant. In the fourth, it is a routing problem. In the first, second and fourth examples, the operational cost leads to a linear constraint, while in the third example, the cost leads to a quadratic constraint.

Throughout this section, we will assume that we are working with linear functions $f$ of the form $\beta^T x$ so that $\Pi(f, \{\tilde{x}_i\}_i)$ can be denoted by $\Pi(\beta, \{\tilde{x}_i\}_i)$. We will set $R(f)$ to be equal to $\|\beta\|_2^2$. We will also use the notation $\mathcal{F}^R$ to denote the set of linear functions that satisfy an additional property:

$$\mathcal{F}^R := \{f \in \mathcal{F}^{unc} : R(f) \leq C_2^*\},$$

where $C_2^*$ is a known constant greater than zero. We will use constant $C_2$ from (1), and also $C_2^*$ from the definition of $\mathcal{F}^R$, to control the extent of regularization. $C_2$ is inversely related to $C_2^*$. We use both versions interchangeably throughout the paper.

### 3.1 Manpower Data and Scheduling with Precedence Constraints

We aim to schedule the starting times of medical staff, who work at 6 stations, for instance, ultrasound, X-ray, MRI, CT scan, nuclear imaging, and blood lab. Current and incoming patients need to go through some of these stations in a particular order. The six stations and the possible orders are shown in Figure 2. Each station is denoted by a line. Work starts at the check-in (at time $\pi_1$) and ends at the check-out (at time $\pi_5$). The stations are numbered 6-11, in order to avoid confusion with the times $\pi_1$-$\pi_5$. The clinic has precedence constraints, where a station cannot be staffed (or work with patients) until the preceding stations are likely to finish with their patients. For instance, the check-out should not start until all the previous stations finish. Also, as shown in Figure 2, station 11 should not start until stations 8 and 9 are complete at time $\pi_4$, and station 9 should not start until station 7 is complete at time $\pi_3$. Stations 8 and 10 should not start until station 6 is complete. (This is related to a similar problem called *planning with preference* posed by F. Malucelli, Politecnico di Milano).

The operational goal is to minimize the total time of the clinic's operation, from when the check-in happens at time $\pi_1$ until the check-out happens at time $\pi_5$. We estimate the time it takes for each station to finish its job with the patients based on two variables: the new load of patients for the day at the station, and the number of current patients already present. The data are available as *manpower* in the R-package *bestglm*, using "Hour," "Load" and "Stay" columns. The training error is chosen to be the least squares loss between the estimated time for stations to finish their jobs

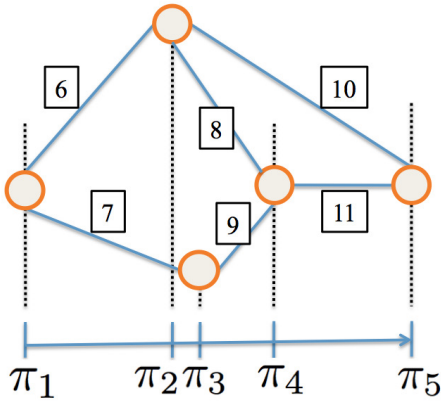



Figure 2: Staffing estimation with bias on scheduling with precedence constraints.

$(\beta^T x_i)$ and the actual times it took to finish $(y_i)$. The unlabeled data are the new load and current patients present at each station for a given period, given as $\tilde{x}_6, .., \tilde{x}_{11}$. Let $\pi$ denote the 5-dimensional real vector with coordinates $\pi_1, ..., \pi_5$.

The operational cost is the total time $\pi_5 - \pi_1$. Step 1, with an optimistic bias, can be written as:

$$\min_{\{\beta: \|\beta\|_2^2 \le C_2^*\}} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2 + C_1 \min_{\pi \in \Pi(\beta, \{\tilde{x}_i\}_i)} (\pi_5 - \pi_1), \tag{4}$$

where the feasible set $\Pi(\beta, \{\tilde{x}_i\}_i)$ is defined by the following constraints:

$$\pi_a + \beta^T \tilde{x}_i \le \pi_b; \ \ (a,i,b) \in \{(1,6,2),(1,7,3),(2,8,4),(3,9,4),(2,10,5),(4,11,5)\}$$
$$\pi_a \ge 0 \text{ for } a = 1, ..., 5.$$

To solve (4) given values of $C_1$ and $C_2$, we used a function-evaluation-based scheme called Nelder-Mead (Nelder and Mead, 1965) where at every iterate of $\beta$, the subproblem for $\pi$ was solved to optimality (using Gurobi).[1] $C_2$ was chosen heuristically based on (1) and kept fixed for the experiment beforehand.

Figure 3 shows the operational cost, training loss, and $r^2$ statistic[2] for various values of $C_1$. For $C_1$ values between 0 and 0.2, the operational cost varies substantially, by $\sim$16%. The $r^2$ values for both training and test vary much less, by $\sim$3.5%, where the best value happened to have the largest value of $C_1$. For small data sets, there is generally a variation between training and test: for this data split, there is a 3.16% difference in $r^2$ between training and test for plain least squares, and this is similar across various splits of the training and test data. This means that for the scheduling problem, there is a range of reasonable predictive models within about $\sim$3.5% of each other.

What we learn from this, in terms of the three questions in the introduction, is that: 1) There is a wide range of possible costs within the range of reasonable optimistic models. 2) We have found a reasonable scenario, supported by data, where the cost is 16% lower than in the sequential case.

1. Gurobi is the Gurobi Optimizer v3.0 from Gurobi Optimization, Inc. 2010.
2. If $\hat{y}_i$ are the predicted labels and $\bar{y}$ is the mean of $\{y_1, ..., y_n\}$, then the value of the $r^2$ statistic is defined as $1 - \sum_i (y_i - \hat{y}_i)^2 / \sum_i (y_i - \bar{y})^2$. Thus $r^2$ is an affine transformation of the sum of squares error. $r^2$ allows training and test accuracy to be measured on a comparable scale.

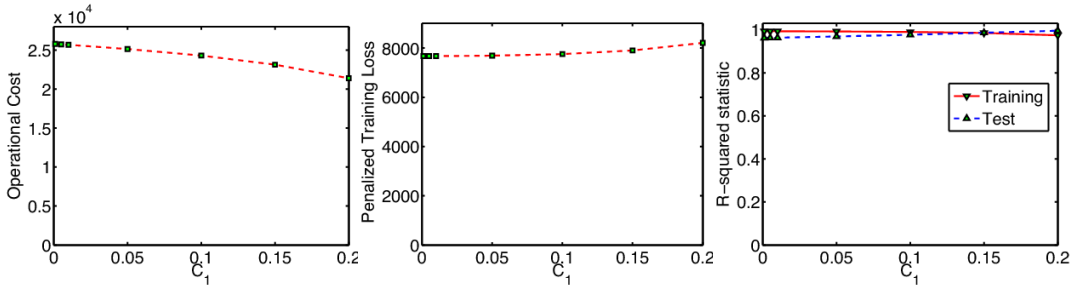



Figure 3: *Left:* Operational cost vs $C_1$. *Center:* Penalized training loss vs $C_1$. *Right:* R-squared statistic. $C_1 = 0$ corresponds to the baseline, which is the sequential formulation.

3) If we have a prior belief that the cost will be lower, the models that are more accurate are the ones with lower costs, and therefore we may not want to designate the full cost suggested by the sequential process. We can perhaps designate up to 16% less.

*Connection to learning theory:* In the experiment, we used tradeoff parameter $C_1$ to provide a soft constraint. Considering instead the corresponding hard constraint $\min_\pi(\pi_5 - \pi_1) \leq \alpha$, the total time must be at least the time for any of the three paths in Figure 2, and thus at least the average of them:

$$\begin{aligned}\alpha \geq & \min_{\pi \in \Pi\{\beta, \{\tilde{x}_i\}_i\}} \pi_5 - \pi_1 \\ & \geq \max\{(\tilde{x}_6 + \tilde{x}_{10})^T\beta, (\tilde{x}_6 + \tilde{x}_8 + \tilde{x}_{11})^T\beta, (\tilde{x}_7 + \tilde{x}_9 + \tilde{x}_{11})^T\beta\} \\ & \geq z^T\beta \end{aligned} \tag{5}$$

where

$$z = \frac{1}{3}[(\tilde{x}_6 + \tilde{x}_{10}) + (\tilde{x}_6 + \tilde{x}_8 + \tilde{x}_{11}) + (\tilde{x}_7 + \tilde{x}_9 + \tilde{x}_{11})].$$

The main result in Section 5, Theorem 6, is a learning theoretic guarantee in the presence of this kind of arbitrary linear constraint, $z^T\beta \leq \alpha$.

### 3.2 Housing Prices and the Knapsack Problem

A developer will purchase 3 properties amongst the 6 that are currently for sale and in addition, will remodel them. She wants to maximize the total value of the houses she picks (the value of a property is its purchase cost plus the fixed remodeling cost). The fixed remodeling costs for the 6 properties are denoted $\{c_i\}_{i=1}^6$. She estimates the purchase cost of each property from data regarding historical sales, in this case, from the *Boston Housing* data set (Bache and Lichman, 2013), which has 13 features. Let policy $\pi \in \{0,1\}^6$ be the 6-dimensional binary vector that indicates the properties she purchases. Also, $x_i$ represents the features of property $i$ in the training data and $\tilde{x}_i$ represents the features of a different property that is currently on sale. The training loss is chosen to be the sum of squares error between the estimated prices $\beta^T x_i$ and the true house prices $y_i$ for historical sales. The cost (in this case, total value) is the sum of the three property values plus the costs for repair work. A pessimistic bias on total value is chosen to motivate a min-max formulation. The resulting

(mixed-integer) program for Step 1 of the simultaneous process is:

$$\min_{\beta\in\{\beta:\beta\in\mathbb{R}^{13},\|\beta\|_2^2\le C_2^*\}} \sum_{i=1}^{n}(y_i-\beta^T x_i)^2 \\ +C_1\left[\max_{\pi\in\{0,1\}^6}\sum_{i=1}^{6}(\beta^T\tilde{x}_i+c_i)\pi_i \quad \text{subject to} \quad \sum_{i=1}^{6}\pi_i\le 3\right]. \tag{6}$$

Notice that the second term above is a 1-dimensional $\{0,1\}$ knapsack instance. Since the set of policies $\Pi$ does not depend on $\beta$, we can rewrite (6) in a cleaner way that was not possible directly with (4):

$$\min_{\beta}\max_{\pi}\left[\sum_{i=1}^{n}(y_i-\beta^T x_i)^2+C_1\sum_{i=1}^{6}(\beta^T\tilde{x}_i+c_i)\pi_i\right]$$

subject to

$$\beta\in\{\beta:\beta\in\mathbb{R}^{13},\|\beta\|_2^2\le C_2^*\}$$

$$\pi\in\left\{\pi:\pi\in\{0,1\}^6,\sum_{i=1}^{6}\pi_i\le 3\right\}. \tag{7}$$

To solve (7) with user-defined parameters $C_1$ and $C_2$, we use fminimax, available through Matlab's Optimization toolbox.[3]

For the training and unlabeled set we chose, there is a change in policy above and below $C_1 = 0.05$, where different properties are purchased. Figure 4 shows the operational cost which is the predicted total value of the houses after remodeling, the training loss, and $r^2$ values for a range of $C_1$. The training loss and $r^2$ values change by less than $\sim$3.5%, whereas the total value changes about 6.5%. We can again draw conclusions in terms of the questions in the introduction as follows.

The pessimistic bias shows that even if the developer chose the best response policy to the prices, she might end up with the expected total value of the purchased properties on the order of 6.5% less if she is unlucky. Also, we can now produce a realistic model where the total value is 6.5% less. We can use this model to help her understand the uncertainty involved in her investment.

Before moving to the next application of the proposed framework, we provide a bound analogous to that of (5). Let us replace the soft constraint represented by the second term of (6) with a hard constraint and then obtain a lower bound:

$$\alpha\ge\max_{\pi\in\{0,1\}^6,\sum_{i=1}^{6}\pi_i\le 3}\sum_{i=1}^{6}(\beta^T\tilde{x}_i)\pi_i\ge\sum_{i=1}^{6}(\beta^T\tilde{x}_i)\pi_i', \tag{8}$$

where $\pi'$ is some feasible solution of the linear programming relaxation of this problem that also gives a lower objective value. For instance picking $\pi_i' = 0.5$ for $i = 1,\ldots,6$ is a valid lower bound giving us a looser constraint. The constraint can be rewritten:

$$\beta^T\left(\frac{1}{2}\sum_{i=1}^{n}\tilde{x}_i\right)\le\alpha.$$

3. The version of the toolbox used is Version 5.1, Matlab R2010b, Mathworks, Inc.

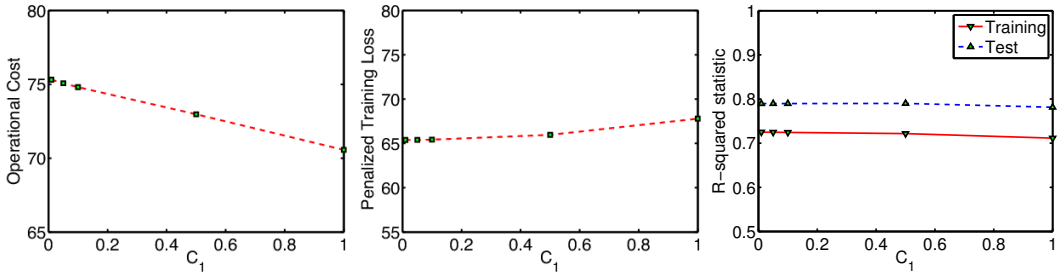



Figure 4: *Left:* Operational cost (total value) vs $C_1$. *Center:* Penalized training loss vs $C_1$. *Right:* R-squared statistic. $C_1 = 0$ corresponds to the baseline, which is the sequential formulation.

This is again a linear constraint on the function class parametrized by $\beta$, which we can use for the analysis in Section 5.

Note that if all six properties were being purchased by the developer instead of three, the knapsack problem would have a trivial solution and the regularization term would be explicit (rather than implicit).

### 3.3 A Call Center's Workload Estimation and Staff Scheduling

A call center management wants to come up with the per-half-hour schedule for the staff for a given day between 10am to 10pm. The staff on duty should be enough to meet the demand based on call arrival estimates $N(i), i = 1, ..., 24$. The staff required will depend linearly on the demand per half-hour. The demand per half-hour in turn will be computed based on the Erlang C model (Aldor-Noiman et al., 2009) which is also known as the square-root staffing rule. This particular model relates the demand $D(i)$ to the call arrival rate $N(i)$ in the following manner: $D(i) \propto N(i) + c\sqrt{N(i)}$ where $c$ determines where on the QED (Quality Efficiency Driven) curve the center wants to operate on. We make the simplifying assumptions that the service time for each customer is constant, and that the coefficient $c$ is 0.

If we know the call arrival rate $N(i)$, we can calculate the staffing requirements during each half hour. If we do not know the call arrival rate, we can estimate it from past data, and make optimistic or pessimistic staffing allocations.

There are additional staffing constraints as shown in Figure 5, namely, there are three sets of employees who work at the center such that: the first set can work only from 10am-3pm, the second can work from 1:30pm-6:30pm, and the third set works from 5pm-10pm. The operational cost is the total number of employees hired to work that day (times a constant, which is the amount each person is paid). The objective of the management is to reduce the number of staff on duty but at the same time maintain a certain quality and efficiency.

The call arrivals are modeled as a poisson process (Aldor-Noiman et al., 2009). What previous studies (Brown et al., 2001) have discovered about this estimation problem is that the square root of the call arrival rate tends to behave as a linear function of several features, including: day of the week, time of the day, whether it is a holiday/irregular day, and whether it is close to the end of the billing cycle.

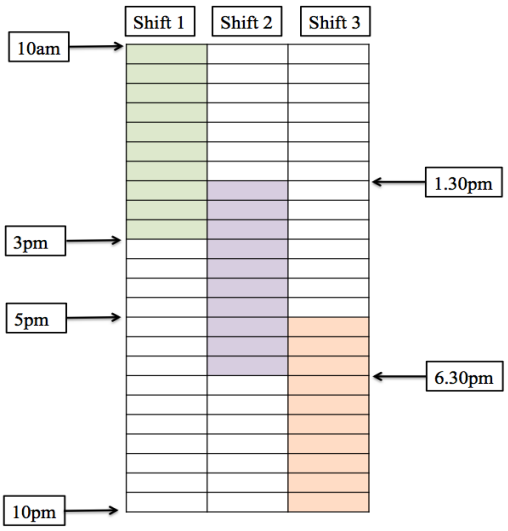



Figure 5: The three shifts for the call center. The cells represent half-hour periods, and there are 24 periods per work day. Work starts at 10am and ends at 10pm.

Data for call arrivals and features were collected over a period of 10 months from Mid-February 2004 to the end of December 2004 (this is the same data set as in Aldor-Noiman et al., 2009). After converting categorical variables into binary encodings (e.g., each of the 7 weekdays into 6 binary features) the number of features is 36, and we randomly split the data into a training set and test set (2764 instances for training; another 3308 for test).

We now formalize the optimization problem for the simultaneous process. Let policy $\pi \in \mathbb{Z}_+^3$ be a size three vector indicating the number of employees for each of the three shifts. The training loss is the sum of squares error between the estimated square root of the arrival rate $\beta^T x_i$ and the actual square root of the arrival rate $y_i := \sqrt{N(i)}$. The cost is proportional to the total number of employees signed up to work, $\sum_i \pi_i$. An optimistic bias on cost is chosen, so that the (mixed-integer) program for Step 1 is:

$$\min_{\beta:\|\beta\|_2^2 \le C_2^*} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2 + C_1 \left[ \min_{\pi} \sum_{i=1}^{3} \pi_i \text{ subject to } a_i^T \pi \ge (\beta^T \tilde{x}_i)^2 \text{ for } i = 1,\ldots,24, \pi \in \mathbb{Z}_+^3 \right], \tag{9}$$

where Figure 5 illustrates the matrix $A$ with the shaded cells containing entry 1 and 0 elsewhere. The notation $a_i$ indicates the $i^{\text{th}}$ row of $A$:

$$a_i(j) = \begin{cases} 1 & \text{if staff } j \text{ can work in half-hour period } i \\ 0 & \text{otherwise.} \end{cases}$$

To solve (9) we first relax the $\ell_2$-norm constraint on $\beta$ by adding another term to the function evaluation, namely $C_2\|\beta\|_2^2$. This, way we can use a function-evaluation based scheme that works for unconstrained optimization problems. As in the manpower scheduling example, we used an

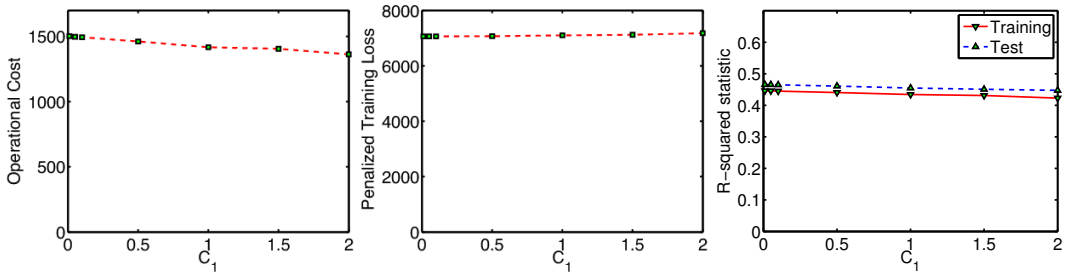



Figure 6: *Left:* Operational cost vs $C_1$. *Center:* Penalized training loss vs $C_1$. *Right:* R-squared statistic. $C_1 = 0$ corresponds to the baseline, which is the sequential formulation.

implementation of the Nelder-Mead algorithm, where at each step, Gurobi was used to solve the mixed-integer subproblem for finding the policy.

Figure 6 shows the operational cost, the training loss, and $r^2$ values for a range of $C_1$. The training loss and $r^2$ values change only $\sim$1.6% and $\sim$3.9% respectively, whereas the operational cost changes about 9.2%. Similar to the previous two examples, we can again draw conclusions in terms of the questions in Section 1 as follows. The optimistic bias shows that the management might incur operational costs on the order of 9% less if they are lucky. Further, the simultaneous process produces a reasonable model where costs are about 9% less. If the management team believes they will be reasonably lucky, they can justify designating substantially less than the amount suggested by the traditional sequential process.

Let us now investigate the structure of the operational cost regularization term we have in (9). For convenience, let us stack the quantities $(\beta^T \tilde{x}_i)^2$ as a vector $b \in \mathbb{R}^{24}$. Also let boldface symbol $\mathbf{1}$ represent a vector of all ones. If we replace the soft constraint represented by the second term with a hard constraint having an upper bound $\alpha$, we get:

$$\alpha \geq \min_{\pi \in \mathbb{Z}_+^3; A\pi \geq b} \sum_{i=1}^{3} \mathbf{1}^T \pi \overset{(\dagger)}{\geq} \min_{\pi \in \mathbb{R}_+^3; A\pi \geq b} \sum_{i=1}^{3} \mathbf{1}^T \pi \overset{(\ddagger)}{=} \max_{w \in \mathbb{R}_+^{24}; A^T w \leq \mathbf{1}} \sum_{i=1}^{24} w_i (\beta^T \tilde{x}_i)^2$$
$$\overset{(*)}{\geq} \sum_{i=1}^{24} \frac{1}{10} (\beta^T \tilde{x}_i)^2.$$

Here $\alpha$ is related to the choice of $C_1$ and is fixed. $(\dagger)$ represents an LP relaxation of the integer program with $\pi$ now belonging to the positive orthant rather than the cartesian product of set of positive integers. $(\ddagger)$ is due to LP strong duality and $(*)$ is by choosing an appropriate feasible dual variable. Specifically, we pick $w_i = \frac{1}{10}$ for $i = 1, \ldots, 24$, which is feasible because staff cannot work more than 10 half hour shifts (or 5 hours). With the three inequalities, we now have a constraint on $\beta$ of the form:

$$\sum_{i=1}^{24} (\beta^T \tilde{x}_i)^2 \leq 10\alpha.$$

This is a quadratic form in $\beta$ and gives an ellipsoidal feasible set. We already had a simple ellipsoidal feasibility constraint while defining the minimization problem of (9) of the form $\|\beta\|_2^2 \leq C_2^*$. Thus, we can see that our effective hypothesis set (the set of linear functionals satisfying these constraints)

has become smaller. This in turn affects generalization. We are investigating generalization bounds for this type of hypothesis set in separate ongoing work.

### 3.4 The Machine Learning and Traveling Repairman Problem (ML&TRP) (Tulabandhula et al., 2011)

Recently, power companies have been investing in intelligent "proactive" maintenance for the power grid, in order to enhance public safety and reliability of electrical service. For instance, New York City has implemented new inspection and repair programs for manholes, where a manhole is an access point to the underground electrical system. Electrical grids can be extremely large (there are on the order of 23,000-53,000 manholes in each borough of NYC), and parts of the underground distribution network in many cities can be as old as 130 years, dating from the time of Thomas Edison. Because of the difficulties in collecting and analyzing historical electrical grid data, electrical grid repair and maintenance has been performed reactively (fix it only when it breaks), until recently (Urbina, 2004). These new proactive maintenance programs open the door for machine learning to assist with smart grid maintenance.

Machine learning models have started to be used for proactive maintenance in NYC, where supervised ranking algorithms are used to rank the manholes in order of predicted susceptibility to failure (fires, explosions, smoke) so that the most vulnerable manholes can be prioritized (Rudin et al., 2010, 2012, 2011). The machine learning algorithms make reasonably accurate predictions of manhole vulnerability; however, they do not (nor would they, using any other prediction-only technique) take the cost of repairs into account when making the ranked lists. They do not know that it is unreasonable, for example, if a repair crew has to travel across the city and back again for each manhole inspection, losing important time in the process. The power company must solve an optimization problem to determine the best repair route, based on the machine learning model's output. We might wish to find a policy that is not only supported by the historical power grid data (that ranks more vulnerable manholes above less vulnerable ones), but also would give a better route for the repair crew. An algorithm that could find such a route would lead to an improvement in repair operations on NYC's power grid, other power grids across the world, and improvements in many different kinds of routing operations (delivery trucks, trains, airplanes).

The simultaneous process could be used to solve this problem, where the operational cost is the price to route the repair crew along a graph, and the probabilities of failure at each node in the graph must be estimated. We call this the "the machine learning and traveling repairman problem" (ML&TRP) and in our ongoing work (Tulabandhula et al., 2011) , we have developed several formulations for the ML&TRP. We demonstrated, using manholes from the Bronx region of NYC, that it is possible to obtain a much more practical route using the ML&TRP, by taking the cost of the route optimistically into account in the machine learning model. We showed also that from the routing problem, we can obtain a linear constraint on the hypothesis space, in order to apply the generalization analysis of Section 5 (and in order to address question Q3 of Section 1).

## 4. Connections to Robust Optimization

The goal of robust optimization (RO) is to provide the best possible policy that is acceptable under a wide range of situations.[4] This is different from the simultaneous process, which aims to find the

4. For more on Robust Optimation see http://en.wikipedia.org/wiki/Robust_optimization.

best policies and costs for specific situations. Note that it is not always desirable to have a policy that is robust to a wide range of situations; this is a question of whether to respond to every situation simultaneously or whether to understand the single worst situation that could reasonably occur (which is what the pessimistic simultaneous formulation handles). In general, robust optimization can be overly pessimistic, requiring us to allocate enough to handle all reasonable situations; it can be substantially more pessimistic than the pessimistic simultaneous process.

In robust optimization, if there are several real-valued parameters involved in the optimization problem, we might declare a reasonable range, called the "uncertainty set," for each parameter (e.g., $a_1 \in [9,10]$, $a_2 \in [1,2]$). Using techniques of RO, we would minimize the largest possible operational cost that could arise from parameter settings in these ranges. Estimation is not usually involved in the study of robust optimization (with some exceptions, see Xu et al., 2009, who consider support vector machines). On the other hand, one could choose the uncertainty set according to a statistical model, which is how we will build a connection to RO. Here, we choose the uncertainty set to be the class of models that fit the data to within $\varepsilon$, according to some fitting criteria.

The major goals of the field of RO include algorithms, geometry, and tractability in finding the best policy, whereas our work is not concerned with finding a robust policy, but we are concerned with estimation, taking the policy into account. Tractability for us is not always a main concern as we need to be able to solve the optimization problem, even to use the sequential process. Using even a small optimization problem as the operational cost might have a large impact on the model and decision. If the unlabeled set is not too large, or if the policy optimization problem can be broken into smaller subproblems, there is no problem with tractability. An example where the policy optimization might be broken into smaller subproblems is when the policy involves routing several different vehicles, where each vehicle must visit part of the unlabeled set; in that case there is a small subproblem for each vehicle. On the other hand, even though the goals of the simultaneous process and RO are entirely different, there is a strong connection with respect to the formulations for the simultaneous process and RO, and a class of problems for which they are equivalent. We will explore this connection in this section.

There are other methods that consider uncertainty in optimization, though not via the lens of estimation and learning. In the simplest case, one can perform both local and global sensitivity analysis for linear programs to ascertain uncertainty in the optimal solution and objective, but these techniques generally only handle simple forms of uncertainty (Vanderbei, 2008). Our work is also related to stochastic programming, where the goal is to find a policy that is robust to almost all of the possible circumstances (rather than all of them), where there are random variables governing the parameters of the problem, with known distributions (Birge and Louveaux, 1997). Again, our goal is not to find a policy that is necessarily robust to (almost all of) the worst cases, and estimation is again not the primary concern for stochastic programming, rather it is how to take known randomness into account when determining the policy.

### 4.1 Equivalence Between RO and the Simultaneous Process in Some Cases

In this subsection we will formally introduce RO. In order to connect RO to estimation, we will define the uncertainty set for RO, denoted $\mathcal{F}_{good}$, to be models for which the average loss on the sample is within $\varepsilon$ of the lowest possible. Then we will present the equivalence relationship between RO and the simultaneous process, using a minimax theorem.

In Section 2, we had introduced the notation $\{(x_i, y_i)\}_i$ and $\{\tilde{x}_i\}_i$ for labeled and unlabeled data respectively. We had also introduced the class $\mathcal{F}^{unc}$ in which we were searching for a function $f^*$ by minimizing an objective of the form (1). The uncertainty set $\mathcal{F}_{good}$ will turn out to be a subset of $\mathcal{F}^{unc}$ that depends on $\{(x_i, y_i)\}_i$ and $f^*$ but not on $\{\tilde{x}_i\}_i$.

We start with plain (non-robust) optimization, using a general version of the vanilla sequential process. Let $f$ denote an element of the set $\mathcal{F}_{good}$, where $f$ is pre-determined, known and fixed. Let the optimization problem for the policy decision $\pi$ be defined by:

$$\min_{\pi \in \Pi(f;\{\tilde{x}\}_i)} \text{OpCost}(\pi, f; \{\tilde{x}_i\}), \qquad \textit{(Base problem)} \tag{10}$$

where $\Pi(f; \{\tilde{x}_i\})$ is the feasible set for the optimization problem. Note that this is a more general version of the sequential process than in Section 2, since we have allowed the constraint set $\Pi$ to be a function of both $f$ and $\{\tilde{x}_i\}_i$, whereas in (2) and (3), only the objective and not the constraint set can depend on $f$ and $\{\tilde{x}_i\}_i$. Allowing this more general version of $\Pi$ will allow us to relate (10) to RO more clearly, and will help us to specify the additional assumptions we need in order to show the equivalence relationship. Specifically, in Section 2, OpCost depends on $(f, \{\tilde{x}_i\}_i)$ but not $\Pi$; whereas in RO, generally $\Pi$ depends on $(f, \{\tilde{x}_i\}_i)$ but not OpCost. The fact that OpCost does not need to depend on $f$ and $\{\tilde{x}_i\}_i$ is not a serious issue, since we can generally remove their dependence through auxiliary variables. For instance, if the problem is a minimization of the form (10), we can use an auxiliary variable, say $t$, to obtain an equivalent problem:

$$\begin{aligned} &\min_{\pi,t} t && \textit{(Base problem reformulated)} \\ \text{such that } &\pi \in \Pi(f; \{\tilde{x}_i\}) \\ &\text{OpCost}(\pi, f; \{\tilde{x}_i\}) \leq t \end{aligned}$$

where the dependence on $(f, \{\tilde{x}_i\}_i)$ is present only in the (new) feasible set. Since we had assumed $f$ to be fixed, this is a deterministic optimization problem (convex, mixed-integer, nonlinear, etc.).

Now, consider the case when $f$ is not known exactly but only known to lie in the uncertainty set $\mathcal{F}_{good}$. The robust counterpart to (10) can then be written as:

$$\min_{\pi \in \bigcap_{g \in \mathcal{F}_{good}} \Pi(g;\{\tilde{x}\}_i)} \max_{f \in \mathcal{F}_{good}} \text{OpCost}(\pi, f; \{\tilde{x}_i\}) \qquad \textit{(Robust counterpart)} \tag{11}$$

where we obtain a "robustly feasible solution" that is guaranteed to remain feasible for all values of $f \in \mathcal{F}_{good}$. In general, (11) is much harder to solve than (10) and is a topic of much interest in the robust optimization community. As we discussed earlier, there is no focus in (11) on estimation, but it is possible to embed an estimation problem within the description of the set $\mathcal{F}_{good}$, which we now define formally.

In Section 3, $\mathcal{F}^R$ (a subset of $\mathcal{F}^{unc}$) was defined as the set of linear functionals with the property that $R(f) \leq C_2^*$. That is,

$$\mathcal{F}^R = \{f : f \in \mathcal{F}^{unc}, R(f) \leq C_2^*\}.$$

We define $\mathcal{F}_{good}$ as a subset of $\mathcal{F}^R$ by adding an additional property:

$$\mathcal{F}_{good} = \left\{ f : f \in \mathcal{F}^R, \sum_{i=1}^{n} l(f(x_i), y_i) \leq \sum_{i=1}^{n} l(f^*(x_i), y_i) + \varepsilon \right\}, \tag{12}$$

for some fixed positive real $\varepsilon$. In (12), again $f^*$ is a solution that minimizes the objective in (1) over $\mathcal{F}^{unc}$. The right hand side of the inequality in (12) is thus constant, and we will henceforth denote it with a single quantity $C_1^*$. Substituting this definition of $\mathcal{F}_{good}$ in (11), and further making an important assumption (denoted **A1**) that $\Pi$ is not a function of $(f, \{\tilde{x}_i\}_i)$, we get the following optimization problem:

$$\min_{\pi\in\Pi} \max_{\{f\in\mathcal{F}^R:\sum_{i=1}^n l(f(x_i),y_i)\le C_1^*\}} \left[\text{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right)\right] \quad \textit{(Robust counterpart with assumptions)} \quad (13)$$

where $C_1^*$ now controls the amount of the uncertainty via the set $\mathcal{F}_{good}$.

Before we state the equivalence relationship, we restate the formulations for optimistic and pessimistic biases on operational cost in the simultaneous process from (2) and (3):

$$\min_{f\in\mathcal{F}^{unc}} \left[\sum_{i=1}^n l\left(f(x_i),y_i\right) + C_2 R(f) + C_1 \min_{\pi\in\Pi} \text{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right)\right] \textit{(Simultaneous optimistic)},$$

$$\min_{f\in\mathcal{F}^{unc}} \left[\sum_{i=1}^n l\left(f(x_i),y_i\right) + C_2 R(f) - C_1 \min_{\pi\in\Pi} \text{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right)\right] \textit{(Simultaneous pessimistic)}. \quad (14)$$

Apart from the assumption **A1** on the decision set $\Pi$ that we made in (13), we will also assume that $\mathcal{F}_{good}$ defined in (12) is convex; this will be assumption **A2**. If we also assume that the objective OpCost satisfies some nice properties (**A3**), and that uncertainty is characterized via the set $\mathcal{F}_{good}$, then we can show that the two problems, namely (14) and (13), are equivalent. Let $\Leftrightarrow$ denote equivalence between two problems, meaning that a solution to one side translates into the solution of the other side for some parameter values $(C_1, C_1^*, C_2, C_2^*)$.

**Proposition 1** *Let $\Pi(f;\{\tilde{x}_i\}_i) = \Pi$ be compact, convex, and independent of parameters $f$ and $\{\tilde{x}_i\}_i$ (assumption **A1**). Let $\{f \in \mathcal{F}^R : \sum_{i=1}^n l(f(x_i),y_i) \le C_1^*\}$ be convex (assumption **A2**). Let the cost (to be minimized)* OpCost$(\pi, f, \{\tilde{x}_i\}_i)$ *be concave continuous in $f$ and convex continuous in $\pi$ (assumption **A3**). Then, the robust optimization problem (13) is equivalent to the pessimistic bias optimization problem (14). That is,*

$$\min_{\pi\in\Pi} \max_{\{f\in\mathcal{F}^R:\sum_{i=1}^n l(f(x_i),y_i)\le C_1^*\}} \left[\text{OpCost}(\pi, f, \{\tilde{x}_i\}_i)\right] \Leftrightarrow$$
$$\min_{f\in\mathcal{F}^{unc}} \left[\sum_{i=1}^n l\left(f(x_i),y_i\right) + C_2 R(f) - C_1 \min_{\pi\in\Pi} \text{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right)\right].$$

**Remark 2** *That the equivalence applies to linear programs (LPs) is clear because the objective is linear and the feasible set is generally a polyhedron, and is thus convex. For integer programs, the objective* OpCost *satisfies continuity, but the feasible set is typically not convex, and hence, the result does not generally apply to integer programs. In other words, the requirement that the constraint set $\Pi$ be convex excludes integer programs.*

To prove Proposition 1, we restate a well-known generalization of von Neumann's minimax theorem and some related definitions.

**Definition 3** *A linear topological space (also called a topological vector space) is a vector space over a topological field (typically, the real numbers with their standard topology) with a topology such that vector addition and scalar multiplication are continuous functions. For example, any normed vector space is a linear topological space. A function $h$ is upper semicontinuous at a point $p_0$ if for every $\varepsilon > 0$ there exists a neighborhood $U$ of $p_0$ such that $h(p) \leq h(p_0) + \varepsilon$ for all $p \in U$. A function $h$ defined over a convex set is quasi-concave if for all $p, q$ and $\lambda \in [0,1]$ we have $h(\lambda p + (1-\lambda)q) \geq \min(h(p), h(q))$. Similar definitions follow for lower semicontinuity and quasi-convexity.*

**Theorem 4** *(Sion's minimax theorem Sion, 1958) Let $\Pi$ be a compact convex subset of a linear topological space and $\Xi$ be a convex subset of a linear topological space. Let $G(\pi, \xi)$ be a real function on $\Pi \times \Xi$ such that*

*(i) $G(\pi, \cdot)$ is upper semicontinuous and quasi-concave on $\Xi$ for each $\pi \in \Pi$;*

*(ii) $G(\cdot, \xi)$ is lower semicontinuous and quasi-convex on $\Pi$ for each $\xi \in \Xi$.*

*Then*

$$\min_{\pi \in \Pi} \sup_{\xi \in \Xi} G(\pi, \xi) = \sup_{\xi \in \Xi} \min_{\pi \in \Pi} G(\pi, \xi).$$

We can now proceed to the proof of Proposition (1).

**Proof** *(Of Proposition 1)* We start from the left hand side of the equivalence we want to prove:

$$\begin{aligned}
&\min_{\pi \in \Pi} \max_{\{f \in \mathcal{F}^R : \sum_{i=1}^n l(f(x_i), y_i) \leq C_1^*\}} \Big[ \mathrm{OpCost}(\pi, f, \{\tilde{x}_i\}_i) \Big] \\
\overset{(a)}{\Leftrightarrow} &\max_{\{f \in \mathcal{F}^R : \sum_{i=1}^n l(f(x_i), y_i) \leq C_1^*\}} \min_{\pi \in \Pi} \Big[ \mathrm{OpCost}(\pi, f, \{\tilde{x}_i\}_i) \Big] \\
\overset{(b)}{\Leftrightarrow} &\max_{f \in \mathcal{F}^{unc}} \Big[ -\frac{1}{C_1} \Big( \sum_{i=1}^n l(f(x_i), y_i) - C_1^* \Big) - \frac{C_2}{C_1} \Big( R(f) - C_2^* \Big) + \min_{\pi \in \Pi} \mathrm{OpCost}(\pi, f, \{\tilde{x}_i\}_i) \Big] \\
\overset{(c)}{\Leftrightarrow} &\min_{f \in \mathcal{F}^{unc}} \left[ \sum_{i=1}^n l\left(f(x_i), y_i\right) + C_2 R(f) - C_1 \min_{\pi \in \Pi} \mathrm{OpCost}\left(\pi, f, \{\tilde{x}_i\}_i\right) \right].
\end{aligned}$$

which is the right hand side of the logical equivalence in the statement of the theorem. In step $(a)$ we applied Sion's minimax theorem (Theorem 4) which is satisfied because of the assumptions we made. In step $(b)$, we picked Lagrange coefficients, namely $\frac{1}{C_1}$ and $\frac{C_2}{C_1}$, both of which are positive. In particular, $C_1^*$ and $C_1$ as well as $C_2^*$ and $C_2$ are related by the Lagrange relaxation equivalence (strong duality). In $(c)$, we multiplied the objective with $C_1$ throughout, pulled the negative sign in front, and removed the constant terms $C_1^*$ and $C_2 C_2^*$ and used the following observation: $\max_a -g(a) = -\min_a g(a)$; and finally, removed the negative sign in front as this does not affect equivalence. ■

The equivalence relationship of Proposition 1 shows that there is a problem class in which each instance can be viewed either as a RO problem or an estimation problem with an operational cost bias. We can use ideas from RO to make the simultaneous process more general. Before doing so, we will characterize $\mathcal{F}_{good}$ for several specific loss functions.

### 4.2 Creating Uncertainty Sets for RO Using Loss Functions from Machine Learning

Let us for simplicity specialize our loss function to the least squares loss. Let $X$ be an $n \times p$ matrix with each training instance $x_i$ forming the $i^{th}$ row. Also let $Y$ be the $n$-dimensional vector of all the labels $y_i$. Then the loss term of (1) can be written as:

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 = \sum_{i=1}^{n}(y_i - \beta^T x_i)^2 = \|Y - X\beta\|_2^2.$$

Let $\beta^*$ be a parameter corresponding to $f^*$ in (1). Then the definition of $\mathcal{F}_{good}$ in terms of the least squares loss is:

$$\mathcal{F}_{good} = \{f : f \in \mathcal{F}^R, \|Y - X\beta\|_2^2 \leq \|Y - X\beta^*\|_2^2 + \varepsilon\} = \{f : f \in \mathcal{F}^R, \|Y - X\beta\|_2^2 \leq C_1^*\}.$$

Since each $f \in \mathcal{F}_{good}$ corresponds to at least one $\beta$, the optimization of (1) can be performed with respect to $\beta$. In particular, the constraint $\|Y - X\beta\| \leq C_1^*$ is an ellipsoid constraint on $\beta$. For the purposes of the robust counterpart in (11), we can thus say that the uncertainty is of the ellipsoidal form. In fact, ellipsoidal constraints on uncertain parameters are widely used in robust optimization, especially because the resulting optimization problems often remain tractable.

Box constraints are also a popular way of incorporating uncertainty into robust optimization. For box constraints, the uncertainty over the $p$-dimensional parameter vector $\beta = [\beta_1, ..., \beta_p]^T$ is written for $i = 1, ..., p$ as $LB_i \leq \beta_i \leq UB_i$, where $\{LB_i\}_i$ and $\{UB_i\}_i$ are real-valued upper and lower bounds that together define the box intervals.

Our main point in this subsection is that one can potentially derive a very wide range of uncertainty sets for robust optimization using different loss functions from machine learning. Box constraints and ellipsoidal constraints are two simple types of constraints that could potentially be the set $\mathcal{F}_{good}$, which arise from two different loss functions, as we have shown. The least squares loss leads to ellipsoidal constraints on the uncertainty set, but it is unclear what the structure would be for uncertainty sets arising from the 0-1 loss, ramp loss, hinge loss, logistic loss and exponential loss among others. Further, it is possible to create a loss function for fitting data to a probabilistic model using the method of maximum likelihood; uncertainty sets for maximum likelihood could thus be established. Table 4.2 shows several different popular loss functions and the uncertainty sets they might lead to. Many of these new uncertainty sets do not always give tractable mathematical programs, which could explain why they are not commonly considered in the optimization literature.

*The sequential process for RO.* If we design the uncertainty sets as described above, with respect to a machine learning loss function, the sequential process described in Section 2 can be used with robust optimization. This proceeds in three steps:

1. use a learning algorithm on the training data to get $f^*$,

2. establish an uncertainty set based on the loss function and $f^*$, for example, ellipsoidal constraints arising from the least squares loss (or one could use any of the new uncertainty sets discussed in the previous paragraph),

3. use specialized optimization techniques to solve for the best policy, with respect to the uncertainty set.

| Loss function | Uncertainty set description |
|---|---|
| least squares | $\|Y - X\beta\|_2^2 \leq \|Y - X\beta^*\|_2^2 + \varepsilon$ (ellipsoid) |
| 0-1 loss | $\mathbf{1}_{[f(x_i)\neq y_i]} \leq \mathbf{1}_{[f^*(x_i)\neq y_i]} + \varepsilon$ |
| logistic loss | $\sum_{i=1}^n \log(1+e^{-y_i f(x_i)}) \leq \sum_{i=1}^n \log(1+e^{-y_i f^*(x_i)}) + \varepsilon$ |
| exponential loss | $\sum_{i=1}^n e^{-y_i f(x_i)} \leq \sum_{i=1}^n e^{-y_i f^*(x_i)} + \varepsilon$ |
| ramp loss | $\sum_{i=1}^n \min(1,\max(0,1-y_i f(x_i))) \leq \sum_{i=1}^n \min(1,\max(0,1-y_i f^*(x_i))) + \varepsilon$ |
| hinge loss | $\sum_{i=1}^n \max(0,1-y_i f(x_i)) \leq \sum_{i=1}^n \max(0,1-y_i f^*(x_i)) + \varepsilon$ |

Table 1: Table showing a summary of different possible uncertainty set descriptions that are based on ML loss functions.

We note that the uncertainty sets created by the 0-1 loss and ramp loss for instance, are non-convex, consequently assumption (**A2**) and Proposition 1 do not hold for robust optimization problems that use these sets.

### 4.3 The Overlap Between The Simultaneous Process and RO

On the other end of the spectrum from robust optimization, one can think of "optimistic" optimization where we are seeking the best value of the objective in the best possible situation (as oppose to the worst possible situation in RO). For optimistic optimization, more uncertainty is favorable, and we find the best policy for the best possible situation. This could be useful in many real applications where one not only wants to know the worst-case conservative policy but also the best case risk-taking policy. A typical formulation, following (11) can be written as:

$$\min_{\pi \in \bigcup_{g\in\mathcal{F}_{good}} \Pi(g;\{\tilde{x}\}_i)} \ \min_{f\in\mathcal{F}_{good}} \text{OpCost}(\pi, f; \{\tilde{x}_i\}). \qquad (\textit{Optimistic optimization})$$

In optimistic optimization, we view operational cost optimistically ($\min_{f\in\mathcal{F}_{good}}$ OpCost) whereas in the robust optimization counterpart (11), we view operational cost conservatively ($\max_{f\in\mathcal{F}_{good}}$ OpCost). The policy $\pi^*$ is feasible in more situations in RO ($\min_{\pi\in\cap_{g\in\mathcal{F}_{good}}\Pi}$) since it must be feasible with respect to each $g \in \mathcal{F}_{good}$, whereas the OpCost is lower in optimistic optimization ($\min_{\pi\in\cup_{g\in\mathcal{F}_{good}}\Pi}$) since it need only be feasible with respect to at least one of the $g$'s. Optimistic optimization has not been heavily studied, possibly because a (min-min) formulation is relatively easier to solve than its (min-max) robust counterpart, and so is less computationally interesting. Also, one generally plans for the worst case more often than for the best case, particularly when no estimation is involved. In the case where estimation is involved, both optimistic and robust optimization could potentially be useful to a practitioner.

Both optimistic optimization and robust optimization, considered with respect to uncertainty sets $\mathcal{F}_{good}$, have non-trivial overlap with the simultaneous process. In particular, we showed in Proposition 1 that pessimistic bias on operational cost is equivalent to robust optimization under specific conditions on OpCost and $\Pi$. Using an analogous proof, one can show that optimistic bias on operational cost is equivalent to optimistic optimization under the same set of conditions. Both robust and optimistic optimization and the simultaneous process encompass large classes of problems, some of which overlap. Figure 7 represents the overlap between the three classes of

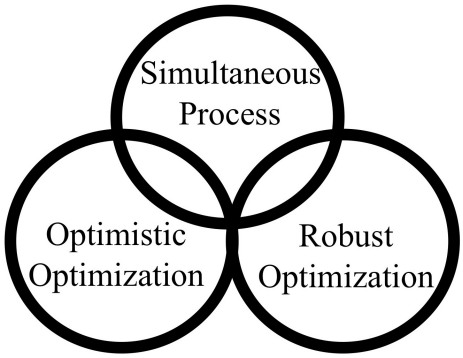



Figure 7: Set based description of the proposed framework (top circle) and its relation to robust (right circle) and optimistic (left circle) optimizations. The regions of intersection are where the conditions on the objective OpCost and the feasible set $\Pi$ are satisfied.

problems. There is a class of problems that fall into the simultaneous process, but are not equivalent to robust or optimistic optimization problems. These are problems where we use operational cost to assist with estimation, as in the call center example and ML&TRP discussed in Section 3. Typically problems in this class have $\Pi = \Pi(f; \{\tilde{x}_i\}_i)$. This class includes problems where the bias can be either optimistic or pessimistic, and for which $F_{good}$ has a complicated structure, beyond ellipsoidal or box constraints. There are also problems contained in either robust optimization or optimistic optimization alone and do not belong to the simultaneous process. Typically, again, this is when $\Pi$ depends on $f$. Note that the housing problem presented in Section 3 lies within the intersection of optimistic optimization and the simultaneous process; this can be deduced from (7).

In Section 5, we will provide statistical guarantees for the simultaneous process. These are very different from the style of probabilistic guarantees in the robust optimization literature. There are some "sample complexity" bounds in the RO literature of the following form: how many observations of uncertain data are required (and applied as simultaneous constraints) to maintain robustness of the solution with high probability? There is an unfortunate overlap in terminology; these are totally different problems to the sample complexity bounds in statistical learning theory. From the learning theory perspective, we ask: how many training instances does it take to come up with a model $\beta$ that we reasonably know to be good? We will answer that question for a very general class of estimation problems.

## 5. Generalization Bound with New Linear Constraints

In this section, we give statistical learning theoretic results for the simultaneous process that involve counting integer points in convex bodies. Generalization bounds are probabilistic guarantees, that often depend on some measure of the complexity of the hypothesis space. Limiting the complexity of the hypothesis space equates to a better bound. In this section, we consider the complexity of hypothesis spaces that results from an operational cost bias.

This enables us to answer in a quantitative manner, question Q3 in the introduction: "Can our intuition about how much it will cost to solve a problem help us produce a better probabilistic model?"

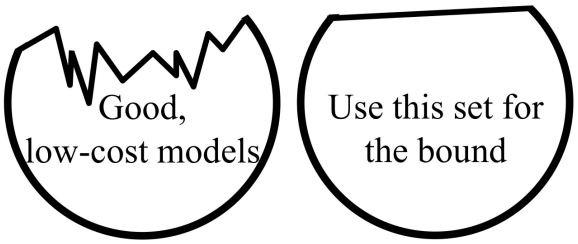



Figure 8: Left: hypothesis space for intersection of good models (circular, to represent $\ell_q$ ball) with low cost models (models below cost threshold, one side of wiggly curve). Right: relaxation to intersection of a half space with an $\ell_q$ ball.

Generalization bounds have been well established for *norm-based* constraints on the hypothesis space, but the emphasis has been more on qualitative dependence (e.g., using big-O notation) and the constants are not emphasized. On the other hand, for a practitioner, every prior belief should reduce the number of examples they need to collect, as these examples may each be expensive to obtain; thus constants within the bounds, and even their approximate values, become important (Bousquet, 2003). We thus provide bounds on the covering number for new types of hypothesis spaces, emphasizing the role of constants.

To establish the bound, it is sufficient to provide an upper bound on the covering number. There are many existing generic generalization bounds in the literature (e.g., Bartlett and Mendelson, 2002), which combined with our bound, will yield a specific generalization bound for machine learning with operational costs, as we will construct in Theorem 10.

In Section 3, we showed that a bias on the operational cost can sometimes be transformed into linear constraints on model parameter $\beta$ (see Equations (5) and (8)). There is a broad class of other problems for which this is true, for example, for applications related to those presented in Section 3. Because we are able to obtain linear constraints for such a broad class of problems, we will analyze the case of linear constraints here. The hypothesis we consider is thus the intersection of an $\ell_q$ ball and a halfspace. This is illustrated in Figure 8.

The plan for the rest of the section is as follows. We will introduce the quantities on which our main result in this section depends. Then, we will state the main result (Theorem 6). Following that, we will build up to a generalization bound (Theorem 10) that incorporates Theorem 6. After that will be the proof of Theorem 6.

**Definition 5** *(Covering Number, Kolmogorov and Tikhomirov, 1959) Let $A \subseteq \Gamma$ be an arbitrary set and $(\Gamma, \rho)$ a (pseudo-)metric space. Let $|\cdot|$ denote set size.*

- *For any $\varepsilon > 0$, an **$\varepsilon$-cover** for $A$ is a finite set $U \subseteq \Gamma$ (not necessarily $\subseteq A$) s.t. $\forall a \in A, \exists u \in U$ with $d_\rho(a,u) \leq \varepsilon$.*
- *The **covering number** of $A$ is $N(\varepsilon, A, \rho) := \inf_U |U|$ where $U$ is an $\varepsilon$-cover for $A$.*

We are given the set of $n$ instances $S := \{x_i\}_{i=1}^n$ with each $x_i \in \mathcal{X} \subseteq \mathbb{R}^p$ where $\mathcal{X} = \{x : \|x\|_r \leq X_b\}$, $2 \leq r \leq \infty$ and $X_b$ is a known constant. Let $\mu_\mathcal{X}$ be a probability measure on $\mathcal{X}$. Let $x_i$ be arranged as rows of a matrix $X$. We can represent the columns of $X = [x_1 \ \ldots \ x_n]^T$ with $h_j \in \mathbb{R}^n$, $j = 1, \ldots, p$,

so $X$ can also be written as $[h_1 \cdots h_p]$. Define function class $\mathcal{F}$ as the set of linear functionals whose coefficients lie in an $\ell_q$ ball and with a set of linear constraints:

$$\mathcal{F} := \{f : f(x) = \beta^T x, \beta \in \mathcal{B}\} \text{ where}$$
$$\mathcal{B} := \left\{ \beta \in \mathbb{R}^p : \|\beta\|_q \leq B_b, \sum_{j=1}^{p} c_{j\nu}\beta_j + \delta_\nu \leq 1, \delta_\nu > 0, \nu = 1, ..., V \right\},$$

where $1/r + 1/q = 1$ and $\{c_{j\nu}\}_{j,\nu}$, $\{\delta_\nu\}_\nu$ and $B_b$ are known constants. The linear constraints given by the $c_{j\nu}$'s force the hypothesis space $\mathcal{F}$ to be smaller, which will help with generalization - this will be shown formally by our main result in this section. Let $\mathcal{F}_{|S}$ be defined as the restriction of $\mathcal{F}$ with respect to $S$.

Let $\{\tilde{c}_{j\nu}\}_{j,\nu}$ be proportional to $\{c_{j\nu}\}_{j,\nu}$:

$$\tilde{c}_{j\nu} \quad := \quad \frac{c_{j\nu} n^{1/r} X_b B_b}{\|h_j\|_r} \quad \forall j = 1, ..., p \text{ and } \nu = 1, ..., V.$$

Let $K$ be a positive number. Further, let the sets $P^K$ parameterized by $K$ and $P_c^K$ parameterized by $K$ and $\{\tilde{c}_{j\nu}\}_{j,\nu}$ be defined as

$$P^K := \left\{ (k_1, ..., k_p) \in \mathbb{Z}^p : \sum_{j=1}^{p} |k_j| \leq K \right\}.$$
$$P_c^K := \left\{ (k_1, ..., k_p) \in P^K : \sum_{j=1}^{p} \tilde{c}_{j\nu} k_j \leq K \; \forall \nu = 1, ..., V \right\}. \tag{15}$$

Let $|P^K|$ and $|P_c^K|$ be the sizes of the sets $P^K$ and $P_c^K$ respectively. The subscript $c$ in $P_c^K$ denotes that this polyhedron is a constrained version of $P^K$. As the linear constraints given by the $c_{j\nu}$'s force the hypothesis space to be smaller, they force $|P_c^K|$ to be smaller. Define $\tilde{X}$ to be equal to $X$ times a diagonal matrix whose $j^{th}$ diagonal element is $\frac{n^{1/r} X_b B_b}{\|h_j\|_r}$. Define $\lambda_{\min}(\tilde{X}^T \tilde{X})$ to be the smallest eigenvalue of the matrix $\tilde{X}^T \tilde{X}$, which will thus be non-negative. Using these definitions, we state our main result of this section.

**Theorem 6** *(Main result, covering number bound)*

$$N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) \leq \begin{cases} \min\{|P^{K_0}|, |P_c^K|\} & \text{if } \varepsilon < X_b B_b \\ 1 & \text{otherwise} \end{cases}, \tag{16}$$

*where*

$$K_0 = \left\lceil \frac{X_b^2 B_b^2}{\varepsilon^2} \right\rceil$$

*and*

$$K = \max \left\{ K_0, \left\lceil \frac{n X_b^2 B_b^2}{\lambda_{\min}(\tilde{X}^T \tilde{X}) \left[ \min_{\nu=1,...,V} \frac{\delta_\nu}{\sum_{j=1}^{p} |\tilde{c}_{j\nu}|} \right]^2} \right\rceil \right\}.$$

The theorem gives a bound on the $\ell_2$ covering number for the specially constrained class $\mathcal{F}_{|S}$. The bound improves as the constraints given by $c_{jv}$ on the operational cost become tighter. In other words, as the $c_{jv}$ impose more restrictions on the hypothesis space, $|P_c^K|$ decreases, and the covering number bound becomes smaller. This bound can be plugged directly into an established generalization bound that incorporates covering numbers, and this is done in what follows to obtain Theorem 10.

Note that $\min\{|P^{K_0}|, |P_c^K|\}$ can be tighter than $|P_c^K|$ when $\varepsilon$ is large. When $\varepsilon$ is larger than $X_b B_b$, we only need one closed ball of radius $\sqrt{n}\varepsilon$ to cover $\mathcal{F}_{|S}$, so $N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) = 1$. In that case, the covering number in Theorem 6 is appropriately bounded by 1. If $\varepsilon$ is large, but not larger than $X_b B_b$, then $|P_c^K|$ can be smaller than $|P^{K_0}|$. $|P^{K_0}|$ is the size of the polytope without the operational cost constraints. $|P_c^K|$ is the size of a potentially bigger polytope, but with additional constraints.

For this problem we generally assume that $n > p$; that is the number of examples is greater than the dimensionality $p$. In such a case, $\lambda_{\min}(\tilde{X}^T\tilde{X})$ can be shown to be bounded away from zero for a wide variety of distributions $\mu_{\mathcal{X}}$ (e.g., sub-gaussian zero-mean). When $\lambda_{\min}(\tilde{X}^T\tilde{X}) = 0$, the covering number bound becomes vacuous.

Let us introduce some notation in order to state the generalization bound results. Given any function $f \in \mathcal{F}$, we would like to minimize the expected future loss (also known as the expected risk), defined as:

$$R^{\text{true}}(l \circ f) := \mathbb{E}_{(x,y)\sim\mu_{\mathcal{X}\times\mathcal{Y}}}\Big[l(f(x),y)\Big] = \int l(f(x),y)\partial\mu_{\mathcal{X}\times\mathcal{Y}}(x,y),$$

where $l : \mathcal{Y}\times\mathcal{Y} \to \mathbb{R}$ is the (fixed) loss function we had previously defined in Section 2. The loss on the training sample (also known as the empirical risk) is:

$$R^{\text{emp}}(l \circ f, \{(x_i,y_i)\}_1^n) := \frac{1}{n}\sum_{i=1}^n l(f(x_i),y_i).$$

We would like to know that $R^{\text{true}}(l \circ f)$ is not too much more than $R^{\text{emp}}(l \circ f, \{(x_i,y_i)\}_1^n)$, no matter which $f$ we choose from $\mathcal{F}$. A typical form of generalization bound that holds with high probability for every function in $\mathcal{F}$ is

$$R^{\text{true}}(l \circ f) \le R^{\text{emp}}(l \circ f, \{(x_i,y_i)\}_1^n) + \text{Bound}(\text{complexity}(\mathcal{F}),n), \tag{17}$$

where the complexity term takes into account the constraints on $\mathcal{F}$, both the linear constraints, and the $\ell_q$-ball constraint. Theorem 6 gives an upper bound on the term Bound(complexity($\mathcal{F}$),$n$) in (17) above. In order to show this explicitly, we will give the definition of Rademacher complexity, restate how it appears in the relation between expected future loss and loss on training examples, and state an upper-bound for it in terms of the covering number.

**Definition 7** *(Rademacher Complexity) The empirical Rademacher complexity of $\mathcal{F}_{|S}$ is*[5]

$$\hat{\mathcal{R}}(\mathcal{F}_{|S}) = \mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\frac{2}{n}\sum_{i=1}^n \sigma_i f(x_i)\right] \tag{18}$$

*where $\{\sigma_i\}$ are Rademacher random variables ($\sigma_i = 1$ with prob. $1/2$ and $-1$ with prob. $1/2$). The Rademacher complexity is its expectation: $\mathcal{R}_n(\mathcal{F}) = \mathbb{E}_{S\sim(\mu_{\mathcal{X}})^n}[\hat{\mathcal{R}}(\mathcal{F}_{|S})]$.*

5. The factor 2 in the defining equation (18) is not very important. Some authors omit this factor and include it explicitly as a pre-factor in, for example, Theorem 8.

The empirical Rademacher complexity $\hat{\mathcal{R}}(\mathcal{F}_{|S})$ can be computed given $S$ and $\mathcal{F}$, and by concentration, will be close to the Rademacher complexity. The following result relates the true risk to the empirical risk and empirical Rademacher complexity for any function class $\mathcal{H}$ (see Bartlett and Mendelson, 2002, and references therein). Let the quantities $\mathcal{H}_{|S}, R^{true}(l \circ h)$ and $R^{\text{emp}}(l \circ h, \{x_i, y_i\}_1^n)$ be analogous to those we had defined for our specific class $\mathcal{F}$.

**Theorem 8** *(Rademacher Generalization Bound) For all $\delta > 0$, with probability at least $1-\delta, \forall h \in \mathcal{H}$,*

$$R^{\text{true}}(l \circ h) \leq R^{\text{emp}}(l \circ h, \{x_i, y_i\}_1^n) + \mathcal{L} \cdot \hat{\mathcal{R}}(\mathcal{H}_{|S}) + \frac{3}{\sqrt{2}} \sqrt{\frac{\log \frac{1}{\delta}}{n}}, \tag{19}$$

*where $\mathcal{L}$ is the Lipschitz constant of the loss function.*

Note that (19) is an explicit form of (17). We will now relate $\hat{\mathcal{R}}(\mathcal{F}_{|S})$ to covering numbers thus justifying the importance of statement (16) in Theorem 6. In particular the following infinite chaining argument also known as Dudley's integral (see Talagrand, 2005) relates $\hat{\mathcal{R}}(\mathcal{F}_{|S})$ to the covering number of the set $\mathcal{F}_{|S}$.

**Theorem 9** *(Relating Rademacher Complexity to Covering Numbers) We are given that $\forall x \in \mathcal{X}$, we have $f(x) \in [-X_b B_b, X_b B_b]$. Then,*

$$\frac{1}{X_b B_b} \hat{\mathcal{R}}(\mathcal{F}_{|S}) \leq 12 \int_0^\infty \sqrt{\frac{2 \log N(\alpha, \mathcal{F}, L_2(\mu_X^n))}{n}} d\alpha = 12 \int_0^\infty \sqrt{\frac{2 \log N(\sqrt{n}\alpha, \mathcal{F}_{|S}, \|\cdot\|_2)}{n}} d\alpha.$$

Our main result in Theorem 6 can be used in conjunction with Theorems 8 and 9, to directly see how the true error relates to the empirical error and the constraints on the restricted function class $\mathcal{F}$ (the $\ell_q$-norm bound on $\beta$ and linear constraint on $\beta$ from the operational cost bias). Explicitly, that bound is here.

**Theorem 10** *(Generalization Bound for ML with Operational Costs) For all $\delta > 0$, with probability at least $1-\delta, \forall f \in \mathcal{F}$,*

$$R^{\text{true}}(l \circ f) \leq R^{\text{emp}}(l \circ f, \{x_i, y_i\}_1^n) + 12 \mathcal{L} X_b B_b \int_0^\infty \sqrt{\frac{2 \log N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2)}{n}} d\varepsilon + \frac{3}{\sqrt{2}} \sqrt{\frac{\log \frac{1}{\delta}}{n}},$$

*where*

$$N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) \leq \begin{cases} \min\{|P^{K_0}|, |P_c^K|\} & \text{if } \varepsilon < X_b B_b \\ 1 & \text{otherwise} \end{cases},$$

$$K_0 = \left\lceil \frac{X_b^2 B_b^2}{\varepsilon^2} \right\rceil,$$

*and*

$$K = \max \left\{ K_0, \left\lceil \frac{n X_b^2 B_b^2}{\lambda_{\min}(\tilde{X}^T \tilde{X}) \left[ \min_{v=1,\ldots,V} \frac{\delta_v}{\sum_{j=1}^p |\tilde{c}_{jv}|} \right]^2} \right\rceil \right\}$$

*are functions of $\varepsilon$.*

This bound implies that prior knowledge about the operational cost can be important for generalization. As our prior knowledge on the cost becomes stronger, the size of the hypothesis space becomes more restrictive, as seen through the constraints given by the $c_{jv}$. When this happens, the $|P_c^K|$ terms become smaller, and the whole bound becomes smaller. Note that the integral over $\varepsilon$ is taken from $\varepsilon = 0$ to $\varepsilon = \infty$. When $\varepsilon$ is larger than $X_b B_b$, as noted earlier, $N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) = 1$ and thus $\log N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) = 0$.

Before we move onto building the necessary tools to prove Theorem 6, we compare our result with the bound in our work on the ML&TRP (Tulabandhula et al., 2011). In that work, we considered a linear function class with a constraint on the $\ell_2$-norm and one additional linear inequality constraint on $\beta$. We then used a sample independent volumetric cap argument to get a covering number bound. Theorem 6 is in some ways an improvement of the other result: (1) we can now have multiple linear constraints on $\beta$; (2) our new result involves a sample-specific bounding technique for covering numbers, which is generally tighter; (3) our result applies to $\ell_q$ balls for $q \in [1,2]$ whereas the previous analysis holds only for $q = 2$. The volumetric argument in Tulabandhula et al. (2011) provided a scaling of the covering number. Specifically, the operational cost term for the ML&TRP allowed us to reduce the covering number term in the bound from $\sqrt{\log N(\cdot,\cdot,\cdot)}$ to $\sqrt{\log(\alpha N(\cdot,\cdot,\|\cdot\|_2))}$, or equivalently $\sqrt{\log N(\cdot,\cdot,\|\cdot\|_2) + \log\alpha}$, where $\alpha$ is a function of the operational cost constraint. If $\alpha$ obeys $\alpha \ll 1$, then there is a noticeable effect on the generalization bound, compared to almost no effect when $\alpha \approx 1$. In the present work, the bound does not scale the covering number like this, instead it is a very different approach giving a more direct bound.

### 5.1 Proof of Theorem 6

We make use of Maurey's Lemma (Barron, 1993) in our proof (in the same spirit as Zhang, 2002). The main ideas of Maurey's Lemma are used in many machine learning papers in various contexts (e.g., Koltchinskii and Panchenko, 2005; Schapire et al., 1998; Rudin and Schapire, 2009). Our proof of Theorem 6 adapts Maurey's Lemma to handle polyhedrons, and allows us to apply counting techniques to bound the covering number.

Recall that $X = [x_1 \ldots x_n]^T$ was also defined column-wise as $[h_1 \ \ldots \ h_p]$. We introduce two scaled sets $\{\tilde{h}_j\}_j$ and $\{\tilde{\beta}_j\}_j$ corresponding to $\{h_j\}_j$ and $\{\beta_j\}_j$ as follows:

$$\tilde{h}_j := \frac{n^{1/r} X_b B_b}{\|h_j\|_r} h_j \ \text{ for } j = 1, \ldots, p; \text{ and}$$
$$\tilde{\beta}_j := \frac{\|h_j\|_r}{n^{1/r} X_b B_b} \beta_j \ \text{ for } j = 1, \ldots, p.$$

These scaled sets will be convenient in places where we do not want to carry the scaling terms separately.

Any vector $y$ that is equal to $X\beta$ can thus be written in three different ways:

$$y = \sum_{j=1}^{p} \beta_j h_j, \text{ or}$$
$$y = \sum_{j=1}^{p} \tilde{\beta}_j \tilde{h}_j, \text{ or}$$
$$y = \sum_{j=1}^{p} |\tilde{\beta}_j| \text{sign}(\tilde{\beta}_j) \tilde{h}_j.$$

Our first lemma is a restatement of Maurey's lemma (revised version of Lemma 1 in Zhang, 2002). We provide a proof based on the law of large numbers (Barron, 1993) though other proof techniques also exist (see Jones, 1992, for a proof based on iterative approximation).

The lemma states that every point $y$ in the convex hull of $\{h_j\}_j$ is close to one of the points $y_K$ in a particular finite set.

**Lemma 11** *Let* $\max_{j=1,\ldots,p} \|\tilde{h}_j\|$ *be less than or equal to some constant b. If y belongs to the convex hull of set* $\{\tilde{h}_j\}_j$*, then for every positive integer* $K \geq 1$*, there exists* $y_K$ *in the convex hull of K points of set* $\{\tilde{h}_j\}_j$ *such that* $\|y - y_K\|^2 \leq \frac{b^2}{K}$.

**Proof** Let $y$ be written in the form:

$$y = \sum_{i=1}^{p} \bar{\gamma}_j \tilde{h}_j,$$

where for each $j = 1, \ldots, p$, $\bar{\gamma}_j \geq 0$ and $\sum_{j=1}^{p} \bar{\gamma}_j \leq 1$. Let $\bar{\gamma}_{p+1} := 1 - \sum_{j=1}^{p} \bar{\gamma}_j$.

Consider a discrete distribution $\mathcal{D}$ formed by the coefficient vector $(\bar{\gamma}_1, .., \bar{\gamma}_p, \bar{\gamma}_{p+1})$. Associate a random variable $\tilde{\mathsf{h}}$ with support set $\{\tilde{h}_1, \ldots, \tilde{h}_p, \mathbf{0}\}$. That is, $\Pr(\tilde{\mathsf{h}} = \tilde{h}_j) = \bar{\gamma}_j$, $j = 1, \ldots, p$ and $\Pr(\tilde{\mathsf{h}} = \mathbf{0}) = \bar{\gamma}_{p+1}$.

Draw $K$ observations $\{\tilde{h}^1, \ldots, \tilde{h}^K\}$ uniformly and independently from $\mathcal{D}$ and form the sample average $\mathsf{y}_K := \frac{1}{K} \sum_{s=1}^{K} \tilde{h}^s$. Here, we are using the superscript index to denote the observation number. The mean of this random variable $\mathsf{y}_K$ is:

$$\begin{aligned} \mathbb{E}_{\mathcal{D}}[\mathsf{y}_K] &= \frac{1}{K} \sum_{s=1}^{K} \mathbb{E}_{\mathcal{D}}[\tilde{h}^s] \text{ where} \\ \mathbb{E}_{\mathcal{D}}[\tilde{h}^s] &= \sum_{j=1}^{p+1} \Pr(\tilde{\mathsf{h}} = \tilde{h}_j) \tilde{h}_j = \sum_{j=1}^{p} \bar{\gamma}_j \tilde{h}_j = y \end{aligned}$$

hence $\mathbb{E}_{\mathcal{D}}[\mathsf{y}_K] = y$.

The expected distance between $\mathsf{y}_K$ and $y$ is:

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}}[\|\mathsf{y}_K - y\|^2] &= \mathbb{E}_{\mathcal{D}}[\|\mathsf{y}_K - \mathbb{E}_{\mathcal{D}}[\mathsf{y}_K]\|^2] = \mathbb{E}\left[\sum_{i=1}^{n}(\mathsf{y}_K - \mathbb{E}_{\mathcal{D}}[\mathsf{y}_K])_i^2\right] \\
&\overset{(\dagger)}{=} \sum_{i=1}^{n}\mathrm{Var}((\mathsf{y}_K)_i) \overset{(*)}{=} \sum_{i=1}^{n}\frac{1}{K}\mathrm{Var}((\tilde{\mathsf{h}})_i) \\
&\overset{(\ddagger)}{=} \frac{1}{K}\sum_{i=1}^{n}\left(\mathbb{E}_{\mathcal{D}}[(\tilde{\mathsf{h}})_i^2] - \mathbb{E}_{\mathcal{D}}[(\tilde{\mathsf{h}})_i]^2\right) \overset{(\circ)}{=} \frac{1}{K}\left(\mathbb{E}_{\mathcal{D}}[\|\tilde{\mathsf{h}}\|^2] - \|\mathbb{E}_{\mathcal{D}}[\tilde{\mathsf{h}}]\|^2\right) \\
&\leq \frac{1}{K}\mathbb{E}_{\mathcal{D}}[\|\tilde{\mathsf{h}}\|^2] \leq \frac{b^2}{K} \qquad (20)
\end{aligned}$$

where we have used $i$ to be the index for the $i^{th}$ coordinate of the $n$ dimensional vectors. $(\dagger)$ follows from the definition of variance coordinate-wise. $(*)$ follows because each component of $\mathsf{y}_K$ is a sample average. $(\ddagger)$ also follows from the definition of variance. At step $(\circ)$, we rewrite the previous summations involving squares into ones that use the Hilbert norm. Our assumption on $\max_{j=1,\ldots,p}\|\tilde{h}_j\|$ tells us that $\mathbb{E}_{\mathcal{D}}[\|\tilde{\mathsf{h}}\|^2] \leq b^2$ leading to (20). Since the squared Hilbert norm of the sample mean is bounded in this way, there exists a $y_K$ that satisfies the inequality, so that

$$\|y_K - y\|^2 \leq \frac{b^2}{K}.$$

■

The following corollary states explicitly that an approximation to $y$ exists that is a linear combination with coefficients chosen from a particular discrete set.

**Corollary 12** *For any $y$ and $K$ as considered above, we can find non-negative integers $m_1, \ldots, m_p$ such that $\sum_{j=1}^{p} m_j \leq K$ and $\|y - \sum_{j=1}^{p}\frac{m_j}{K}\tilde{h}_j\|^2 \leq \frac{b^2}{K}$.*

This follows immediately from the proof of Lemma 11, choosing $m_j$ to be the coefficients of the $\tilde{h}_j$'s such that $y_K = \sum_j \frac{m_j}{K}\tilde{h}_j$.

The above corollary means that counting the number of $p$-tuple non-negative integers $m_1, \ldots, m_p$ gives us a covering of the set that $y$ belongs to. In the case of Lemma 11, this set is the convex hull of $\{\tilde{h}_j\}_j$.

Before we can go further, we need to generalize the argument from the positive orthant of the $\ell_1$ ball to handle any coefficients that are in the whole unit-length $\ell_1$-ball. This is what the following lemma accomplishes.

**Lemma 13** *Let $\max_{j=1,\ldots,p}\|\tilde{h}_j\|$ be less than or equal to some constant $b$. For any $y = \sum_{j=1}^{p}\tilde{\beta}_j\tilde{h}_j$ such that $\|\tilde{\beta}\|_1 \leq 1$, given a positive integer $K$, we can find a $y_K$ such that*

$$\|y - y_K\|_2^2 \leq \frac{b^2}{K}$$

*where $y_K = \sum_{j=1}^{p}\frac{k_j}{K}\tilde{h}_j$ is a combination of $\{\tilde{h}_j\}$ with integers $k_1, \ldots, k_p$ such that $\sum_{j=1}^{p}|k_j| \leq K$.*

**Proof** Lemma 11 cannot be applied directly since the $\{\tilde{\beta}_j\}_j$ can be negative. We rewrite $y$ or equivalently $\sum_{j=1}^p \tilde{\beta}_j \tilde{h}_j$ as

$$y = \sum_{j=1}^{p} |\tilde{\beta}_j| \text{sign}(\tilde{\beta}_j) \tilde{h}_j.$$

Thus $y$ lies in the convex combination of $\{\text{sign}(\tilde{\beta}_j)\tilde{h}_j\}_j$. Note that this step makes the convex hull depend on the $y$ or $\{\tilde{\beta}_j\}_j$ we start with. Nonetheless, we know by substituting $\{\text{sign}(\tilde{\beta}_j)\tilde{h}_j\}_j$ for $\{\tilde{h}_j\}_j$ in the statement of Lemma 11 and Corollary 12 that

1. we can find $y_K$, or equivalently
2. we can find non-negative integers $m_1, ..., m_p$ with $\sum_{j=1}^p m_j \leq K$,

such that $\|y - y_K\|_2^2 \leq \frac{b^2}{K}$ where $y_K = \sum_{j=1}^p \frac{m_j}{K} \text{sign}(\tilde{\beta}_j)\tilde{h}_j$ holds. This implies there exist integers $k_1, ..., k_p$ such that $y_K = \sum_{j=1}^p \frac{k_j}{K}\tilde{h}_j$ where $\sum_{j=1}^p |k_j| \leq K$. We simply let $k_j = m_j \text{sign}(\tilde{\beta}_j)$. Thus, we absorbed the signs of the $\tilde{\beta}_j$'s, and the coefficients no longer need to be nonnegative.

In other words, we have shown that if a particular $y_K$ is in the convex hull of points $\{\text{sign}(\tilde{\beta}_j)\tilde{h}_j\}_j$, then the same $y_K$ is a linear combination of $\{\tilde{h}_j\}_j$ where the coefficients of the combination $k_1/K, ..., k_p/K$ obey $\sum_{j=1}^p |k_j| \leq K$. This concludes the proof. ■

We now want to answer the question of whether the $k_1/K, ..., k_p/K$ can obey (related) linear constraints if the original $\{\tilde{\beta}_j\}_j$ did so. These constraints on the $\{\tilde{\beta}_j\}_j$'s are the ones coming from constraints on the operational cost. In other words, we want to know that our (discretized) approximation of $y$ also obeys a constraint coming from the operational cost.

Let $\{\tilde{\beta}_j\}_j$ satisfy the linear constraints within the definition of $\mathcal{B}$, in addition to satisfying $\|\tilde{\beta}\|_1 \leq 1$:

$$\sum_{j=1}^{p} \tilde{c}_{j\nu}\tilde{\beta}_j + \delta_\nu \leq 1, \text{ for fixed } \delta_\nu > 0, \nu = 1, ..., V.$$

We now want that for large enough $K$, the $p$-tuple $k_1/K, ..., k_p/K$ also meets certain related linear constraints.

We will make use of the matrix $\tilde{X}$, defined before Theorem 6. It has the elements of the scaled set $\{\tilde{h}_j\}_j$ as its columns: $\tilde{X} := [\tilde{h}_1 \ \ldots \ \tilde{h}_p]$.

**Lemma 14** *Take any $y = \sum_{j=1}^p \tilde{\beta}_j \tilde{h}_j$, and any $y_K = \sum_{j=1}^p \frac{k_j}{K}\tilde{h}_j$, with:*

$$\sum_{j=1}^{p} \tilde{c}_{j\nu}\tilde{\beta}_j + \delta_\nu \leq 1, \textit{ for fixed } \delta_\nu > 0, \nu = 1, ..., V \textit{ where } \|\tilde{\beta}\|_1 \leq 1$$

*and $\|y - y_K\|_2^2 \leq b^2/K$. Whenever*

$$K \geq \frac{b^2}{\left[\min_{\nu=1,...,V} \frac{\delta_\nu}{\sum_{j=1}^p |\tilde{c}_{j\nu}|}\right]^2 \lambda_{\min}(\tilde{X}^T\tilde{X})},$$

*then the following linear constraints on $k_1/K, ..., k_p/K$ hold:*

$$\sum_{j=1}^{p} \tilde{c}_{j\nu} \frac{k_j}{K} \leq 1, \ \nu = 1, ..., V.$$

This lemma states that as long as the discretization is fine enough, our approximation $y_K$ obeys similar operational cost constraints to $y$.

**Proof**

Let $\kappa := [k_1/K \ \dots \ k_p/K]^T$. Using the definition of $\tilde{X}$,

$$\begin{aligned} \frac{b^2}{K} &\geq \|y - y_K\|_2^2 = \|\tilde{X}\tilde{\beta} - \tilde{X}\kappa\|_2^2 = \|\tilde{X}(\tilde{\beta} - \kappa)\|_2^2 \\ &= (\tilde{\beta} - \kappa)^T \tilde{X}^T \tilde{X} (\tilde{\beta} - \kappa) \overset{(*)}{\geq} \lambda_{\min}(\tilde{X}^T \tilde{X}) \|\tilde{\beta} - \kappa\|_2^2. \end{aligned} \tag{21}$$

In $(*)$, we used the fact that for a positive (semi-)definite matrix $M$ and for every non-zero vector $z$, $z^T M z \geq \lambda_{\min}(M) z^T I z$. (If $\tilde{\beta} = \kappa$, we are done since $\kappa$ will obey the constraints $\tilde{\beta}$ obeys.) Also, for any $z$, in each coordinate $j$, $|z_j| \leq \max_{j=1,...,p} |z_j| = \|z\|_\infty \leq \|z\|_2$. Combining this with (21), we have:

$$\left| \tilde{\beta}_j - \frac{k_j}{K} \right| \leq \|\tilde{\beta} - \kappa\|_2 \leq \frac{b}{\sqrt{K \lambda_{\min}(\tilde{X}^T \tilde{X})}}.$$

This implies that $\kappa$ itself component-wise satisfies

$$\tilde{\beta}_j - A \leq \frac{k_j}{K} \leq \tilde{\beta}_j + A \text{ where } A := \frac{b}{\sqrt{K \lambda_{\min}(\tilde{X}^T \tilde{X})}}.$$

So far we know that for all $\nu = 1, ..., V$, $\sum_{j=1}^{p} \tilde{c}_{j\nu} \tilde{\beta}_j + \delta_\nu \leq 1$, with $\delta_\nu > 0$, and each coordinate $k_j/K$ within $\kappa$ varies from $\tilde{\beta}_j$ by at most an amount $A$. We would like to establish that the linear constraints $\sum_{j=1}^{p} \tilde{c}_{j\nu} \frac{k_j}{K} \leq 1, \ \nu = 1, ..., V$; always hold for such a $\kappa$. For each constraint $\nu$, substituting the extremal values of $k_j$ according to the sign of $\tilde{c}_{j\nu}$, we get the following upper bound:

$$\sum_{j=1}^{p} \tilde{c}_{j\nu} \frac{k_j}{K} \leq \sum_{\tilde{c}_{j\nu} > 0} \tilde{c}_{j\nu} (\tilde{\beta}_j + A) + \sum_{\tilde{c}_{j\nu} < 0} \tilde{c}_{j\nu} (\tilde{\beta}_j - A) = \sum_{j=1}^{p} \tilde{c}_{j\nu} \tilde{\beta}_j + A \sum_{j=1}^{p} |\tilde{c}_{j\nu}|.$$

This sum $\sum_{j=1}^{p} \tilde{c}_{j\nu} \tilde{\beta}_j + A \sum_{j=1}^{p} |\tilde{c}_{j\nu}|$ is less than or equal to 1 iff $A \sum_{j=1}^{p} |\tilde{c}_{j\nu}| \leq \delta_\nu$.

Thus we would like $A \leq \frac{\delta_\nu}{\sum_{j=1}^{p} |\tilde{c}_{j\nu}|}$ for all $\nu = 1, ..., V$. That is,

$$\begin{aligned} &\frac{b}{\sqrt{K \lambda_{\min}(\tilde{X}^T \tilde{X})}} = A \leq \min_{\nu=1,...,V} \frac{\delta_\nu}{\sum_{j=1}^{p} |\tilde{c}_{j\nu}|} \\ \Leftrightarrow \ \ &K \geq \frac{b^2}{\left[ \min_{\nu=1,...,V} \frac{\delta_\nu}{\sum_{j=1}^{p} |\tilde{c}_{j\nu}|} \right]^2 \lambda_{\min}(\tilde{X}^T \tilde{X})}. \end{aligned}$$

■

We now proceed with the proof of our main result of this section. The result involves covering numbers, where the cover for the set will be the vectors with discretized coefficients that we have been working with in the lemmas above.

**Proof** *(of Theorem 6)*

Recall that

- the matrix $X$ is defined as $[h_1 \ ... \ h_p]$;
- the scaled versions of vector $\{h_j\}_j$ are $\tilde{h}_j = \frac{n^{1/r} X_b B_b}{\|h_j\|_r} h_j$ for $j = 1, ..., p$;
- the scaled versions of coefficients $\{\beta_j\}_j$ are $\tilde{\beta}_j = \frac{\|h_j\|_r}{n^{1/r} X_b B_b} \beta_j$ for $j = 1, ..., p$; and
- any vector $y = X\beta = \sum_{j=1}^p \beta_j h_j$ can be rewritten as $\sum_{j=1}^p \tilde{\beta}_j \tilde{h}_j$.

We will prove three technical facts leading up to the result.

**Fact 1.** If $\|\beta\|_q \leq B_b$, then $\|\tilde{\beta}\|_1 \leq 1$.

Because $1/r + 1/q = 1$, by Hölder's inequality we have:

$$\begin{aligned} \sum_{j=1}^p |\tilde{\beta}_j| =& \frac{1}{n^{1/r} B_b X_b} \sum_{j=1}^p \|h_j\|_r |\beta_j| \\ \leq& \frac{1}{n^{1/r} B_b X_b} \left( \sum_{j=1}^p \|h_j\|_r^r \right)^{1/r} \left( \sum_{j=1}^p |\beta^j|^q \right)^{1/q}. \end{aligned} \tag{22}$$

To bound the above notice that in our notation, $(h_j)_i = (x_i)_j$. That is, the $i^{th}$ component of feature vector $h_j$, that is, $(h_j)_i$ is also the $j^{\text{th}}$ component of example $x_i$. Thus,

$$\begin{aligned} \left( \sum_{j=1}^p \|h_j\|_r^r \right)^{1/r} &= \left( \sum_{j=1}^p \sum_{i=1}^n ((h_j)_i)^r \right)^{1/r} = \left( \sum_{i=1}^n \sum_{j=1}^p ((h_j)_i)^r \right)^{1/r} \\ &= \left( \sum_{i=1}^n \|x_i\|_r^r \right)^{1/r} \leq (n X_b^r)^{1/r} = n^{1/r} X_b. \end{aligned}$$

Plugging this into (22), and using the fact that $\|\beta\|_q \leq B_b$, we have

$$\sum_{j=1}^p |\tilde{\beta}_j| \leq \frac{1}{n^{1/r} B_b X_b} n^{1/r} X_b B_b = 1,$$

that is, $\|\tilde{\beta}\|_1 \leq 1$.

**Fact 2.** Corresponding to the set of linear constraints on $\beta$:

$$\sum_{j=1}^p c_{j\nu} \beta_j + \delta_\nu \leq 1, \delta_\nu > 0, \nu = 1, ..., V,$$

there is a set of linear constraints on $\tilde{\beta}_j$, namely $\sum_{j=1}^{p} \tilde{c}_{j\nu}\tilde{\beta}_j + \delta_\nu \leq 1, \nu = 1,...,V$.

Recall that $\beta \in \mathcal{B}$ also means that $\sum_{j=1}^{p} c_{j\nu}\beta_j + \delta_\nu \leq 1$ for some $\delta_\nu > 0$ for all $\nu = 1,...,V$. Thus, for all $\nu = 1,...,V$:

$$\begin{aligned} & \sum_{j=1}^{p} c_{j\nu}\beta_j + \delta_\nu \leq 1 \\ \Leftrightarrow \quad & \sum_{j=1}^{p} c_{j\nu}\left(\frac{n^{1/r}X_b B_b}{\|h_j\|_r}\frac{\|h_j\|_r}{n^{1/r}X_b B_b}\right)\beta_j + \delta_\nu \leq 1 \\ \Leftrightarrow \quad & \sum_{j=1}^{p} \tilde{c}_{j\nu}\tilde{\beta}_j + \delta_\nu \leq 1 \end{aligned}$$

which is the set of corresponding linear constraints on $\{\tilde{\beta}_j\}_j$ we want.

**Fact 3.** $\forall j = 1,...,p,\ \|\tilde{h}_j\|_2 \leq n^{1/2}X_b B_b$.

Jensen's inequality implies that for any vector $z$ in $\mathbb{R}^n$, and for any $r \geq 2$, it is true that $\frac{1}{n^{1/2}}\|z\|_2 \leq \frac{1}{n^{1/r}}\|z\|_r$. Using this for our particular vector $\tilde{h}_j$ and our given $r$, we get

$$\|\tilde{h}_j\|_2 \leq \|\tilde{h}_j\|_r n^{1/2}\frac{1}{n^{1/r}}.$$

But we know

$$\|\tilde{h}_j\|_r = \left\|\frac{n^{1/r}X_b B_b}{\|h_j\|_r}h_j\right\|_r = \frac{n^{1/r}X_b B_b}{\|h_j\|_r}\|h_j\|_r = n^{1/r}X_b B_b.$$

Thus, we have $\|\tilde{h}_j\|_2 \leq n^{1/2}X_b B_b$ for each $j$, and thus, $\max_{j=1,...,p}\|\tilde{h}_j\|_2 \leq n^{1/2}X_b B_b$.

With those three facts established, we can proceed with the proof of Theorem 6. Facts 1 and 2 show that the requirements on $\tilde{\beta}$ for Lemma 13 and Lemma 14 are satisfied. Fact 3 shows that the requirement on $\{\tilde{h}_j\}_j$ for Lemma 13 is satisfied with constant $b$ being set to $n^{1/2}X_b B_b$. Since the requirements on $\{\tilde{h}_j\}_j$ and $\{\tilde{\beta}_j\}_j$ are satisfied, we want to choose the right value of positive integer $K$ such that Lemma 14 is satisfied and also we would like the squared distance between $y$ and $y_K$ to be less than $n\varepsilon^2$. To do this, we pick $K$ to be the bigger of the two quantities: $X_b^2 B_b^2/\varepsilon^2$ and that given in Lemma 14. That is,

$$K = \left\lceil \max\left\{ \frac{X_b^2 B_b^2}{\varepsilon^2}, \frac{nX_b^2 B_b^2}{\left[\min_{\nu=1,...,V}\frac{\delta_\nu}{\sum_{j=1}^{p}|\tilde{c}_{j\nu}|}\right]^2 \lambda_{\min}(\tilde{X}^T\tilde{X})} \right\} \right\rceil. \tag{23}$$

This will force our discretization for the cover to be sufficiently fine that things will work out: we will be able to count the number of cover points in our finite set, and that will be our covering number.

To summarize, with this choice, for any $y \in \mathcal{F}_{|S}$, we can find integers $k_1,...,k_p$ such that the following hold simultaneously:

*a.* (It gives a valid discretization of $y$.) $\sum_{i=1}^{p}|k_i| \leq K$,

*b*. (It gives a good approximation to $y$.) The approximation $y_K = \sum_{j=1}^{p} \frac{k_j}{K} \tilde{h}_j$ is $\varepsilon\sqrt{n}$ close to $y = \sum_{j=1}^{p} \tilde{\beta}_j \tilde{h}_j$. That is,

$$\|y - y_K\|_2^2 \leq \frac{nX_b^2 B_b^2}{K} \leq n\varepsilon^2, \text{and}$$

*c*. (It obeys operational cost constraints.) $\sum_{j=1}^{p} \tilde{c}_{j\nu} \frac{k_j}{K} \leq 1,\ \nu = 1, ..., V$.

In the above, the existence of $k_1, ..., k_p$ satisfying $(a)$ and $(b)$ comes from Lemma 13 where we have also used $K$ satisfying $K \geq X_b^2 B_b^2/\varepsilon^2 \geq 1$. Lemma 14 along with the choice of $K$ from (23) guarantees that $(c)$ holds as well for this choice of $k_1, ..., k_p$.

Thus, by $(b)$, any $y \in \mathcal{F}_{|S}$ is within $\varepsilon\sqrt{n}$ in $\ell_2$ distance of at least one of the vectors with coefficients $k_1/K, ..., k_p/K$. Therefore counting the number of $p$-tuple integers $k_1, ..., k_p$ such that $(a)$ and $(c)$ hold, or equivalently the number of solutions to (15), gives a bound on the covering number, which is $|P_c^K|$. That is,

$$N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) \leq |P_c^K|.$$

If we did not have any linear constraints, we would have the following bound,

$$N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) \leq |P^{K_0}|,$$

where $K_0 := \left\lceil \frac{X_b^2 B_b^2}{\varepsilon^2} \right\rceil$ by using Lemma 13 and very similar arguments as above.

In addition, when $\varepsilon \geq X_b B_b$, the covering number is exactly equal to 1 since we can cover the set $\mathcal{F}_{|S}$ by a closed ball of radius $\sqrt{n} X_b B_b$.

Thus we modify our upper bound by taking the minimum of the two quantities $|P^{K_0}|$ and $|P_c^K|$ appropriately to get the result:

$$N(\sqrt{n}\varepsilon, \mathcal{F}_{|S}, \|\cdot\|_2) \leq \begin{cases} \min\{|P^{K_0}|, |P_c^K|\} & \text{if } \varepsilon < X_b B_b \\ 1 & \text{otherwise.} \end{cases}$$

■

Since Theorem 6 suggests that $|P_c^K|$ may be an important quantity for the learning process, we discuss how to compute it. We assume that $\tilde{c}_{j\nu}$ are rationals for all $j = 1, .., p, \nu = 1, ..., V$, so that we can multiply each of the $V$ constraints describing $P_c^K$ by the corresponding gcd of the $p$ denominators. This is without loss of generality because the rationals are dense in the reals. This ensures that all the constraints describing polyhedron $P_c^K$ have integer coefficients. Once this is achieved, we can run Barvinok's algorithm (using for example, Lattice Point Enumeration, see De Loera, 2005, and references therein) that counts integer points inside polyhedra and runs in polynomial time for fixed dimension (which is $p$ here). Using the output of this algorithm within our generalization bound will yield a much tighter bound than in previous works (for example, the bound in Zhang, 2002, Theorem 3), especially when $(r, q) = (\infty, 1)$; this is true simply because we are counting more carefully. Note that counting integer points in polyhedrons is a fundamental question in a variety of fields including number theory, discrete optimization, combinatorics to name a few, and making an explicit connection to bounds on the covering number for linear function classes can potentially open doors for better sample complexity bounds.

## 6. Discussion and Conclusion

The perspective taken in this work contrasts with traditional decision analysis and predictive modeling; in these fields, a single decision is often the only end goal. Our goal involves exploring how predictive modeling influences decisions and their costs. Unlike traditional predictive modeling, our regularization terms involve optimization problems, and are not the usual vector norms.

The simultaneous process serves as a way to understand uncertainty in decision-making, and can be directly applied to real problems. We centered our discussion and demonstrations around three questions, namely: "What is a reasonable amount to allocate for this task so we can react best to whatever nature brings?" (answered in Section 3), "Can we produce a reasonable probabilistic model, supported by data, where we might expect to pay a specific amount?" (answered in Section 3), and "Can our intuition about how much it will cost to solve a problem help us produce a better probabilistic model?" (answered in Section 5). The first two were answered by exploring how optimistic and pessimistic views can influence the probabilistic models and the operational cost range. Given the range of reasonable costs, we could allocate resources effectively for whatever nature brings. Also given a specific cost value, we could pick a corresponding probabilistic model and verify that it can be supported by data. The third question was comprehensively answered in Section 5 by evaluating how intuition about the operational cost can restrict the probabilistic model space and in turn lead to better sample complexity if the intuition is correct.

These are questions that are not handled in a natural way by current paradigms. Answering these three questions are not the only uses for the simultaneous process. For instance, domain experts could use the simultaneous process to explore the space of probabilistic models and policies, and then simply pick the policy among these that most agrees with their intuition. Or, they could use the method to refine the probabilistic model, in order to exclude solutions that the simultaneous process found that did not agree with their intuition.

The simultaneous process is useful in cases where there are many potentially good probabilistic models, yielding a large number of (optimal-response) policies. This happens when the training data are scarce, or the dimensionality of the problem is large compared to the sample size, and the operational cost is not smooth. These conditions are not difficult to satisfy, and do occur commonly. For instance, data can be scarce (relative to the number of features) when they are expensive to collect, or when each each instance represents a real-world entity where few exist; for instance, each example might be a product, customer, purchase record, or historic event. Operational cost calculations commonly involve discrete optimization; there can be many scheduling, knapsack, routing, constraint-satisfaction, facility location, and matching problems, well beyond what we considered in our simple examples. The simultaneous process can be used in cases where the optimization problem is difficult enough that sampling the posterior of Bayesian models, with computing the policy at each round, is not feasible.

We end the paper by discussing the applicability of our policy-oriented estimation strategy in the real world. Prediction is the end goal for machine learning problems in vision, image processing and biology, and in other scientific domains, but there are many domains where the learning algorithm is used to make recommendations for a subsequent task. We showed applications in Section 3 but it is not hard to find applications in other domains, where using either the traditional sequential process, decision theory, or robust optimization may not suffice. Here are some other potential domains:

- Internet advertising, where the goal of the advertising platform is to choose which ad to show a customer. For each customer and advertiser, there is an uncertain estimate of the probability

that the customer will click the ad from that advertiser. These estimates determine which ad will be shown next, which is a discrete decision (Muthukrishnan et al., 2007).

- Portfolio management, where we allocate our budget among $n$ risky assets with uncertain returns, and each asset has a different cost associated with the investment (Konno and Yamazaki, 1991).

- Maintenance applications (in addition to the ML&TRP Tulabandhula et al., 2011), where we estimate probabilities of failure for each piece of equipment, and create a policy for repairing, inspecting, or replacing the equipment. Certain repairs are more expensive than others, so the costs of various policy decisions could potentially change steeply as the probability model changes.

- Traffic flows on transportation networks, where the problem can be that of load balancing based on resource constraints and forecasted demands (Koulakezian et al., 2012).

- Policy decisions based on dynamical system simulations, for instance, climate policy, where a politician wants to understand the uncertainty in policy decisions based on the results of a large-scale simulation. If the simulation cannot be computed for all initial values, its result can be estimated using a machine learning algorithm (Barton et al., 2010).

- Pharmaceutical companies choosing a subset of possible drug targets to test, where the drugs are predicted to be effective, and cannot be overly expensive to produce (Yu et al., 2012). This might be similar in many ways to the real-estate purchasing problem discussed in Section 3.

- Machine task scheduling on multi-core processors, where we need to allocate processors to various jobs during a large computation. This could be very similar to the problem of scheduling with constraints addressed in Section 3. If we optimistically estimate the amount of time each job takes, we will hopefully free up processors on time so they can be ready for the next part of the computation.

We believe the simultaneous process will open the door for other methods dealing with the interaction of machine learning and decision-making that fall outside the realm of the usual paradigms.

## Acknowledgments

Funding for this project comes in part from a Fulbright Science and Technology Fellowship, an award from the Solomon Buchsbaum Research Fund, and NSF grant IIS-1053407.

## References

Sivan Aldor-Noiman, Paul D. Feigin, and Avishai Mandelbaum. Workload forecasting for a call center: Methodology and a case study. *The Annals of Applied Statistics*, 3(4):1403–1447, 2009.

Martin Anthony and Peter L. Bartlett. *Neural network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

Kevin Bache and Moshe Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *Information Theory, IEEE Transactions on*, 39(3):930–945, 1993.

Peter L. Bartlett and Shahar Mendelson. Gaussian and Rademacher complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Russell R. Barton, Barry L. Nelson, and Wei Xie. A framework for input uncertainty analysis. In *Winter Simulation Conference*, pages 1189–1198. WSC, 2010.

John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. Springer Verlag, 1997.

Pierre Bonami, Lorenz T. Biegler, Andrew R. Conn, Gérard Cornuéjols, Ignacio E. Grossmann, Carl D. Laird, Jon Lee, Andrea Lodi, François Margot, Nicolas W. Sawaya, and Andreas Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

Olivier Bousquet. New approaches to statistical learning theory. *Annals of the Institute of Statistical Mathematics*, 55(2):371–389, 2003.

Lawrence D. Brown, Ren Zhang, and Linda Zhao. Root-unroot methods for nonparametric density estimation and poisson random-effects models. *Department of Statistics University of Pennsylvania, Tech. Rep*, 2001.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

Jeśus A. De Loera. The many aspects of counting lattice points in polytopes. *Mathematische Semesterberichte*, 52(2):175–195, 2005.

Simon French. *Decision Theory: An Introduction to the Mathematics of Rationality*. Halsted Press, 1986.

Sven Ove Hansson. *Decision Theory: A Brief Introduction*. Online manuscript. Department of Philosophy and the History of Technology, Royal Institute of Technology, Stockholm, 1994.

Yaochu Jin. *Multi-Objective Machine Learning, In Studies in Computational Intelligence*, volume 16. Springer, 2006.

Lee K. Jones. A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *The Annals of Statistics*, 20(1): 608–613, 1992.

Andrey Nikolaevich Kolmogorov and Vladimir Mikhailovich Tikhomirov. ε-entropy and ε-capacity of sets in function spaces. *Uspekhi Matematicheskikh Nauk*, 14(2):3–86, 1959.

Vladimir Koltchinskii and Dmitriy Panchenko. Complexities of convex combinations and bounding the generalization error in classification. *The Annals of Statistics*, 33(4):1455–1496, 2005.

Hiroshi Konno and Hiroaki Yamazaki. Mean-absolute deviation portfolio optimization model and its applications to Tokyo stock market. *Management Science*, pages 519–531, 1991.

Agop Koulakezian, Hazem M. Soliman, Tang Tang, and Alberto Leon-Garcia. Robust traffic assignment in transportation networks using network criticality. In *Proceedings of 2012 IEEE 76th Vehicular Technology Conference*, 2012.

S. Muthukrishnan, Martin Pal, and Zoya Svitkina. Stochastic models for budget optimization in search-based advertising. *Internet and Network Economics*, pages 131–142, 2007.

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

David Pollard. *Convergence of Stochastic Processes*. Springer, 1984.

Cynthia Rudin and Robert E. Schapire. Margin-based ranking and an equivalence between AdaBoost and RankBoost. *The Journal of Machine Learning Research*, 10:2193–2232, 2009.

Cynthia Rudin, Rebecca Passonneau, Axinia Radeva, Haimonti Dutta, Steve Ierome, and Delfina Isaac. A process for predicting manhole events in Manhattan. *Machine Learning*, 80:1–31, 2010.

Cynthia Rudin, Rebecca Passonneau, Axinia Radeva, Steve Ierome, and Delfina Isaac. 21st-century data miners meet 19th-century electrical cables. *IEEE Computer*, 44(6):103–105, June 2011.

Cynthia Rudin, David Waltz, Roger N. Anderson, Albert Boulanger, Ansaf Salleb-Aouissi, Maggie Chow, Haimonti Dutta, Philip Gross, Bert Huang, Steve Ierome, Delfina Isaac, Arthur Kressner, Rebecca J. Passonneau, Axinia Radeva, and Leon Wu. Machine learning for the New York City power grid. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):328–345, February 2012.

Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, pages 1651–1686, 1998.

Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171–176, 1958.

Michel Talagrand. *The Generic Chaining*. Springer, 2005.

Theja Tulabandhula and Cynthia Rudin. Machine learning with operational costs. In *Proceedings of the International Symposium on Artificial Intelligence and Mathematics*, 2012.

Theja Tulabandhula, Cynthia Rudin, and Patrick Jaillet. The machine learning and traveling repairman problem. In Ronen I. Brafman, Fred S. Roberts, and Alexis Tsoukiàs, editors, *ADT*, volume 6992 of *Lecture Notes in Computer Science*, pages 262–276. Springer, 2011.

Ian Urbina. Mandatory safety rules are proposed for electric utilities. *New York Times*, 2004. August 21, Late Edition, Section B, Column 3, Metropolitan Desk, Page 2.

Robert J. Vanderbei. *Linear Programming: Foundations and Extensions, Third Edition*. Springer, 2008.

Vladimir Naumovich Vapnik. *Statistical Learning Theory*, volume 2. Wiley New York, 1998.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10:1485–1510, December 2009.

Hua Yu, Jianxin Chen, Xue Xu, Yan Li, Huihui Zhao, Yupeng Fang, Xiuxiu Li, Wei Zhou, Wei Wang, and Yonghua Wang. A systematic prediction of multiple drug-target interactions from chemical, genomic, and pharmacological data. *PLoS ONE*, 5(7), 2012.

Tong Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2:527–550, 2002.

Xiaojin Zhu. Semi-supervised learning literature survey. Technical report, Computer Sciences TR 1530, University of Wisconsin Madison, December 2007.

Journal of Machine Learning Research 14 (2013) 2029-2066 Submitted 7/11; Revised 1/13; Published 7/13

# Approximating the Permanent with Fractional Belief Propagation

**Michael Chertkov** CHERTKOV@LANL.GOV
*Theory Division & Center for Nonlinear Studies*
*Los Alamos National Laboratory*
*Los Alamos, NM 87545, USA*

**Adam B. Yedidia*** ADAMY@MIT.EDU
*Massachusetts Institute of Technology*
*77 Massachusetts Avenue*
*Cambridge, MA 02139, USA*

**Editor:** Tony Jebara

## Abstract

We discuss schemes for exact and approximate computations of permanents, and compare them with each other. Specifically, we analyze the belief propagation (BP) approach and its fractional belief propagation (FBP) generalization for computing the permanent of a non-negative matrix. Known bounds and Conjectures are verified in experiments, and some new theoretical relations, bounds and Conjectures are proposed. The fractional free energy (FFE) function is parameterized by a scalar parameter $\gamma \in [-1;1]$, where $\gamma = -1$ corresponds to the BP limit and $\gamma = 1$ corresponds to the exclusion principle (but ignoring perfect matching constraints) mean-field (MF) limit. FFE shows monotonicity and continuity with respect to $\gamma$. For every non-negative matrix, we define its special value $\gamma_* \in [-1;0]$ to be the $\gamma$ for which the minimum of the $\gamma$-parameterized FFE function is equal to the permanent of the matrix, where the lower and upper bounds of the $\gamma$-interval corresponds to respective bounds for the permanent. Our experimental analysis suggests that the distribution of $\gamma_*$ varies for different ensembles but $\gamma_*$ always lies within the $[-1;-1/2]$ interval. Moreover, for all ensembles considered, the behavior of $\gamma_*$ is highly distinctive, offering an empirical practical guidance for estimating permanents of non-negative matrices via the FFE approach.

**Keywords:** permanent, graphical models, belief propagation, exact and approximate algorithms, learning flows

## 1. Introduction

This work is motivated by computational challenges associated with learning stochastic flows from two consecutive snapshots/images of $n$ identical particles immersed in a flow (Chertkov et al., 2008; Chertkov et al., 2010). The task of learning consists in maximizing the permanent of an $n \times n$ matrix, with elements constructed of probabilities for a particle in the first image to correspond to a particle in the second image, over the low-dimensional parametrization of the reconstructed flow. The permanents in this enabling application are nothing but a weighted number of perfect matchings relating particles in the two images.

In this manuscript we continue the thread of Watanabe and Chertkov (2010) and focus on computations of positive permanents of non-negative matrices constructed from probabilities. The exact

*. Also at Theory Division & Center for Nonlinear Studies, Los Alamos National Laboratory, Los Alamos, NM 87545, USA.

©2013 Michael Chertkov and Adam B. Yedidia.

computation of the permanent is difficult, that is, it is a problem of likely exponential complexity, with the fastest known general Algorithm for computing the permanent of a full $n \times n$ matrix based on the formula from Ryser (1963) requiring $O(n2^n)$ operations. In fact, the task of computing the permanent of a non-negative matrix was one of the first problems established to be in the #-P complexity class, and the task is also complete in the class (Valiant, 1979).

Therefore, recent efforts have mainly focused on developing approximate Algorithms. Three independent developments, associated with the mathematics of strict bounds, Monte-Carlo sampling, and graphical models, contributed to this field.

The focus of the mathematics of permanent approach was on establishing rigorous lower and upper bounds for permanents. Many significant results in this line of research are related to the Conjecture of van der Waerden (1926) that the minimum of the permanent over doubly stochastic matrices is $n!/n^n$, and it is only attained when all entries of the matrix are $1/n$. The Conjecture remained open for over 50 years before Falikman (1981) and Egorychev (1981) proved it. Recently, Gurvits (2008) found an alternative, surprisingly short and elegant proof that also allowed for a number of unexpected extensions. See, for example, the discussion of Laurent and Schrijver (2010).

A very significant breakthrough in the Monte-Carlo sampling was achieved with the invention of the fully polynomial randomized Algorithmic schemes (fpras) for the permanent problem (Jerrum et al., 2004): the permanent is approximated in polynomial time, provably with high probability and within an arbitrarily small relative error. The complexity of the fpras of Jerrum et al. (2004) is $O(n^{11})$ in the general case. Even though the scaling was improved to $O(n^4 \log n)$ in the case of very dense matrices (Huber and Law, 2008), the approach is still impractical for the majority of realistic applications.

Belief propagation (BP) heuristics applied to permanent showed surprisingly good performance (Chertkov et al., 2008; Huang and Jebara, 2009; Chertkov et al., 2010). The BP family of Algorithms, originally introduced in the context of error-correction codes by Gallager (1963), artificial intelligence by Pearl (1988), and related to some early theoretical work in statistical physics by Bethe (1935), and Peierls (1936) on tree graphs, can generally be stated for any GM according to Yedidia et al. (2005). The exactness of the BP on any tree, that is, on a graph without loops, suggests that the Algorithm can be an efficient heuristic for evaluating the partition function, or for finding a maximum likelihood (ML) solution of the GM defined on sparse graphs. However, in the general loopy cases, one would normally not expect BP to work very well, making the heuristic results of Chertkov et al. (2008), Huang and Jebara (2009) and Chertkov et al. (2010) somehow surprising, even though not completely unexpected in view of the existence of polynomially efficient Algorithms for the ML version of the problem (Kuhn, 1955; Bertsekas, 1992), which were shown by Bayati et al. (2008) to be equivalent to an iterative Algorithm of the BP type. This raises questions about understanding the performance of BP. To address this challenge (Watanabe and Chertkov, 2010) established a theoretical link between the exact permanent and its BP approximation. The permanent of the original non-negative matrix was expressed as a product of terms, including the BP-estimate and another permanent of an auxiliary matrix, $\beta .* (1-\beta)$,[1] where $\beta$ is the doubly stochastic matrix of the marginal probabilities of the links between the particles in the two images (edges in the underlying GM) calculated using the BP approach. (See Theorem 3.) The exact relation of Watanabe and Chertkov (2010) followed from the general loop calculus technique of Chertkov and Chernyak (2006a,b), but it also allowed a simple direct derivation. Combining this

1. Here and below we will follow Matlab notations for the component-wise operations on matrices, such as $A .* B$ for the component-wise, Hadamard, product of the matrices $A$ and $B$.

exact relation with aforementioned results from the mathematics of permanents led to new lower and upper bounds for the original permanent. Moreover this link between the math side and the GM side gained a new level of prominence with the recent proof by Gurvits (2011) of the fact that the variational formulation of BP in terms of the Bethe free energy (BFE) function, discussed earlier by Chertkov et al. (2008), Chertkov et al. (2010) and Watanabe and Chertkov (2010), and shown to be convex by Vontobel (2013), gives a provable lower bound to the permanent. Remarkably, this proof of Gurvits was based on an inequality suggested earlier by Schrijver (1998) for the object naturally entering the exact, loop calculus based, BP formulas, $\mathrm{perm}(\beta.*(1-\beta))$.

This manuscript contributes two-fold, theoretically and experimentally, to the new synergy developing in the field. On the theory side, we generalize the BP approach to approximately computing permanents, suggesting replacing the BFE function by its FFE generalization in the general spirit of Wiegerinck and Heskes (2003), differing from the BFE function of Yedidia et al. (2005) in the entropy term, and then derive new exact relations between the original permanent and the results of the FFE-based approach (see Theorem 13). The new object, naturally appearing in the theory, is $\mathrm{perm}(\beta.*(1-\beta).^{-\gamma})$, where $\gamma\in[-1;1]$. The case of $\gamma=-1$ corresponds to BP and the case of $\gamma=1$ corresponds to the so-called exclusion principle (Fermi), but ignoring perfect matching constraints, mean field (MF) approximation discussed earlier by Chertkov et al. (2008). Using recent results from the "mathematics of permanents," in particular from Gurvits (2011), we show, that considered as an approximation, the FFE-based estimate of the permanent is a monotonic continuous function of the parameter $\gamma$ with $\gamma=-1$ and $\gamma=0$ setting, respectively, the lower bound (achievable on trees) and an upper bound. We also analyze existing and derive new lower and upper bounds. We adopt for our numerical experiments the so-called zero-suppressed binary decision diagrams (ZDDs) approach of Minato (1993) (see, for example, Knuth 2009), which outperforms Ryser's formula for realistic (sparsified) matrices, for exactly evaluating permanents, develop numerical schemes for efficiently evaluating the fractional generalizations of BP, test the aforementioned lower and upper bounds for different ensembles of matrices and study the special, matrix dependent, $\gamma_*$, which is defined to be the special $\gamma$ for which the FFE-based estimate is equal to the permanent of the matrix.[2]

The material in the manuscript is organized as follows: the technical introduction, stating the computation of the permanent as a GM, is explained in Section 2 and Appendix A. The BP-based optimization formulations, approximate methods, iterative Algorithms and related exact formulas are discussed in Section 3 and Appendices B, C, D, E. Section 4 is devoted to permanental inequalities, discussing the special values of $\gamma$ and Conjectures. Our numerical experiments are presented and discussed in Section 5 and Appendices F, G, H. We conclude and discuss the path forward in Section 6.

## 2. Technical Introduction

The permanent of a square matrix $p$, $p=(p_{ij}|i,j=1,\ldots,n)$, is defined as

$$\mathrm{perm}(p)=\sum_{s\in S_n}\prod_{i=1}^{n}p_{is(i)},$$

2. Note that a methodologically similar approach, of searching for the best/special FFE-based coefficient, was already discussed in the literature by Cseke and Heskes (2011) for a Gaussian BP example.

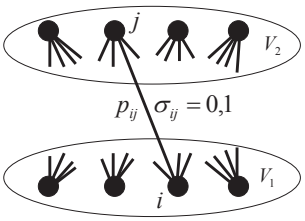



Figure 1: Illustration of graphical model for perfect matchings and permanent.

where $S_n$ is the set of all permutations of the set, $\{1,\ldots,n\}$. Here and below we will only discuss permanents of non-negative matrices, that is, with $\forall i,j=1,\ldots,n: p_{ij}\geq 0$. We also assume that $\mathrm{perm}(p)>0$.

An example of a physics problem, where computations of permanents are important, is given by particle tracking experiments and measurements techniques, of the type discussed in Chertkov et al. (2008) and Chertkov et al. (2010). In this case, an element of the matrix, $p=(p_{ij}|i,j=1,\ldots,n)$, is interpreted as an unnormalized probability that the particle labeled $i$ in the first image moves to the position labeled $j$ in the second image. In its most general formulation, the task of learning a low dimensional parametrization of the flow from two consecutive snapshots consists of maximizing the partition function $Z=\mathrm{perm}(p)$ over the "macroscopic" flow parameters affecting $p$. Computing the permanent for a given set of values of the parameters constitutes an important subtask, the one we are focusing on in this manuscript.

### 2.1 Computation of the Permanent as a Graphical Model Problem

The permanent of a matrix can be interpreted as the partition function $Z$ of a graphical model (GM) defined over a bipartite undirected graph, $\mathcal{G}=(\mathcal{V}=(\mathcal{V}_1,\mathcal{V}_2),\mathcal{E})$, where $\mathcal{V}_1$, $\mathcal{V}_2$ are of equal size, $|\mathcal{V}_1|=|\mathcal{V}_2|=n$, and $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{E}$ stand for the set of $n$ vertices/labels of particles in the first and second images and the set of edges (possible relations) between particles in the two images, respectively. The basic binary variables, $\sigma_{ij}=0,1$, are associated with the edges, while the perfect matchings are enforced via the constraints associated with vertexes, $\forall i\in\mathcal{V}_1:\quad \sum_{j\in V_2}\sigma_{ij}=1$ and $\forall j\in\mathcal{V}_2:\quad \sum_{i\in V_1}\sigma_{ij}=1$, as illustrated in Figure 1. A non-negative element of the matrix, $p_{ij}$, turns into the weight associated with the edge $(i,j)$. In summary, the GM relates the following probability to any of $n!$ perfect matchings, $\sigma$:

$$\rho(\sigma)=Z(p)^{-1}\prod_{(i,j)\in\mathcal{E}}(p_{ij})^{\sigma_{ij}}, \tag{1}$$

$$\sigma=\left(\sigma_{ij}=0,1\,\middle|\,(i,j)\in\mathcal{E};\ \forall i\in\mathcal{V}_1:\ \sum_{j\in\mathcal{V}_2}\sigma_{ij}=1;\forall j\in\mathcal{V}_2:\ \sum_{i\in\mathcal{V}_1}\sigma_{ij}=1\right),$$

$$Z(p)=\mathrm{perm}(p)=\sum_{\sigma}\prod_{(i,j)\in\mathcal{E}}(p_{ij})^{\sigma_{ij}}. \tag{2}$$

The GM formulation (1) also suggests a variational, Kullback-Leibler (KL) scheme for computing the permanent. The only minimum of the so-called exact free energy (FE) function,

$$F(b|p) = \sum_{\sigma} b(\sigma) \log \left( \frac{b(\sigma)}{\prod_{(i,j)\in\mathcal{E}} (p_{ij})^{\sigma_{ij}}} \right), \tag{3}$$

computed over $b(\sigma) \geq 0$ for all $\sigma$ under the normalization condition, $\sum_{\sigma} b(\sigma) = 1$, is achieved at $b(\sigma) = \rho(\sigma)$, and the value of the exact FE function at the minimum over $b(\sigma)$ is $-\log(Z(p))$. Here, the general and the optimal $b(\sigma)$ are interpreted as, respectively, the proxy and the probability of the perfect matching $\sigma$.

The relation between the problem of computing the permanent and the problem of finding the most probable (maximum) perfect matching is discussed in Appendix A.1.

### 2.2 Exact Methods for Computing Permanents

Computing the permanent of a matrix is a #-P hard problem, that is, it is a problem which most likely requires a number of operations exponential in the size of the matrix. In Appendix H, we experiment and compare the performance of the following two exact deterministic ways to evaluate permanents:

- A general method based on zero-suppressed binary decision diagrams (ZDDs), explained in more detail in Knuth (2009). See also detailed explanations below in Appendix G. As argued in Knuth (2009), the ZDD approach may be a very efficient practical tool for computing partition functions in general graphical models. This thesis was illustrated by Yedidia (2009) on the example of counting independent sets and kernels of graphs.

- A permanent-specific method based on Ryser's formula:

$$Z(p) = (-1)^n \sum_{S\subseteq\{1,\ldots,n\}} (-1)^{|S|} \prod_{i=1}^{n} \sum_{j\in S} p_{ij}.$$

  We use code from TheCodeProject for implementing the Ryser's formula.

Note that in most practical cases many entries of $p$ are very small and they do not affect the permanent of $p$ significantly. These entries do, however, take computational resources if accounted for in the Algorithm. To make computations efficient we sparsify the resulting matrix $p$, implementing the heuristic pruning technique explained in Appendix F.

We also verify some of our results against randomized computations of the permanent using the FPRAS from Jerrum et al. (2004), with a specific implementation from Chertkov et al. (2008).

## 3. Approximate Methods and Exact Relations

We perform an approximate computation of the permanent by following the general BFE approach of Yedidia et al. (2005) and the associated belief propagation/Bethe-Peierls (BP) Algorithm, discussed in detail for the case of permanents of non-negative matrices in Chertkov et al. (2008), Huang and Jebara (2009) and Chertkov et al. (2010). See also Appendix B reproducing the description of Chertkov et al. (2008) and Chertkov et al. (2010) and presented in this manuscript for convenience.

In our BP experiments we implement the Algorithm discussed by Chertkov et al. (2008) with a special type of initialization corresponding to the best perfect matching of $p$. We also generalize the BP scheme by modifying the entropy term in the BFE.

In the following Subsections we re-introduce the BFE approach, the related but different MF FE approach, and also consider a fractional FE approach generalizing and interpolating between Bethe/BP and MF approaches. Even though these optimization approaches and respective Algorithms can be thought of as approximating the permanent we will show that they also generate some exact relations for the permanent.

### 3.1 Belief Propagation/Bethe-Peierls Approach

Let us start by defining some useful notation.

**Definition 1 ($\beta$-polytope)** *Call the $\beta$-polytope of the non-negative matrix $p$ (or just $\beta$-polytope for short) the set of doubly stochastic non-negative matrices with elements corresponding to zero elements of $p$ equal to zero, that is, $\mathcal{B}_p = (\beta_{ij}|\forall i: \sum_{(i,j)\in\mathcal{E}}\beta_{ij}=1; \forall j: \sum_{(i,j)\in\mathcal{E}}\beta_{ij}=1; \forall(i,j)$ with $p_{ij}=0$ $\beta_{ij}=0$ holds). We say that $\beta$ lies in the interior of the $\beta$-polytope, $\beta\in\mathcal{B}_p^{(int)}$, if $\forall(i,j)$ with $p_{ij}\neq 0$ $\beta_{ij}\neq 0,1$ holds.*[3]

In English, the interior solution means that all elements of the doubly stochastic matrices $\beta$ are non-integer, under exception of the case when $p_{ij}=0$ and, respectively, $\beta_{ij}=0$.

**Definition 2 (Bethe free energy for the permanent)** *The following function of $\beta\in\mathcal{B}_p$*

$$F_{BP}(\beta|p)=\sum_{(i,j)}\left(\beta_{ij}\log(\beta_{ij}/p_{ij})-(1-\beta_{ij})\log(1-\beta_{ij})\right), \tag{4}$$

*conditioned to a given $p$, is called the Bethe free energy (BFE) or the belief-Propagation/Bethe-Peierls (BP) function (for the permanent).*[4]

To motivate the definition above let us briefly discuss the concept of the Bethe FE which was introduced in Yedidia et al. (2005) for the case of a general pair-wise interaction GM (with variables associated with vertices of the graph). Schematically, the logic extended to the case with variables associated with edges of the graph and leading to Equation (4) for the permanent is as follows. (See Watanabe and Chertkov 2010 for a detailed discussion.) Consider a GM with binary variables associated with edges of the graph. If the graph is a tree, then the following exact relation holds, $\rho(\sigma)=\prod_i\rho_i(\sigma_i)/\prod_{(i,j)}\rho_{ij}(\sigma_{ij})$, where $\sigma_i=(\sigma_{ij}=0,1|(i,j)\in\mathcal{E})$. Here, $\rho_i(\sigma_i)$ and $\rho_{ij}(\sigma_{ij})$ are marginal probabilities associated, respectively, with vertex $i$ and edge $(i,j)$ of the graph. Replacing the probabilities by their proxies/beliefs, $\rho(\sigma)\to b(\sigma)$, $\rho_i(\sigma_i)\to b_i(\sigma_i)$ and $\rho_{ij}(\sigma_{ij})\to b_{ij}(\sigma_{ij})$, substituting the ratio of probabilities expression for $b(\sigma)$ in the exact FE function (3), and accounting for relations between the marginal beliefs, one arrives at the general expression for the Bethe FE function. This expression for the Bethe FE function is exact on a tree only, and it is similar in spirit to the one introduced in Yedidia et al. (2005) as an approximation for GM on a graph with loops.

3. What we call here "interior" would be mathematically more accurate to call "relative interior".
4. In the following, and whenever Bethe, MF, or fractional FE are mentioned, we will drop the clarifying—for the permanent—as only permanents are discussed in this manuscript.

When the graph is bi-partite with an equal number of nodes in the two parts the BP replacement for $b(\sigma)$ becomes

$$b_{BP}(\sigma) = \frac{\prod_{(i,j)\in\mathcal{E}:\sigma_{ij}=1}\beta_{ij}}{\prod_{(i,j)\in\mathcal{E}:\sigma_{ij}=0}(1-\beta_{ij})}, \tag{5}$$

where $\beta_{ij} = b_{ij}(1)$ is the marginal belief corresponding to finding the edge $(i,j)$ in the matching. Then, substituting $b(\sigma)$ by $b_{BP}(\sigma)$ in Equation (3) results in the Bethe FE expression (4) for the perfect matchings (permanents). Note, that while the exact FE (3) is the sum of $O(n!)$ terms, there are only $O(n^2)$ terms in the Bethe FE (4).

According to the Loop Calculus approach of Chertkov and Chernyak (2006a,b), extended to the case of the permanent in Chertkov et al. (2008), Chertkov et al. (2010) and Watanabe and Chertkov (2010), the BP expression and the permanent are related to each other as follows:

**Theorem 3 (Permanent and BP)** *If the BP Equations following from minimization of the the BFE (4) over the doubly stochastic matrix $\beta$, that is,*

$$\forall(i,j): \quad (1-\beta_{ij})\beta_{ij} = \frac{p_{ij}}{u_i u^j}, \tag{6}$$

*where $\log(u_i)$ and $\log(u^j)$ are positive-valued Lagrangian multipliers conjugated to $\sum_{j\in\mathcal{V}_2}\beta_{ij}=1$ and $\sum_{i\in\mathcal{V}_1}\beta_{ij}=1$, respectively, have a solution in the interior of the $\beta$-polytope, $\beta\in\mathcal{B}_p^{(int)}$, then*

$$Z = Z_{BP}(p)\mathrm{perm}(\beta.*(1-\beta))\frac{1}{\prod_{i,j}(1-\beta_{ij})}, \tag{7}$$

*where $Z_{BP}(p) = -\log(F_{BP}(\beta|p))$.*

The proof of the Theorem 3 also appears in Appendix B.1. An iterative heuristic Algorithm solving BP Equations (6) for the doubly stochastic matrix $\beta$ efficiently is discussed in Appendix B.2.

Let us recall that the $(i,j)$ element of the doubly stochastic matrix $\beta$, $\beta_{ij}$, is interpreted as the proxy (approximation) to the marginal probability for the $(i,j)$ edge of the bipartite graph $\mathcal{G}$ to be in a perfect matching, that is, $\beta_{ij}$, should be thought of as an approximation for $\rho_{ij} = \sum_{\sigma:\sigma_{ij}=1}\rho(\sigma)$.

Note also that $\log(u_i)$ and $\log(u^j)$ in Equations (6) are the Lagrange multipliers related to the $2n$ double stochasticity (equality) constraints on $\beta$.

### 3.1.1 BP AS THE MINIMUM OF THE BETHE FREE ENERGY

**Definition 4 (Optimal Bethe free energy)** *We define optimal BFE, $F_{o-BP}(p)$, and respective counting factor, $Z_{o-BP}(p)$, according to*

$$-\log(Z_{o-BP}(p)) = F_{o-BP}(p) = \min_{\beta\in\mathcal{B}} F_{BP}(\beta|p), \tag{8}$$

*where $F_{BP}(\beta|p)$ is defined in Equation (4).*

Considered in the general spirit of Yedidia et al. (2005), $F_{o-BP}(p)$, just defined, should be understood as an approximation to $-\log(\mathrm{perm}(p))$. To derive Equation (8) one needs to replace $b(\sigma)$ by (5). See Watanabe and Chertkov (2010) for more details.

The relation between the optimization formulation (8) and the BP Equations (6) requires some clarifications stated below in terms of the following two Propositions.

**Proposition 5 (Partially resolved BP solutions)** *Any doubly stochastic matrix* $\beta$ *solving Equations (6) and lying on the boundary of the* $\beta$ *polytope, that is,* $\beta \in \mathcal{B}_p$ *but* $\beta \notin \mathcal{B}_p^{(int)}$*, can be reduced by permutations of rows and columns of* $\beta$ *(and p, respectively) to a block diagonal matrix, with one block consisting of* $0,1$ *elements only and corresponding to a partial perfect matching, and the other block having all elements strictly smaller than unity, and nonzero if the respective* $p_{ij} \neq 0$*. We call such a solution of the BP Equations (6) partially resolved solutions, emphasizing that a part of the solution forms a partial perfect matching, and any other perfect matching over this subset is excluded by the solution (in view of the probabilistic interpretation of* $\beta$*). A doubly stochastic matrix* $\beta$ *corresponding to a full perfect matching is called a fully resolved solution of the BP Equations (6).*

**Proof** This statement follows directly from the double stochasticity of $\beta$ and from the form of the BP Equations (6), and it was already discussed in Chertkov et al. (2008) and Watanabe and Chertkov (2010) for the fully resolved case. ■

**Proposition 6 (Optimal Bethe FE and BP equations)** *The optimal Bethe FE,* $F_{o-BP}(\beta)$ *over* $\beta \in \mathcal{B}_p$*, can only be achieved at a solution of the BP Equations (6), possibly with the Lagrange multipliers* $u_i$*,* $u^j$ *taking the value* $+\infty$*.*

**Proof** This statement is an immediate consequence of the fact that Proposition 5 is valid for any $p$, and so a continuous change in $p$ (capable of covering all possible achievable $p$) can only result in an interior solution for the doubly stochastic matrix $\beta$ merging into a vertex of the $\beta$-polytope, or emerging from the vertex (then respective Lagrangian multipliers take the value $+\infty$), but never reaching an edge of the polytope at any other location but a vertex. Therefore, we can exclude the possibility of achieving the minimum of the Bethe FE anywhere but at an interior solution, partially resolved solution or a fully resolved solution (corresponding to a perfect matching) of the BP equations. ■

Note, that an example where the minimum in Equation (8) is achieved at the boundary of the $\beta - polytope$ (in fact, at the most probable perfect matching corner of the polytope) was discussed in Watanabe and Chertkov (2010).

Another useful and related statement, made recently in Vontobel (2013), is

**Proposition 7 (Convexity of the Bethe FE)** *The Bethe FE (4) is a convex function of* $\beta \in \mathcal{B}_p$*.*

A few remarks are in order. First, the statement above is nontrivial as, considered naively, individual edge contributions in Equation (4) associated with the entropy term, $\beta_{ij}\log\beta_{ij} - (1-\beta_{ij})\log(1-\beta_{ij})$, are not convex for $\beta_{ij} > 1/2$, and the convexity is restored only due to the global (double stochasticity) condition. Second, the convexity means that if the optimal solution is not achieved at the boundary of $\mathcal{B}_p$, then either the solution is unique (general case) or the situation is degenerate and one finds a continuous family of solutions all giving the same value of the Bethe FE. The degeneracy means that $p$ should be fine tuned to get into the situation, and addition of almost any small (random) perturbation to $p$ would remove the degeneracy. To illustrate how the degeneracy may occur, consider an example of a $(2\times 2)$ matrix $p$ with all elements equal to each other. We first

observe that regardless of $p$ for $n=2$, the entropy contributions to the Bethe FE are identical to zero for any doubly stochastic $(2\times 2)$ matrix, $\sum_{(i,j)}^{i,j=1,2}(\beta_{ij}\log\beta_{ij}-(1-\beta_{ij})\log(1-\beta_{ij}))=0$. Moreover, the remaining, linear in $\beta$, contribution to the Bethe FE (which is also called the self-energy in physics) turns into a constant for the special choice of $p$. Thus one finds that in this degenerate $n=2$ case,

$$\beta=\begin{pmatrix}\alpha & 1-\alpha\\ 1-\alpha & \alpha\end{pmatrix},$$

with any $\alpha\in[0;1]$, is a solution of Equations (6) also achieving the minimum of the Bethe FE. Creating any asymmetry between the four components of the $(2\times 2)$ $p$ will remove the degeneracy, moving the solution of Equations (6) achieving the minimum of the Bethe FE to one of the two perfect matchings, corresponding to $\alpha=0$ and $\alpha=1$, respectively. It is clear that this special "double" degeneracy (first, cancellation of the entropy contribution, and then constancy of the self-energy term) will not appear at all if the doubly stochastic matrix $\beta$, solving Equations (6) in the $n>2$ case has more than two nonzero components in every row and column. Combined with Proposition 7, this observation translates into the following statement.

**Corollary 8 (Uniqueness of interior BP solution)** *If $n>2$ and an interior, $\beta\in\mathcal{B}_p^{(int)}$, solution of Equations (6) has more than two nonzero elements in every row and column, then the solution is unique*

In the following, discussing an interior BP solution, $\beta\in\mathcal{B}_p^{(int)}$ and aiming to focus only on the interesting/nontrivial cases, we will be assuming that $n>2$ and $p$ has more than two nonzero elements in every row and column.

### 3.2 Mean-Field Approach

**Definition 9 (Mean field free energy)** *For $\beta\in\mathcal{B}_p$, the MF FE is defined as*

$$F_{MF}(\beta|p)=\sum_{(i,j)}\left(\beta_{ij}\log(\beta_{ij}/p_{ij})+(1-\beta_{ij})\log(1-\beta_{ij})\right). \tag{9}$$

Let us precede discussion of the MF notion/approach by a historical and also motivational remark. MF is normally thought of as an approximation ignoring correlations between variables. Then, the joint distribution function of $\sigma$ is expressed in terms of the product of marginal distributions of its components. In our case of the perfect matching GM over the bi-partite graph, the MF approximation constitutes the following substitution for the exact beliefs,

$$b(\sigma)\to\prod_{(i,j)\in\mathcal{E}}b_{ij}(\sigma_{ij}), \tag{10}$$

into Equation (3). Making the substitution and relating the marginal edge beliefs to $\beta$ according to, $\forall(i,j)\in\mathcal{E}:\ b_{ij}(1)=\beta_{ij},\ b_{ij}(0)=1-\beta_{ij}$, one arrives at Equation (9). Because of how the perfect matching problem is defined, the two states of an individual variable, $\sigma_{ij}=0$ and $\sigma_{ij}=1$, are in the exclusion relation, and so one can also associate the special form of Equation (9) with the exclusion or Fermi- (for Fermi-statistics of physics) principle.

Direct examination of Equation (9) reveals that

**Proposition 10 (MF FE minimum is always in the interior)** *$F_{MF}(\beta|p)$ is strictly convex and its minimum is achieved at $\beta \in \mathcal{B}_p^{(int)}$.*

Looking for the interior minimum of Equation (9) one arrives at the following MF equations for the (only) stationary point of the MF FE function

$$\forall (i,j) \in \mathcal{E}: \quad \beta_{ij} = \frac{1}{1 + v_i v^j / p_{ij}},$$

where $\log(v_i)$ and $\log(v^j)$ are Lagrangian multipliers enforcing the conditions, $\sum_j \beta_{ij} = 1$ and $\sum_i \beta_{ij} = 1$, respectively. The equations can also be rewritten as

$$\forall (i,j): \quad \frac{\beta_{ij}}{1-\beta_{ij}} = \frac{p_{ij}}{v_i v^j}, \tag{11}$$

making comparison with the respective BP Equations (6) transparent. An efficient heuristic for solving the MF equations (11) is discussed in Appendix C.2.

Direct examination shows that (unlike in the BP case) $\beta$ with a single element equal to unity or zero (when the respective $p$ element is nonzero) cannot be a solution of the MF Equations (11) over doubly stochastic matrix $\beta$—fully consistently with the Proposition 10 above. Moreover, $-\log(Z_{o-MF}(p))$, defined as the minimum of the MF FE (9), is simply equal to $-\log(Z_{MF}(p)$, defined as $F_{MF}(\beta)$ evaluated at the (only) doubly stochastic matrix solution of Equation (11).

Note also that the MF function (9) cannot be considered as a variational proxy for the permanent, bounding its value from below. This is because the substitution on the right-hand side of Equation (10) does not respect the perfect matching constraints, assumed reinforced on the left-hand side of Equation (10). In particular, the probability distribution function on the right-hand side of Equation (10) allows two edges of the graph adjacent to the same vertex to be in the active, $\sigma_{ij} = 1$, state simultaneously. However, this state is obviously prohibited by the original probability distribution, on the left-hand side of Equation (10) defined only over $n!$ of states corresponding to the perfect matchings. As shown below in Section 4.2, the fact that the MF ignores the perfect matching constraints results in the estimate $Z_{MF}(p)$ upper bounding $\mathrm{perm}(p)$, contrary to what a standard MF (not violating any original constraints) would do.

Finally and most importantly (for the MF discussion of this manuscript), the MF approximation for the permanent, $Z_{MF}$, can be related to the permanent itself as follows:

**Theorem 11 (Permanent and MF)**

$$Z(p) = \mathrm{perm}(p) = Z_{MF}(p)\mathrm{perm}\left(\beta./(1-\beta)\right) \prod_{(i,j)\in\mathcal{E}} (1-\beta_{ij}), \tag{12}$$

*where $\beta$ is the only interior minimum of (9).*

The proof of this statement is given in Appendix C.1.

### 3.3 FFE-based Approach

Similarity between the exact BP expression (7) and the exact MF expression (12) suggests that the two formulas are the limiting instances of a more general relation. We define

**Definition 12 (Fractional free energy)** *For $\beta \in \mathcal{B}_p$, the FFE is defined as*

$$F_f^{(\gamma)}(\beta|p) = -\log(Z_f^{(\gamma)}(\beta|p)) = \sum_{(i,j)} (\beta_{ij}\log(\beta_{ij}/p_{ij}) + \gamma(1-\beta_{ij})\log(1-\beta_{ij}))\,. \tag{13}$$

Then, one finds that

**Theorem 13 (FFE-based representation for permanent)** *For any non-negative $p$ and doubly stochastic matrix $\beta$ which solves*

$$\forall(i,j): \quad \frac{\beta_{ij}}{(1-\beta_{ij})^{\gamma}} = \frac{p_{ij}}{w_i w^j}, \tag{14}$$

*for $\gamma \in [-1;1]$, and if the solution found is in the interior of the domain, that is, $\beta \in \mathcal{B}_p^{(int)}$, the following relation holds*

$$\mathrm{perm}(p) = Z_f^{(\gamma)}(\beta|p)\mathrm{perm}\left(\frac{\beta.}{(1-\beta).^{\gamma}}\right)\prod_{(i,j)}(1-\beta_{ij})^{\gamma}, \tag{15}$$

*where FFE was defined above in Equation (13) and $\log(w_i)$ and $\log(w^j)$ in Equation (14) are the Lagrangian multipliers enforcing the conditions, $\sum_j \beta_{ij} = 1$ and $\sum_i \beta_{ij} = 1$, respectively.*

The proof of Equation (13) is given in Appendix D.1. An iterative heuristic Algorithm solving Equations (14) efficiently is described in Appendix D.2.

Following the general GM logic and terminology introduced in Wiegerinck and Heskes (2003) and Yedidia et al. (2005), we call $F_f^{(\gamma)}(\beta|p)$ the fractional FE. Obviously the two extremes of $\gamma = -1$ and $\gamma = 1$ correspond to BP and MF limits, respectively. Many features of the BP and MF approaches extend naturally to the FFE-based case. In particular, one arrives at the following statement which appears in Theorem 60 of Vontobel (2013).

**Proposition 14 (Fractional convexity)** *The FFE-based function defined in Equation (13), $F_f^{(\gamma)}(\beta|p)$, is a convex function, convex over $\beta \in \mathcal{B}_p$ for any $\gamma \in [-1;1]$ and any non-negative $p$.*

Obviously, this statement generalizes Proposition 7. Also, the following statement becomes a direct consequence of Proposition 14:

**Corollary 15 (Uniqueness of the interior fractional minimum)** *If the minimum of $F_f^{(\gamma)}(\beta|p)$ is realized at $\beta \in \mathcal{B}_p^{(int)}$, it is unique.*

### 3.4 Minimal FFE-based Solution

It is clear that at $\gamma > 0$ the FFE-based Equations (14) cannot have a perfect matching solution, thus suggesting that at least in this case the solution, if it exists, is in the interior, $\beta \in \mathcal{B}_p^{(int)}$. On the other hand general existence (for any $p$) of such a solution follows immediately from the existence in a special case, for example of $p$ with all elements equal, and then from the continuity of the Equations (14) solution with respect to $p$.

The case of $\gamma \in ]-1;0]$ is a bit trickier. In this case, Equations (14) formally do allow a perfect matching solution. However, for all but degenerate $p$, that is, one reducible by permutations to a

diagonal matrix, the perfect matching solution is an isolated point. Indeed, let us consider a vicinity of a degenerate $p$. If one picks (without loss of generality) a diagonal, $p^{(0)} = (a_i \delta_{ij} | (i,j) \in \mathcal{E})$, and consider $p = p^{(0)} + \delta p$, where $\delta p$ is a small positive matrix, then one observes that Equations (14) do allow a solution, $\beta = 1 + \varepsilon b$, where $\varepsilon$ is a small positive scalar and $b = (b_{ij} | (i,j) \in \mathcal{E}; \forall i \in \mathcal{V}_1 : \sum_j b_{ij} = 0;\ \forall j \in \mathcal{V}_2 :\ \sum_i b_{ij} = 0)$ is a matrix with $O(1)$ elements, if the following scaling relation holds, $|\delta| \sim \varepsilon^{1+\gamma}$. Moreover, one also finds that a solution $\beta$ is $\varepsilon$-close to a perfect matching only if $p$ is $\varepsilon^{1+\gamma}$-close to a diagonal matrix. Now we apply the same continuity and existence arguments, as used above in the $\gamma > 0$ case arguments, to find out that the following statement holds.

**Proposition 16 (FFE-based Minima)** *The minimal FFE-based solution, defined by*

$$-\log(Z^{(\gamma)}_{o-f}(p)) = F^{(\gamma)}_{o-f}(p) = \min_{\beta \in \mathcal{B}_p} F^{(\gamma)}_f(\beta|p),$$

*can only be achieved for $\gamma > -1$ and general (non-degenerate) $p$ at $\beta \in \mathcal{B}^{(int)}_p$.*

Then the following statement follows.

**Proposition 17 ($\gamma$-monotonicity and continuity)** *For any non-negative $p$, $Z^{\gamma}_{o-f}(p)$ is a monotonically increasing and continuous function of $\gamma$ in $[-1;1]$.*

**Proof** Observe that for any non-negative $p$ and doubly stochastic matrix $\beta$, $\sum_{(i,j)\in\mathcal{E}}(1-\beta_{ij})\log(1-\beta_{ij}) < 0$, so for any $\gamma_{1,2} \in [-1;1]$ such that $\gamma_1 > \gamma_2$, $F^{(\gamma_1)}_f(\beta|p) \le F^{(\gamma_2)}_f(\beta|p)$. Then according to the definition of $F^{(\gamma)}_{o-f}(p)$, $F^{(\gamma_1)}_{o-f}(p) \le F^{(\gamma_1)}_f(\beta|p) \le F^{(\gamma_2)}_f(\beta|p)$, for any doubly stochastic matrix $\beta$, in particular for $\beta$ which is optimal for $\gamma_2$. Finally, $F^{(\gamma_1)}_{o-f}(p) \le F^{(\gamma_2)}_{o-f}(p)$, proving monotonicity. The continuity of $Z^{(\gamma)}_{o-f}(p)$ with respect to $\gamma$ in $]-1;1]$ follows from Proposition 16 combined with $F^{(\gamma)}_f(\beta|p)$ continuity with respect to both $\gamma \in [-1;1]$ and $\beta \in \mathcal{B}^{(int)}_p$. (The intuition with respect to the continuity is as follows: an increase in $\gamma$ pushes the optimal $\beta$ away from the boundary of the $\mathcal{B}_p$ polytope.) ■

## 4. Permanent Inequalities, Special Value of $\gamma$, and Conjectures

We start this Section discussing in Subsection 4.1 the recently derived permanent inequalities related to BP and MF analysis. Then, we switch to describing new results of this manuscript in Subsection 4.2, which are mainly related to the FFE-based generalizations of the inequalities discussed in Subsection 4.1. We also discuss in Subsection 4.2 the special (and $p$-dependent) value of the fractional coefficient $\gamma$ for which perm$(p)$ is equal to $Z^{\gamma}_{o-f}(p)$. Finally, Subsection 4.3 is devoted to discussing Conjectures whose resolutions should help to tighten bounds for the permanent.

### 4.1 Recently Derived Inequalities

In this Subsection we discuss a number of upper and lower bounds on permanents of positive matrices introduced recently. Our task is two-fold. First, we wish to relate the bounds/inequalities to the BP and MF approaches introduced and discussed in the preceding Section. Some of these relations and interpretations are new. However, we also aim to test these bounds, and specifically to

characterize the tightness of the bounds by testing the gap as a function of advection and diffusion parameters in the 2d diffusion+advection model in Section 5.

The first bound of interest is

**Proposition 18 (BP lower bound)** *For any non-negative $p$*

$$\mathrm{perm}(p) \geq Z_{o-BP}(p). \tag{16}$$

This statement, as an experimental but unproven observation, was made in Chertkov et al. (2008). It was stated as a Theorem (theorem # 14) in Vontobel (2010), but the proof was not provided. See also discussion in Vontobel (2013) following Theorem 49/Corollary 50. The statement was proven in Gurvits (2011). Interpreted in terms of the terminology and logic introduced in the preceding Section, the proof of Gurvits (2011) consists (roughly) in combining the inequality by (Schrijver, 1998)

$$\mathrm{perm}(\beta .* (1-\beta)) \geq \prod_{(i,j)} (1-\beta_{ij}), \tag{17}$$

stated for any doubly stochastic matrix $\beta$, with some (gauge) manipulations/transformations of the type discussed above in Sections 3.1.1. We give our version of the proof (similar to the one in Gurvits (2011) in spirit, but somewhat different in details) in Appendix E. One direct Corollary of the bound (16) discussed in Gurvits (2011), is that

**Corollary 19** *For an arbitrary doubly stochastic matrix $\phi$*

$$\mathrm{perm}(\phi) \geq Z_{o-BP}(\phi) \geq \prod_{(i,j)} (1-\phi_{ij})^{1-\phi_{ij}}. \tag{18}$$

Next, the following two lower bounds follow from analysis of Equation (7).

**Proposition 20 (BP lower bound #1)** *For any non-negative $p$ and doubly stochastic matrix $\beta \in \mathcal{B}_p^{(int)}$ solving Equations (6) (if the solution exists) results in*

$$\mathrm{perm}(p) \geq Z_{BP}(\beta|p) \prod_{(i,j)} (1-\beta_{ij})^{\beta_{ij}-1} \frac{n!}{n^n}.$$

This is the statement of Corollary 7 of Watanabe and Chertkov (2010) valid for any interior point solution of the BP-equations, and it follows from the Gurvits-van der Waerden Theorem of Gurvits (2008) and Laurent and Schrijver (2010), also stated as Theorem 6 in Watanabe and Chertkov (2010).

**Proposition 21 (BP lower bound #2)** *For any non-negative $p$ and $\beta \in \mathcal{B}_p^{(int)}$ solving Equations (6) (if the solution exists) results in*

$$Z \geq 2 Z_{BP}(\beta|p) \Big(\prod_{i,j} (1-\beta_{ij})\Big)^{-1} \prod_i \beta_{i\Pi(i)} (1-\beta_{i\Pi(i)}),$$

*where $\Pi$ is an arbitrary permutation.*

This statement was made in Theorem 8 in Watanabe and Chertkov (2010) and it is also related to an earlier observation of Engel and Schneider (1973).[5]

**Proposition 22 (BP upper bound)** *For any non-negative $p$ and $\beta \in \mathcal{B}_p^{(int)}$ solving Equations (6) (if the solution exists)*

$$\mathrm{perm}(p) \leq Z_{BP}(\beta|p)\Big(\prod_{(i,j)\in\mathcal{E}}(1-\beta_{ij})\Big)^{-1}\prod_j(1-\sum_i(\beta_{ij})^2),$$

This statement was made in Proposition 9 of Watanabe and Chertkov (2010).

### 4.2 New Bounds and $\gamma_*$

Of the bounds discussed above, three are related to BP and one to MF, while as argued in Section 4.3 the FFE-based approach interpolates between BP and MF. This motivates exploring below new FFE-based generalizations of the previously known (and discussed in the preceding Subsection) BP and MF bounds.

We first derive a new lower bound generalizing Proposition 20 to the FFE-based case.

**Proposition 23** *The following is true for any doubly stochastic matrix $\beta$ and any $\gamma \in [-1;1]$*

$$\mathrm{perm}\left(\beta.*(1-\beta).^{-\gamma}\right) \geq \frac{n!}{n^n}\prod_{(i,j)}(1-\beta_{ij})^{-\gamma\beta_{ij}}.$$

**Proof** This bound generalizes Corollary 7 of Watanabe and Chertkov (2010), and it follows directly from the Gurvits-van der Waerden Theorem of Gurvits (2008) and Laurent and Schrijver (2010) (see also Proposition 8 of Watanabe and Chertkov (2010), where a misprint should be corrected $n^n/n! \to n!/n^n$), and the inequality, $\sum_j \beta_{ij}(1-\beta_{ij})^{-\gamma}x_j \geq \prod_j\left((1-\beta_{ij})^{-\gamma}x_j\right)^{\beta_{ij}}$. ■

Then, combining Proposition 23 with Theorem 13, one arrives at the following statement, generalizing Proposition 20:

**Corollary 24 (FFE-based lower bound)** *For any non-negative $p$ and $\beta \in \mathcal{B}_p^{(int)}$ solving Equations (14) (if the solution exists), the following lower bound holds for any $\gamma \in [-1;1]$*

$$\mathrm{perm}(p) \geq Z_f^{(\gamma)}(\beta|p)\frac{n!}{n^n}\prod_{(i,j)\in\mathcal{E}}(1-\beta_{ij})^{\gamma(1-\beta_{ij})}. \tag{19}$$

Next, one arrives at the following FFE-based generalization of Proposition 22.

**Corollary 25 (FFE-based upper bound #1)** *For any non-negative $p$ and $\beta \in \mathcal{B}_p^{(int)}$ solving Equations (14) (if the solution exists), the following upper bound holds for any $\gamma \in [-1;1]$*

$$\mathrm{perm}(p) \leq Z_f^{(\gamma)}(\beta|p)\Big(\prod_{(i,j)\in\mathcal{E}}(1-\beta_{ij})^{\gamma}\Big)\prod_j\sum_i\beta_{ij}(1-\beta_{ij})^{-\gamma}.$$

5. The proof of the Theorem 8 in Watanabe and Chertkov (2010) contained a misprint that was corrected in the erratum available at `https://sites.google.com/site/mchertkov/publications/mypapers/91_erratum.pdf`.

This upper bound follows from combining Theorem 13, with the simple (and standard) upper bound, $\mathrm{perm}(A) \leq \prod_j (\sum_i A_{i,j})$ applied to $A = \beta .* (1-\beta).^{-\gamma}$.

Note that Corollary 25, applied to the $\gamma = 0$ case and reinforced by the observation, that for $\gamma \geq 0$ the minimum of the FFE-based function (13) is achieved in $\beta \in \mathcal{B}_p^{(int)}$, translates into

$$\mathrm{perm}(p) \leq Z_{o-f}^{(\gamma=0)}(p). \tag{20}$$

Combined with Proposition 17, Equation (20) results in the following:

**Corollary 26 (FFE-based upper bound #2)** *For any non-negative $p$*

$$\forall \gamma \geq 0: \quad \mathrm{perm}(p) \leq Z_{o-f}^{(\gamma)}(p).$$

This completes the list of inequalities we were able to derive generalizing the BP and MF inequalities stated in the preceding Subsection for the FFE-based case. These generalizations are valid (at least) for any $\gamma \in [0;1]$. Therefore, one may hope to derive somewhat stronger statements reinforcing the continuous family of inequalities with the mononotonicity of the FFE-based approach stated in Proposition 17.

Indeed, combining Equations (16) with Propositions 17,26 one arrives at

**Proposition 27 (Special $\gamma_*$)** *For any non-negative $p$ there exists a special $\gamma_* \in [-1;0]$, such that $\mathrm{perm}(p) = Z_{o-f}^{(\gamma_*)}(p)$, and the minimal FFE-based solution upper (lower) bounds the permanent at $0 \geq \gamma > \gamma_*$ ($-1 \leq \gamma < \gamma_*$).*

proposition 27 motivates our experimental analysis of the $\gamma_*(p)$ dependence discussed in Section 5.

Note also that due to the monotonicity stated in Proposition 17, the $\gamma = 0$ upper bound on the permanent is tighter than the MF, $\gamma = 1$, upper bound. However, and as discussed in more details in the next Subsection, even the $\gamma = 0$ upper bound on the permanent is not expected to be tight.

### 4.3 Conjectures

It was Conjectured in Vontobel (2010) that

$$\mathrm{perm}(p) \leq Z_{o-BP}(p) * f(n), \tag{21}$$

and also that $f \sim \sqrt{n}$. The second part of the Conjecture was disproved by Gurvits (2011) with an explicit counter-example. The inequality in Equation (21) turns into the equality $f(n) = \sqrt{2}^n$ when $p$ is doubly stochastic matrix and block diagonal, with all the elements in the $2 \times 2$ blocks equal to $1/2$.[6] Then it was Conjectured in Gurvits (2011) that

**Conjecture 28 (BP upper bound Gurvits, 2011)** *For any non-negative $p$, $f(n)$ in Equation (21) is $\sim \sqrt{2}^n$.*

Another related (but not identical) Conjecture of Gurvits (2011) is as follows:

6. Note that this special form of the $2 \times 2$ block corresponds to the "double degeneracy" discussed in the paragraph preceding Corollary 8.

**Conjecture 29** *The following inequality holds for any doubly stochastic $n \times n$ matrix $\phi$:*

$$\mathrm{perm}(\phi) \leq \sqrt{2}^n \prod_{(i,j)} (1-\phi_{ij})^{(1-\phi_{ij})}. \tag{22}$$

Note that if Equation (22) is true it implies according to Linial et al. (2000) a deterministic polynomial-time Algorithm to approximate the permanent of $n \times n$ nonnegative matrices within the relative factor $\sqrt{2}^n$.

It can be verified directly that the condition (22) is achieved (i.e., inequality is turned into equality) for the special matrix built of the "doubly degenerate" blocks. This special case results in $Z_f^{(\gamma)}$ with $\gamma = -1/2$ on the right-hand side of Equation (22). Therefore one reformulates Conjecture 28 as

**Conjecture 30** *The following inequality holds for any non-negative $p$*

$$\mathrm{perm}(p) \leq Z_{o-f}^{\gamma=-1/2}(p).$$

We refer an interested reader to Vontobel (2013) for discussion of some other Conjectures related to permanents.

## 5. Experiments

We experiment with deterministic and random (drawn from an ensemble) non-negative matrices.

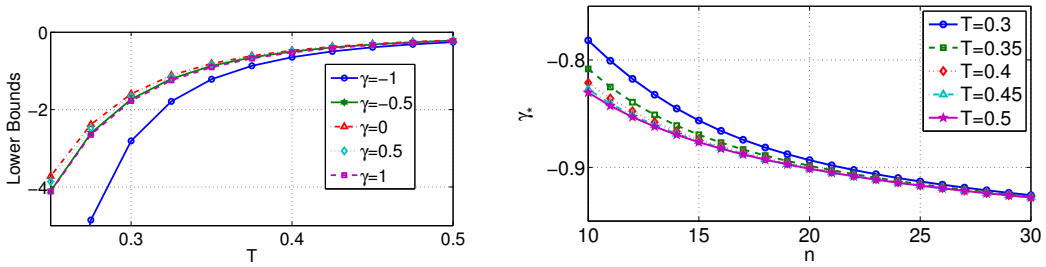
Our simple deterministic example is of the matrices with elements taking two different values such that all the diagonal are the same and such that all the off-diagonal elements are the same (Watanabe and Chertkov, 2010).

In our experiments with stochastic matrices we consider the following four different ensembles

- $(\lambda_{\rm in}, \lambda_{\rm out})$*:* Ensemble of matrices motivated by Chertkov et al. (2008) and Chertkov et al. (2010) and corresponding to a mapping between two consecutive images in 2d flows parameterized by the vector $\lambda = (a,b,c,\kappa)$, where $\kappa$ is the diffusion coefficient and $(a,b,c)$ stand for the three parameters of the velocity gradient tensor (stretching, shear and rotation, respectively—see Chertkov et al. 2010 for details). In generating such a matrix $p$ we need to construct two sets of $\lambda$ parameters. The first one, $\lambda_{\rm in}$, is used to generate an instance of particle positions in the second image, assuming that particles are distributed uniformly at random in the first image. The second one, $\lambda_{\rm out}$, corresponds to an instance of the guessed values of the parameters in the learning problem, where computation of the permanent is an auxiliary step. (Actual optimal learning consists in computing the maximum of the permanent over $\lambda_{\rm out}$.) In our simulations we test the quality of the permanent approximations in the special case, when $\lambda_{\rm in} = \lambda_{\rm out}$, and also in other cases when the guessed values of the parameters do not coincide with the input ones, $\lambda_{\rm in} \neq \lambda_{\rm out}$.

- $[0;\rho]$-*uniform:* In this case one generates elements of the matrix independently at random and distributed uniformly within the $[0;\rho]$-range.

- $\delta$-*exponential:* In this case one generates elements of the matrix independently at random. Any element is an exponentially distributed random variable with mean $\delta$.




(a) log(Right hand side of Equation (19)/perm($p$)) vs. $T$ at $n = 20$, $w = 2$ and different values of $\gamma$

(b) $\gamma_*$ vs. $n$ at different values of $T$ and $w = 2$

Figure 2: Illustration for the case of the deterministic matrix (23). Figure (a) shows the gap between the exact permanent and its lower bound estimate by Equation (24). Figure (b) shows dependence of the special $\gamma_*$ on the parameters.

- $[0;\rho]$-*shifted:* We generate the block diagonal matrix with $\begin{pmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{pmatrix}$ blocks and add independent random and uniformly distributed in the $[0;\rho]$ interval components to all elements of the matrix. The choice of this ensemble is motivated by the special role played by the (doubly degenerate) block-diagonal matrix in the Gurvits Conjecture discussed in Section 4.3.

To make the task of the exact computation of the permanent of a random matrix tractable we consider sparsified versions of the ensembles defined above. To achieve this goal we either prune full matrix from the bare (i.e., not yet pruned) ensemble, according to the procedure explained in Appendix F, or in the case of the $[0;\rho]$-uniform ensemble we first generate a sufficiently sparse sub-graph of the fully connected bipartite graph (for example picking a random subgraph of fixed $O(1)$ degree) and then generate nonzero elements corresponding only to edges of the sub-graph.

### 5.1 Deterministic Example

We consider a simple example which was already discussed in Watanabe and Chertkov (2010). The permanent of the matrix $p$ with elements

$$p_{ij} = \begin{cases} w^{1/T}, & i = j \\ 1, & i \neq j \end{cases},$$

where $w > 1$ and $T > 0$, can be evaluated through the recursion,

$$\sum_{k=0}^{n} W^{(n-k)/T} \begin{pmatrix} n \\ k \end{pmatrix} D_k,$$

$D_0 = 1$, $D_1 = 0$, and $\forall k \geq 2, D_k = (k-1)(D_{k-1} + D_{k-2})$. On the other hand, seeking for solution of the FFE-based Equations (14) in the form of a doubly stochastic matrix $\beta$, where

$$\beta_{ij} = \begin{cases} 1 - \varepsilon(n-1), & i = j \\ \varepsilon, & i \neq j \end{cases}, \tag{23}$$

one finds that $\varepsilon$ should satisfy the following transcendental equation,

$$(1-\varepsilon(n-1))(1-\varepsilon)^{\gamma} = w^{1/T}(n-1)^{\gamma}\varepsilon^{1+\gamma}.$$

At $T\to\infty$ this equation has a unique uniform, $\varepsilon\to 1/(n-1)$, solution. An interior, $\varepsilon>0$, solution of Equation (23) exists, and it is also unique, at any finite $T$ for $\gamma>-1$. According to Watanabe and Chertkov (2010), the interior solution does not exists at $\gamma=-1$ and $T<\log\omega/\log(n-1)$.

To test the gap between the exact expression for the permanent and the FFE-based lower bound of Corollary 24, we fix $w=2, n=20$ and vary the temperature parameter, $T$. The results are shown in Figure 2a. One finds that the gap depends on $\gamma$ with $\gamma=0$ resulting in the best lower bound for all the tested temperatures. One also observes that the $\gamma$-dependence of the gap decreases with increase in $T$. Figure 2b shows dependence of the special $\gamma_*$, defined in Proposition 27, on $n$ and $T$ at $w=2$. One finds that, consistently with the Conjecture 30, $\gamma_*$ is always smaller than $-1/2$ and it also decreases with increase in either $n$ or $T$.

### 5.2 Random Matrices. Special $\gamma_*$.

We search for the special $\gamma=\gamma_*$, defined in Proposition 27, by calculating the permanent of a full matrix, $p$, of size $n\times n$, with $n=3,\ldots,14$, and of a pruned matrix with $n=10,\ldots,40$, and then comparing it with the FFE-based value $Z_f^{(\gamma)}(\beta,p)$,[7] where the doubly stochastic matrix $\beta$ solves Equations (14) for given $p$, for different $\gamma$. By repeatedly evaluating the FFE-based approximation for different values of $\gamma$ and then taking advantage of the $Z_f^{(\gamma)}$ monotonicity and continuity with respect to $\gamma$ and performing a search we find the special $\gamma$ for a specific $p$.

In general we observed that the special $\gamma_*$ for tested matrices was always less than or equal to $-1/2$, which is consistent with Conjecture 30. We also observed, estimating or extrapolating the approximate value of the special $\gamma_*$ for a given matrix, that it might be possible to estimate the permanent of a matrix efficiently and very accurately for some ensembles.

#### 5.2.1 THE $(\lambda_{\rm in},\lambda_{\rm out})$ ENSEMBLE

In this Subsection we describe experiments with several of the $(\lambda_{\rm in},\lambda_{\rm out})$ ensembles. We are interested in studying the dependence of the special $\gamma_*$, defined in Proposition 27, on the matrix size and other parameters of the ensemble. We consider here a variety of cases.

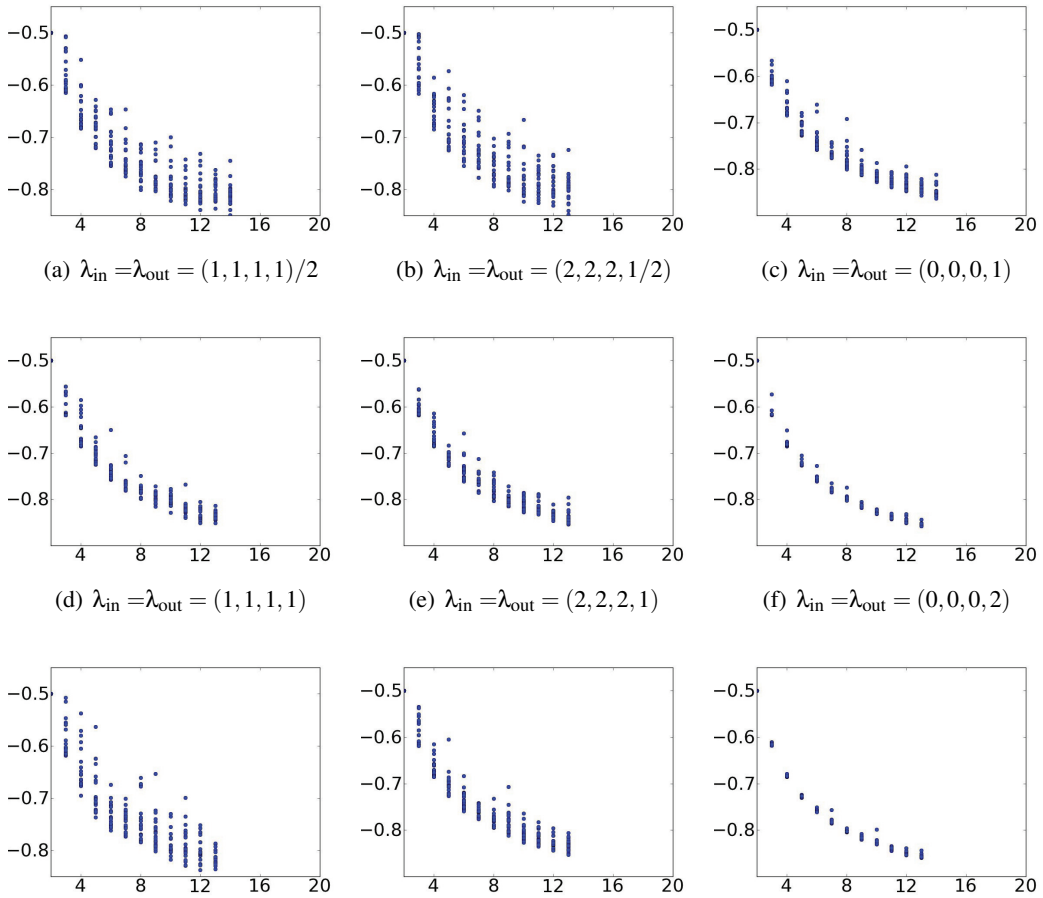
Figure 3 shows the results of experiments with full but (relatively) small matrices and different values $\lambda_{\rm in},\lambda_{\rm out}$. The results are presented in the form of a scatter plot, showing results for different matrix instances from the same ensemble.

As can be seen from the grouping of the first five plots in Figure 3, the dependence of the special $\gamma_*$ on the matrix size at $\lambda_{\rm in}=\lambda_{\rm out}$ is largely sensitive to the diffusion parameter $\kappa$ and it is not so dependent on the advection parameters $a,b,c$. Indeed, Figures 3 (a,b) are similar to each other, as are Figures 3(c-e), despite having different values of $a,b,c$.

Figures 3(a-e) also demonstrate an interesting feature: the lower $\kappa$, the more erratic the behavior of the special $\gamma_*$.

Analyzing the three last cases in Figure 3 with $\lambda_{\rm in}\neq\lambda_{\rm out}$, we observed that the larger the value of $\kappa_{out}$, the more regular the resulting behavior.

7. In the following we will use the shorter notation, $Z_f^{(\gamma)}$ for this object.




(a) $\lambda_{\text{in}} = \lambda_{\text{out}} = (1,1,1,1)/2$ (b) $\lambda_{\text{in}} = \lambda_{\text{out}} = (2,2,2,1/2)$ (c) $\lambda_{\text{in}} = \lambda_{\text{out}} = (0,0,0,1)$

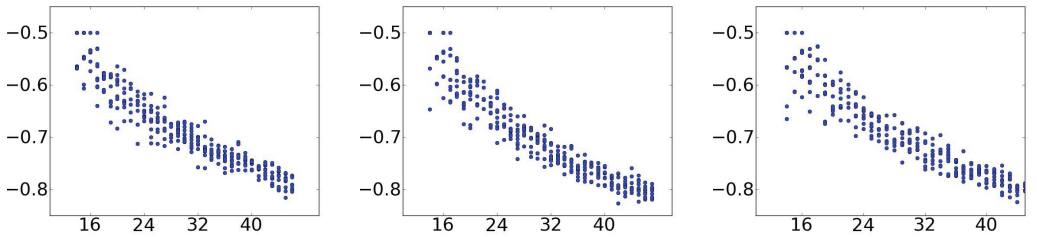
(d) $\lambda_{\text{in}} = \lambda_{\text{out}} = (1,1,1,1)$ (e) $\lambda_{\text{in}} = \lambda_{\text{out}} = (2,2,2,1)$ (f) $\lambda_{\text{in}} = \lambda_{\text{out}} = (0,0,0,2)$

(g) $\lambda_{\text{in}} = (1,1,1,1)$, $\lambda_{\text{out}} = (0,0,0,1)$ (h) $\lambda_{\text{in}} = (0,0,0,2)$, $\lambda_{\text{out}} = (0,0,0,1)$ (i) $\lambda_{\text{in}} = (0,0,0,1)$, $\lambda_{\text{out}} = (0,0,0,2)$

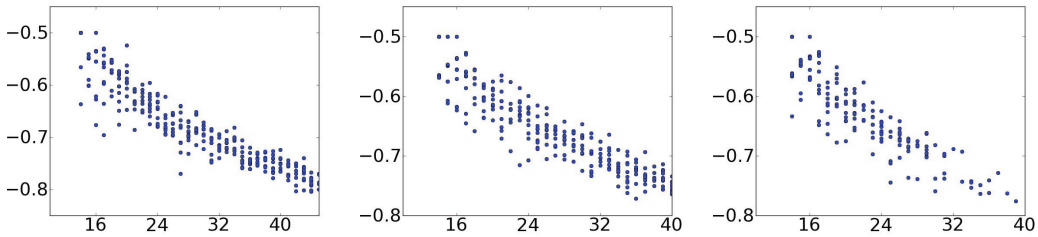
Figure 3: Scatter plot of the special $\gamma_*$ calculated for instances from the $(\lambda_{\text{in}}, \lambda_{\text{out}})$ ensemble and varying the matrix size within the $2 \div 14$ range (no pruning).

Figure 4 shows the same scatter plots as in Figure 3, observed for larger but sparser (90% pruned) matrices. We observed the general tendency for the average special $\gamma_*$ to decrease with increasing $n$; however, it is not clear from the observations if the resulting level of fluctuations decreases with the increase in $n$ or remains the same.

Summarizing, for the $(\lambda_{\text{in}}, \lambda_{\text{out}})$ ensemble, we found that the behavior of the special $\gamma_*$ with respect to matrix size to be largely dependent on $\kappa_{out}$, the diffusion coefficient used to generate the matrix, while the dependence on other factors is significantly less pronounced. The average special $\gamma_*$ decreases with increasing $n$, while respective variance remains roughly the same.

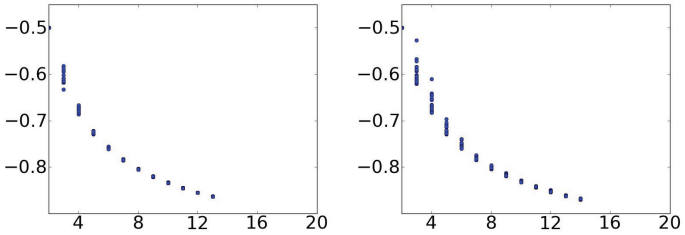



(a) $\lambda_{\text{in}} = \lambda_{\text{out}} = (1,1,1,1)/2$ (b) $\lambda_{\text{in}} = \lambda_{\text{out}} = (1,1,1,1)$ (c) $\lambda_{\text{in}} = \lambda_{\text{out}} = (0,0,0,1)$




(d) $\lambda_{\text{in}} = \lambda_{\text{out}} = (2,2,2,1/2)$ (e) $\lambda_{\text{in}} = (0,0,0,2)$, $\lambda_{\text{out}} = (0,0,0,1)$ (f) $\lambda_{\text{in}} = (1,1,1,1), \lambda_{\text{out}} = (0,0,0,1)$

Figure 4: Scatter plot of the estimated special $\gamma_*$ calculated for instances from the $(\lambda_{\text{in}}, \lambda_{\text{out}})$ ensemble and varying the matrix size within the $15 \div 40$ range (with 90% pruning).



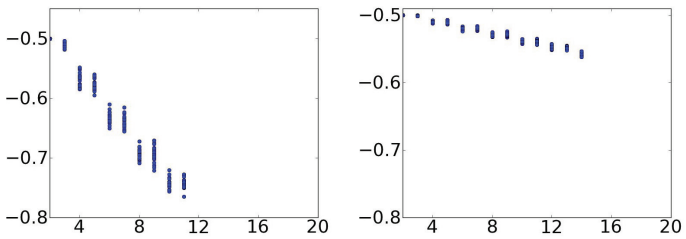

(a) uniform ensemble. Full (small) matrix. (b) $\delta$-exponential ensemble with $\delta = 1$. Full (small) matrix.

Figure 5: Scatter plot of the estimated special $\gamma$ calculated for instances from a random matrix ensemble and varying the matrix size.

### 5.2.2 UNIFORM AND $\delta$-EXPONENTIAL ENSEMBLES

Figure 5 shows scatter plots for examples of the (a) $[0;1]$-uniform ensemble,[8] and (b) $\delta$-exponential ensemble. Here we found a very impressive decrease in variance with increase in the matrix size. Besides, we observe that in spite of their difference, the two ensembles show qualitatively similar behavior of $\gamma_*$ as a function of $n$. This indicates that for large matrices, whose entries are inde-

8. The simulations results of Section 4 of Huang and Jebara (2009) and the Conjectures in Section VII.B of Vontobel (2013) apply to the same ensemble.




(a) $\rho = 1/20$-shifted (b) $\rho = 1/200$-shifted

Figure 6: Scatter plot of the estimated special $\gamma_*$ calculated for instances from two examples of the $[0;\rho]$-shifted ensembles and varying the matrix size.

pendent random variables, we could achieve excellent accuracy by extrapolating on the special $\gamma_*$ and estimating $Z_f(\gamma_*) = \text{perm}(p)$. Indeed, the rapidly decreasing variance suggests that most of the error would come from the extrapolation of the average special $\gamma_*$ for a given matrix size and not the error of $Z_f(\gamma_{*,\text{average}}) - Z_f(\gamma_{*,\text{actual}})$, since $\gamma_{*,\text{average}}$ and $\gamma_{*,\text{actual}}$ will be very close in value.

#### 5.2.3 THE $[0;\rho]$-SHIFTED ENSEMBLE

We also studied the special $\gamma_*$ in "badly-behaved" cases such as the one brought up earlier, with $2 \times 2$ squares of $1/2$'s positioned along the diagonal. (See discussion in Section 4.3.) It can be easily shown that the special value of $\gamma_*$ of the bare block-diagonal matrix is $-1/2$. Unsurprisingly, our experiments, documented in Figure 6, showed that: (a) the resulting $\gamma$ is always smaller than $-1/2$, and (b) as more noise was introduced, the special $\gamma_*$ decreased in value faster with respect to matrix size. However, this decrease with $n$ towards smaller $\gamma_*$ was much slower than in other ensembles, particularly for low noise.

### 5.3 Random Matrices: Testing Inequalities and Conjectures

Figures 7,8 and Figures 9,10, showing the average behavior and scatter plots for smaller and larger (pruned) matrices, respectively (see Figure captions for explanations), present experimental verification to the variety of inequalities discussed in Section 4.1. The ensemble used for these plots was the ensemble $\lambda_{\text{in}} = \lambda_{\text{out}} = (1,1,1,1)/2$. The data suggests that neither of the bounds are actually tight, and moreover the values of the gaps, between the exact expression and respective estimates tested, fluctuate more strongly with increasing matrix size. We also observe from Figures 7 and Figures 8f, that Equation (21) has $f(n)$ growing faster with $n$ than $\sim \sqrt{n}$ even on average. In the case of larger pruned matrices we removed the two expressions $\log((Z_{BP}(\prod_{(i,j)\in\mathcal{E}}(1-\beta_{ij}))^{-1}\prod_j(1-\sum_i(\beta_{ij})^2))/Z)$ and $\log((2Z_{BP}(\prod_{i,j}(1-\beta_{ij}))^{-1}\prod_i\beta_{i\Pi(i)}(1-\beta_{i\Pi(i)}))/Z)$. We removed the former because in the case of pruning the resulting $\beta$ is often partially-resolved (with some elements of $\beta$ equal to one) and in this case the inequality does not carry any restriction. We removed the latter because, in the pruned case and for a randomly chosen permutation, it is very likely that at least one element of $\beta$ is zero, making the bound discussed unrestrictive.

Figure 11 shows that the bounds given by the corollaries 24, 25 do not depend much on $\gamma$ and that they in practice depend more on matrix size and on peculiarities of individual matrices. There

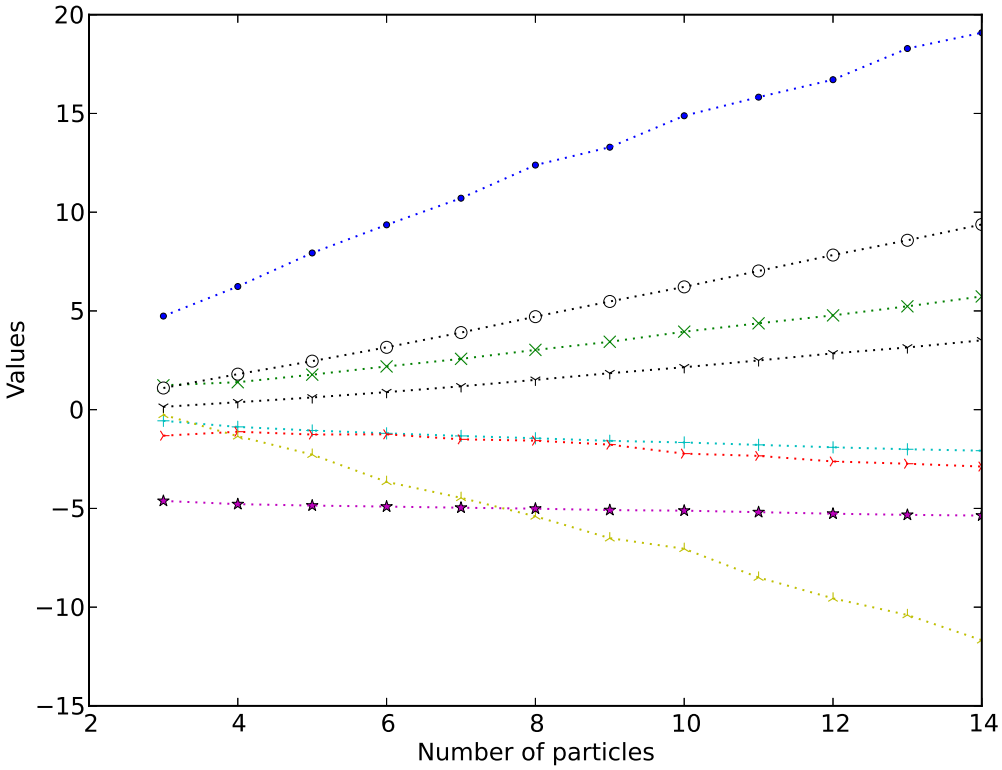
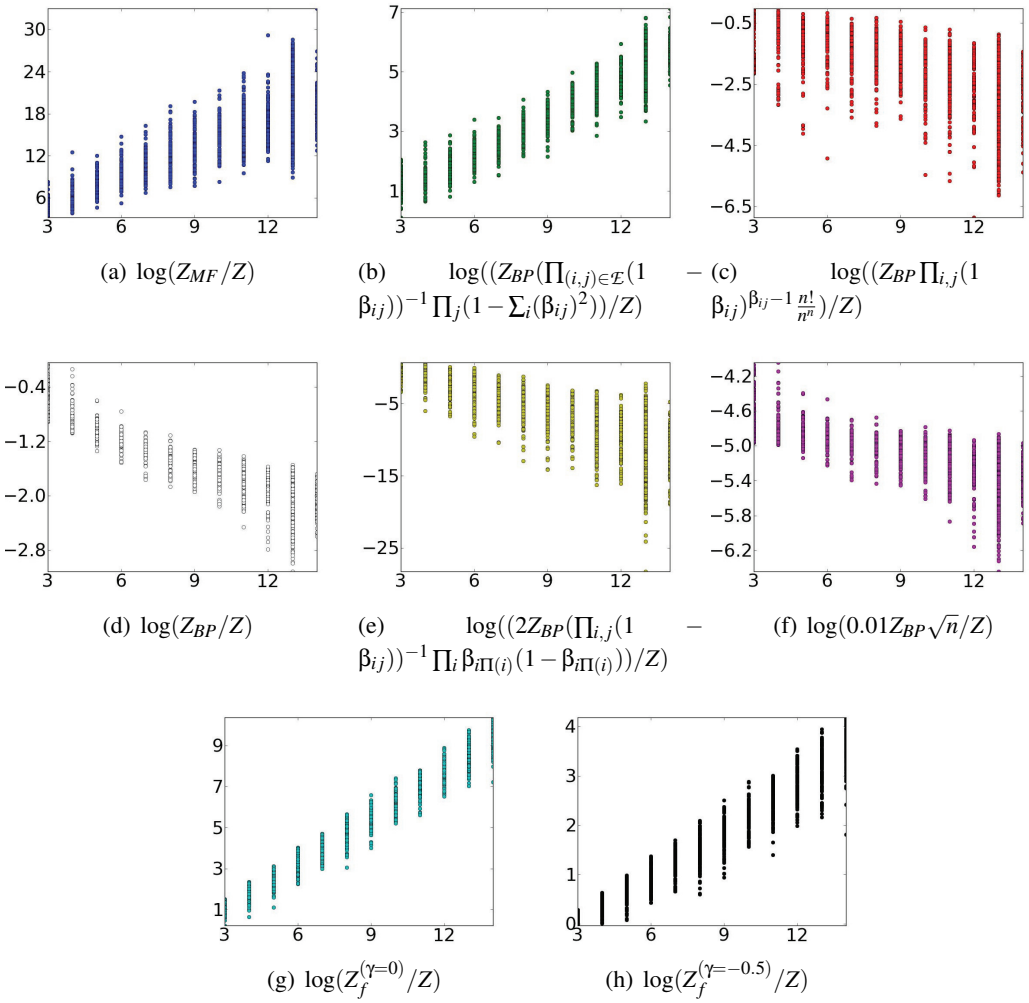



Figure 7: This Figure describes the gap between the actual value of the permanent and the values of the various theoretical upper and lower bounds described in the paper and averaged over different simulation trials. Each color corresponds to an upper or lower bound as follows: Blue dots corresponds to the MF approximation to the permanent, $Z_{MF}$. Green X's correspond to $\log(Z_{BP}(\prod_{(i,j)\in\mathcal{E}}(1-\beta_{ij}))^{-1}\prod_j(1-\sum_i(\beta_{ij})^2)/Z)$. Red sideways Y's correspond to $\log(Z_{BP}\prod_{i,j}(1-\beta_{ij})^{\beta_{ij}-1}\frac{n!}{n^n}/Z)$. Cyan "+" symbols correspond to $\log(Z_{BP}/Z)$. Yellow inverted Y's correspond to $\log(2Z_{BP}(\prod_{i,j}(1-\beta_{ij}))^{-1}\prod_i\beta_{ii}(1-\beta_{ii})/Z)$. Purple stars correspond to $\log(0.01Z_{BP}\sqrt{n}/Z)$. Black circles correspond to $\log(Z_f^{\gamma=0}/Z)$. Black Y's correspond to $\log(Z_f^{\gamma=-0.5}/Z)$. To make a data point, 100 instances, each corresponding to a new matrix are drawn, and the log of the ratio of the bound to the actual permanent is recorded. The data shown corresponds to matrices from the $(\lambda_{\text{in}},\lambda_{\text{out}})$ ensemble.

is a slight change for values of $\gamma$ near $-1$, but otherwise the plot is nearly flat, so it seems that unfortunately little tightening of the bounds can be achieved by tweaking $\gamma$. Another noteworthy observation is that a higher upper bound implies a higher lower bound, and vice-versa.

Figure 12 is related to discussions of Corollary 19 for the permanent of a doubly stochastic matrix. We generate an instance of a doubly stochastic matrix and calculate the respective BP expression in three steps (this is the procedure of Knopp and Sinkhorn (1967), also discussed by



(a) $\log(Z_{MF}/Z)$ (b) $\log((Z_{BP}(\prod_{(i,j)\in\mathcal{E}}(1-\beta_{ij}))^{-1}\prod_j(1-\sum_i(\beta_{ij})^2))/Z)$ (c) $\log((Z_{BP}\prod_{i,j}(1-\beta_{ij})^{\beta_{ij}-1}\frac{n!}{n^n})/Z)$

(d) $\log(Z_{BP}/Z)$ (e) $\log((2Z_{BP}(\prod_{i,j}(1-\beta_{ij}))^{-1}\prod_i\beta_{i\Pi(i)}(1-\beta_{i\Pi(i)}))/Z)$ (f) $\log(0.01Z_{BP}\sqrt{n}/Z)$

(g) $\log(Z_f^{(\gamma=0)}/Z)$ (h) $\log(Z_f^{(\gamma=-0.5)}/Z)$

Figure 8: Scatter plots for the data shown in Figures (7). For better presentation the data is split into 8 sub-figures. The vertical axis of each scatter plot is specific to the behavior of the expression with respect to the matrix size and the color coding for different objects tested coincides with that used in Figure 7.

Huang and Jebara 2009): (a) generate a non-negative matrix from the $[0;1]$ ensemble; (b) re-scale rows and columns of the matrix iteratively to get a respective doubly stochastic matrix;[9] and (c) apply the BP- ($\gamma=-1$) procedure to evaluate the $Z_{o-BP}$ estimate for the resulting doubly stochastic matrix. In agreement with Equation (18), the average value of the log corresponding to the BP-lower bound is positive and smaller than the respective expression for the average of the log of the explicit expression on the right-hand side of Equation (18). (The hierarchical relation obviously holds as well for any individual instance of the doubly stochastic matrix $\beta$ from the generated ensemble.) We

9. The rescaling is a key element of Linial et al. (2000), and we can also think of the procedure as of a version of the $\gamma=0$ iterative Algorithm.

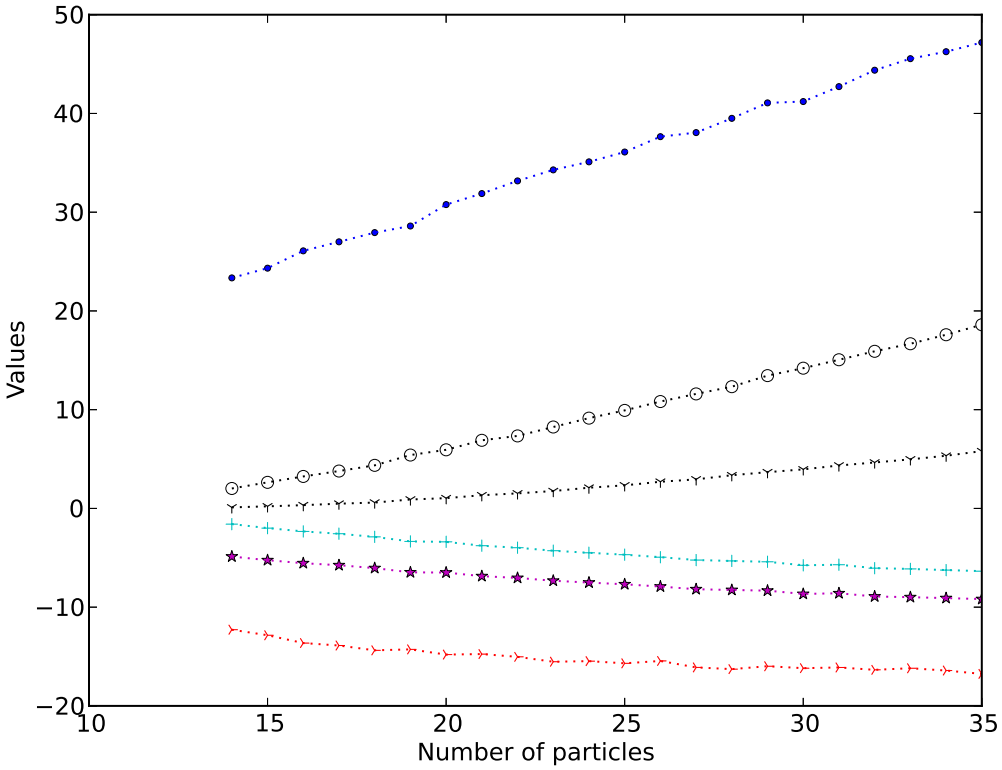
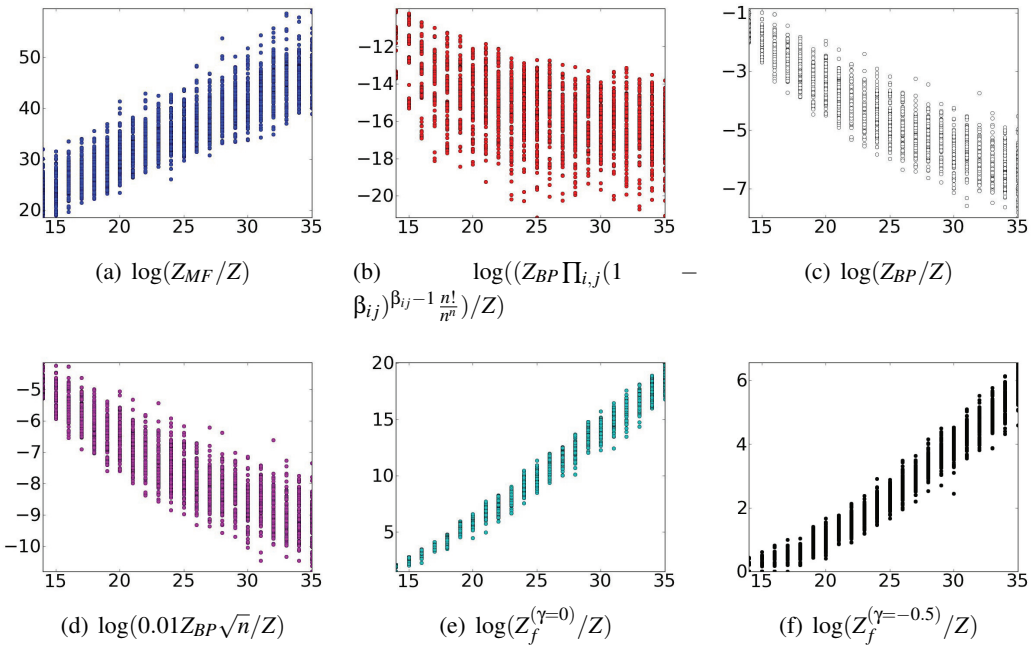



Figure 9: The data is shown like in Figure 7, but for large, sparsified matrices. Less meaningful expressions were removed from the plot. Each color/symbol pair corresponds to a mathematical expression, as follows, blue dots: $\log(Z_{MF}/Z)$, red sideways Y's: $\log(Z_{BP}\prod_{i,j}(1-\beta_{ij})^{(\beta_{ii}-1)}\cdot n!/(n^n Z))$, cyan "+" symbols: $\log(Z_{BP}/Z)$, purple stars: $\log(Z_{BP}\sqrt{n}/(100Z))$, black circles: $\log(Z_f(\gamma=0)/Z)$, black Y's: $\log(Z_f(\gamma=-1/2)/Z)$; where $Z=\mathrm{perm}(p)$.

also observe that the average values of the curves show a tendency to saturate, while the standard deviation decreases dramatically, suggesting that for large $n$ this random ensemble may be well approximated by either BP or, even more simply, by its explicit lower bound from the right-hand side of Equation (18), the latter being in the agreement with the proposal of Gurvits (2011).

## 6. Conclusions and Path Forward

The main message of this and other related recent papers by Chertkov et al. (2008), Huang and Jebara (2009), Chertkov et al. (2010), Watanabe and Chertkov (2010), Vontobel (2010), Vontobel (2013) and Gurvits (2011) is that the BP approach and improvements not only give good heuristics for computing permanents of non-negative matrices, but also provide theoretical guarantees and thus reliable deterministic approximations. The main highlights of this manuscript are



(a) $\log(Z_{MF}/Z)$ (b) $\log((Z_{BP}\prod_{i,j}(1-\beta_{ij})^{\beta_{ij}-1}\frac{n!}{n^n})/Z)$ (c) $\log(Z_{BP}/Z)$

(d) $\log(0.01Z_{BP}\sqrt{n}/Z)$ (e) $\log(Z_f^{(\gamma=0)}/Z)$ (f) $\log(Z_f^{(\gamma=-0.5)}/Z)$

Figure 10: Scatter plots for the data shown in Figures (9). For better presentation the data is split into 6 sub-figures. Each scatter plot is specific to the behavior of the expression with respect to the matrix size and the color coding for different objects tested coincides with the one used in Figure 9.

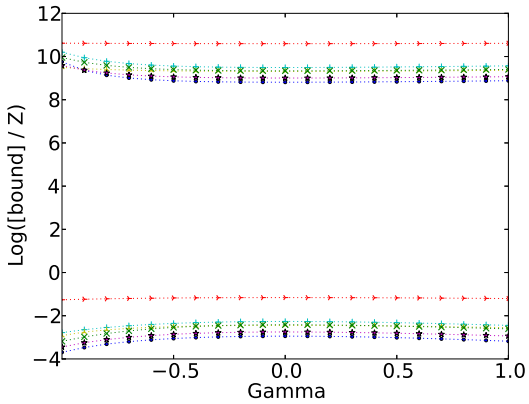
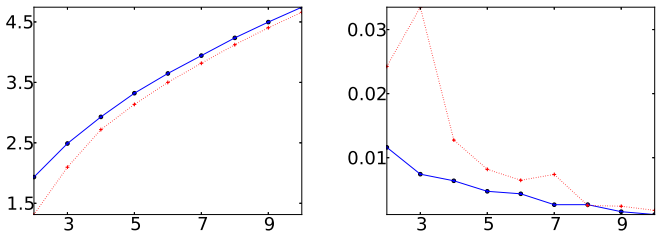



Figure 11: This plot shows $\gamma$-dependence of the gap between upper and lower bounds corresponding to Corollaries 24, 25. Here, the bounds were plotted for six different matrices, each generated with $\lambda_{\text{in}} = \lambda_{\text{out}} = (1,1,1,1)/2$, where the upper and lower bounds are color-coded and use matching symbols to indicate that they correspond to the same matrix.




(a) (b)

Figure 12: A plot of the average (Subfigure a) and standard deviation (Subfigure b) of $\mathrm{perm}(\beta)/\prod_{i,j}(1-\beta_{ij})^{1-\beta_{ij}}$ (in blue dots connected by a solid line) and $\mathrm{perm}(\beta)/Z_{o-BP}(\beta)$ (in red crosses connected by a dashed line), where $\beta$ is a doubly stochastic matrix picked from 100 instances of the random ensemble described in the text, shown as a function of $n$.

- The construction of the FFE-based approach, parameterized by $\gamma \in [-1;1]$ and interpolating between BP ($\gamma=-1$) and MF ($\gamma=1$) limits.
- The discovery of the exact relation between the permanent of a non-negative matrix, $\mathrm{perm}(p)$ and the respective FFE-based expression, $Z_f^{(\gamma)}(p)$, where the latter is computationally tractable.
- The proof of the continuity and monotonicity of $Z_f^{(\gamma)}(p)$ with $\gamma$, also suggesting that for some $\gamma_* \in [-1;0]$, $\mathrm{perm}(p) = Z_f^{(\gamma_*)}(p)$.
- The extension of the list of known BP-based upper and lower bounds for the permanent by their FFE-based counterparts.
- The experimental analysis of permanents of different ensembles of interest, including those expressing relations between consecutive images of stochastic flows visualized with particles.
- Our experimental tests include analysis of the gaps between exact expression for the permanent, evaluated within the ZDD technique adapted to permanents, and the aforementioned BP- and fractional-based lower/upper bounds.
- The experimental analysis of variations in the special $\gamma$ for different ensembles of matrices suggests the following conclusions. First, the behavior of the special $\gamma$ varies for different ensembles, but the general trend remains the same: as long as there is some element of randomness in the ensemble, the special $\gamma$ decreases as matrix size increases. Second, for each ensemble the behavior of the special $\gamma$ is highly distinctive. For some considered random matrix ensembles, the variance decreases quickly with increasing matrix size. All of the above suggest that the FFE-based approach offers a lot of potential for estimating matrix permanents.

We view these results as creating a foundation for further analysis of theoretical and computational problems associated with permanents of large matrices. Of the multitude of possible future problems, we consider the following ones listed below as the most interesting and important:

- Improving BP and FFE-based approaches and making the resulting lower and upper bounds tighter.
- Further analysis of the $\gamma$-dependence, making theoretical statements for statistics of log-permanents at large $n$ and for different random ensembles.
- Using the new permanental estimations and bounds for learning flows in the setting of Chertkov et al. (2010). Combining within the newly introduced FFE-based approach the $\beta$-optimization with optimization over flow parameters (by analogy with what is done in Chertkov et al. 2010). Applying the improved technique to various particle image velocimetry (PIV) experiments of interest in fluid mechanics in general, and specifically to describe spatially smooth multi-pole flows in micro-fluidics, see, for example, discussion of the most recent relevant experiments in Drescher et al. (2010) and Guasto et al. (2010) and references therein.[10]
- Addressing other GM problems of the permanental type, for example, counting matchings (and not only perfect matchings) on arbitrary graphs, drawing inspiration from Sanghavi et al. (2011) generalizing Bayati and Nair (2006) and Bayati et al. (2008) in the ML setting, and higher dimensional matchings, in particular corresponding to matching of paths between multiple consecutive images within the "learning the flow" setting.

## Acknowledgments

We are thankful to Leonid Gurvits, Yusuke Watanabe, Pascal Vontobel, Vladimir Chernyak, Jonathan Yedidia and Jason Johnson for multiple discussions and very helpful advice, as well as to Leonid Gurvits and Pascal Vontobel for sharing their recent results (Gurvits, 2011; Vontobel, 2013) prior to public release. We also very much appreciate the helpful comments and multiple suggestions made by the reviewers. ABY acknowledges support of the Undergraduate Research Assistant Program at LANL and he is also grateful to CNLS at LANL for its hospitality. Research at LANL was carried out under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at Los Alamos National Laboratory under Contract No. DE C52-06NA25396.

## Appendix A. Most Probable Perfect Matching

This short Appendix is introduced to guide the reader through material which is related, but only indirectly (through physics motivation and historical links), to the main subject of the manuscript.

### A.1 Most Probable Perfect Matching over Bi-Partite Graphs

According to Equation (2), the permanent can be interpreted as the partition function of a GM. The partition function represents a weighted counting of the $n!$ perfect matchings. Using "physics terminology" one says that this perfect matching representation allows to interpret the permanent as the statistical mechanics of perfect matchings (called dimers in the physics literature) over the bi-partite graph. This is statistical mechanics at finite temperatures, as the partition function represents a (statistical) sum over the perfect matchings.

10. We are thankful to Eric Lauga for suggesting to us the micro-fluidics experiments as one possible application for the "learning the flow" BP-based approach.

However, it is still of interest to discuss (at least in the context of establishing historical links) the "zero temperature," or maximum likelihood (ML) version of Equation (2)

$$-\log(Z_{ML}(p)) = \min_{\sigma} \sum_{(i,j)\in\mathcal{E}} \sigma_{ij} \log(1/p_{ij}). \tag{24}$$

According to the logic of Yedidia et al. (2005), Equation (24) can also be stated in the probability space (i.e., in terms of $b(\sigma)$) as

$$-\log(Z_{ML}(p)) = \min_{b} \sum_{(i,j)\in\mathcal{E}} \sum_{\sigma} \sigma_{ij} \log(1/p_{ij}) b(\sigma). \tag{25}$$

Then, the function of $b(\sigma)$ which is the object of minimization over beliefs in Equation (25) is naturally the ML (zero temperature) version of the FE function (3).

By construction, $Z_{ML}(p) \leq Z(p)$ for any $p$. Note also that Equation (24) is a linear programming (LP) equation, but one which at first sight appears intractable, giving an optimization defined over a huge polytope and spanning all the perfect matchings with nonzero probability. For a general GM the LP-ML formulation is indeed intractable, but for the specific problem under consideration (finding the perfect matching over a bipartite graph) the ML-LP problem (25) becomes tractable, as discussed below in the next Subsection. Given classical results from optimization theory, related to the so-called Hungarian Algorithm, by Kuhn (1955), and the auction Algorithm, by Bertsekas (1992), this special solvability (reduced complexity) of the ML perfect matching problem is not surprising.

### A.2 Linear Programming Relaxation of BP

The Bethe FE (4) can be split naturally into the self-energy term and the self-entropy terms (at unit temperature), $F_{BP} = E_{BP} - S_{BP}$:

$$E_{BP}(\beta|p) = -\sum_{(i,j)} \beta_{ij} \log(p_{ij}),$$
$$S_{BP}(\beta|p) = \sum_{(i,j)} \left(-\beta_{ij}\log(\beta_{ij}) + (1-\beta_{ij})\log(1-\beta_{ij})\right).$$

If the entropy term is ignored in Equation (8) the problem turns into the linear programming (LP) formulation of BP

$$-\log(Z_{LP}(p)) = \min_{\beta} E_{BP}(\beta). \tag{26}$$

One can also arrive at the same LP formulation (26) by relaxing the original ML-LP setting (25). As shown in Bayati et al. (2008) and Chertkov (2008), the relaxation is provably tight for any $p$ , that is, $Z_{LP}(p) = Z_{ML}(p)$, as the resulting matrix of constraints in the LP problem (26) describing the doubly stochasticity of $\beta$ is totally uni-modular, so the corners of the respective polytope are in one-to-one correspondence with the perfect matching configurations/corners of the higher-dimensional polytope from Equation (25), also in accordance with the Birkhoff-von Neumann Theorem by (Kőnig, 1936; Birkhoff, 1946; von Neumann, 1953).

## Appendix B. Bethe-Free Energy Approach

This Appendix provides some additional details on the Bethe-Free Energy Approach.

### B.1 Exact BP-based Relations for Permanents

We present here a simple proof of Equation (7), essentially following a slightly modified version of what was the main statement of Watanabe and Chertkov (2010).

Consider an interior minimum of the Bethe FE function (4) achieved with a strictly nonzero (for elements with positive $p_{ij}$) doubly stochastic matrix $\beta$. Then, the minimum satisfies Equations (6), where $\log(u)$ are respective Lagrangian multipliers. Weighting the logarithm of Equations (6) with $\beta$, summing up the result over all the edges, using Equation (4) and the double stochasticity of $\beta$, one derives

$$\begin{aligned}&\sum_{(i,j)\in\mathcal{E}}\beta_{ij}\log(u_iu_j)=\sum_{i\in\mathcal{V}_1}\log u_i+\sum_{j\in\mathcal{V}_2}\log u^j\\&=\sum_{(i,j)\in\mathcal{E}}(\beta_{ij}\log(p_{ij}/\beta_{ij})-\beta_{ij}\log(1-\beta_{ij}))=\log Z_{BP}-\sum_{(i,j)\in\mathcal{E}}\log(1-\beta_{ij}).\end{aligned}\tag{27}$$

On the other hand, applying the permanent to both sides of Equation (6) one arrives at

$$\mathrm{perm}(p)=\mathrm{perm}(\beta.*(1-\beta))\left(\prod_{i\in\mathcal{V}_1}u_i\right)\left(\prod_{j\in\mathcal{V}_2}u^j\right).\tag{28}$$

Combining Equation (27) with Equation (28) results in Equation (7).

### B.2 Iterative Algorithm(s) for Finding Solution of BP Equations

First of all, let us recall that according to Proposition 7, (4) is convex. However, as explained above the convexity is not trivial, as it is enforced by global constraints. This lack of convexity of individual edge-local terms in Equation (4) creates a technical obstacle to finding a valid fixed point of $F_{BP}$, suggesting that an iterative Algorithm converging to the fixed point of $F_{BP}$ will be more elaborate than the one discussed below in the MF case.

To find a valid solution of BP in our numerical experiments we use the following practical iterative scheme (heuristics), previously described in Chertkov et al. (2008) (see Equations (7,8) as well as preceding and following explanations):

$$\forall(i,j):\ \ \beta_{ij}(n+1)=\lambda\beta_{ij}(n)+\frac{(1-\lambda)p_{ij}}{p_{ij}+(\sum_k\beta_{kj}(n)/2+\sum_k\beta_{ik}(n)/2-\beta_{ij}(n))^2(u_i(n)u^j(n))},\tag{29}$$

$$\forall i:\ \ u_i(n+1)=\frac{\sum_k p_{ik}/u^k(n)}{1-\sum_j(\beta_{ij}(n))^2},\quad \forall j:\ \ u^j(n+1)=\frac{\sum_k p_{kj}/u_k(n)}{1-\sum_i(\beta_{ij}(n))^2},\tag{30}$$

where the arguments of the $\beta$'s indicate the order of the iterations. The damping parameter $\lambda$ (typically chosen $0.4\div0.5$) helps with convergence. To ensure appropriate accuracy for solutions with $\beta$'s close to zero or unity we also insert a normalization step after Equations (29) but prior to Equations (30), making the following two transformations subsequently, (a) $\forall(i,j)$: $\beta_{ij}\to\beta_{ij}/\sum_k\beta_{ik}$, and (b) $\forall(i,j)$: $\beta_{ij}\to\beta_{ij}/\sum_k\beta_{kj}$. (The two steps implement an elementary step of the Sinkhorn operation from Huang and Jebara 2009.) The Algorithm is sensitive to initial values for $\beta$ and $u$. To ensure convergence, one initiates the Algorithm with the output of the MF scheme (which converges much better) as described in Appendix C.2, that is, $\beta(0)=\beta_{MF}$ and $u(0)=v_{MF}$. Numerical experiments show that this procedure always converges to an interior stationary point of the BFE

(4), when one exists and is not degenerate. In the special cases when the solution is on the boundary it seemed to converge there as well, but we did not study this systematically to make a definitive statement.

Note that the Algorithm presented above is certainly not the only option one can use to find a doubly stochastic matrix solution of BP Equations (6). In fact, the standard sum-product Algorithm (SPA) of Yedidia et al. (2005), stated for the problem of computing the permanent in Chertkov et al. (2010), is a serious competitor, which according to Theorem 32 of Vontobel (2013) always converges to the minimum of the Bethe FE. Future work is required to compare the convergence speed of the two Algorithms.

## Appendix C. Mean-Field (Fermi) Approach

This Appendix provides some additional details on the Mean-Field Approach.

### C.1 Exact MF-based Relations for Permanents

We present here a simple proof of Equation (12), essentially following the logic of what was described above for BP in Appendix B.1.

Weighting the logarithm of Equations (11) with that doubly stochastic matrix $\beta$ which minimizes Equation (12), summing the result over all the edges, and making use of Equations (11,9), one derives

$$\begin{aligned}&\sum_{(i,j)\in\mathcal{E}}\beta_{ij}\log(v_i v_j)=\sum_{i\in\mathcal{V}_1}\log v_i+\sum_{j\in\mathcal{V}_2}\log v^j\\&=\sum_{(i,j)\in\mathcal{E}}(\beta_{ij}\log(p_{ij}/\beta_{ij})+\beta_{ij}\log(1-\beta_{ij}))=\log Z_{MF}(p)+\sum_{(i,j)\in\mathcal{E}}\log(1-\beta_{ij}).\end{aligned}\tag{31}$$

On the other hand, applying the permanent to both sides of Equation (11) one arrives at

$$\mathrm{perm}(p)=\mathrm{perm}(\beta./(1-\beta))\left(\prod_{i\in\mathcal{V}_1}v_i\right)\left(\prod_{j\in\mathcal{V}_2}v^j\right).\tag{32}$$

Combining Equation (31) with Equation (32) results in Equation (12).

### C.2 Iterative Scheme for Solving Mean-Field Equations

An efficient heuristic way to find a (unique) solution of the MF system of Equations (11) for doubly stochastic matrix $\beta$ is to initialize with $v_i(0)=v^j(0)=1$ and iterate according to

$$\beta_{ij}(n+1)=\frac{p_{ij}}{p_{ij}+v_i(n)v^j(n)},\tag{33}$$

$$v_i(n+1)=v_i(n)\sum_j\beta_{ij}(n),\quad v^j(n+1)=v^j(n)\sum_i\beta_{ij}(n),\tag{34}$$

until the tolerance $\delta>\max(\mathrm{abs}(\beta(n+1)-\beta(n)))$ is met.

## Appendix D. FFE-based Approach

This Appendix provides some additional details on the FFE-based Approach.

### D.1 Exact Relations for Permanents

We present here a simple proof of Equation (15), which is a direct generalization of what was discussed above in Appendices B.1, C.1.

Weighting the logarithm of Equations (14) with that doubly stochastic matrix $\beta$ which minimizes Equation (15), summing the result over all the edges, and making use of Equations (14,13), one derives

$$\begin{aligned}
&\sum_{(i,j)\in\mathcal{E}} \beta_{ij}\log(w_i w_j) = \sum_{i\in\mathcal{V}_1}\log w_i + \sum_{j\in\mathcal{V}_2}\log w^j \\
&= \sum_{(i,j)\in\mathcal{E}} (\beta_{ij}\log(p_{ij}/\beta_{ij}) + \gamma\beta_{ij}\log(1-\beta_{ij})) \\
&= \log Z_f^{(\gamma)}(\beta|p) + \gamma\sum_{(i,j)\in\mathcal{E}}\log(1-\beta_{ij}).
\end{aligned} \tag{35}$$

On the other hand, applying the permanent to both sides of Equation (14) one arrives at

$$\mathrm{perm}(p) = \mathrm{perm}(\beta./(1-\beta).^{\gamma})\left(\prod_{i\in\mathcal{V}_1} w_i\right)\left(\prod_{j\in\mathcal{V}_2} w^j\right). \tag{36}$$

Combining Equation (35) with Equation (36) results in Equation (15).

### D.2 Iterative Scheme for Solving FFE-based Equations

All edge-local terms in the FFE-based function (13) are convex in $\beta\in[0;1]$ for $\gamma>0$, while for negative $\gamma$ the edge-term convexity holds only when all elements of $\beta$ are smaller than a threshold $\beta_c\geq 1/2$, which is a solution of $\beta_c\log(\beta_c) = -\gamma(1-\beta_c)\log(1-\beta_c)$. This suggests different iterative schemes for positive and negative $\gamma$.

When $\gamma>0$ we use the following modification of the MF scheme (33,34):

$$\begin{aligned}
&\beta_{ij}(n+1) = \frac{p_{ij}(1-\beta_{ij}(n))^{\gamma-1}}{p_{ij}(1-\beta_{ij}(n))^{\gamma-1} + w_i(n)w^j(n)}, \\
&w_i(n+1) = w_i(n)\sum_j \beta_{ij}(n), \quad w^j(n+1) = w^j(n)\sum_i \beta_{ij}(n),
\end{aligned}$$

In the case of $\gamma\leq 0$ we use the following modification of the BP scheme

$$\begin{aligned}
&\forall(i,j):\ \ \beta_{ij}(n+1) = \lambda\beta_{ij}(n) \\
&+\frac{(1-\lambda)p_{ij}(1+\beta_{ij}(n))^{1+\gamma}}{p_{ij}(1+\beta_{ij}(n))^{1+\gamma} + (\sum_k \beta_k^j(n)/2 + \sum_k \beta_i^k(n)/2 - \beta_{ij}(n))^2 (w_i(n)w^j(n))},
\end{aligned} \tag{37}$$

$$\begin{aligned}
&\forall i:\ \ w_i(n+1) = \frac{\sum_k p_{ik}(1+\beta_{ik}(n))^{1+\gamma}/w^k(n)}{1-\sum_j(\beta_{ij}(n))^2}, \\
&\forall j:\ \ w^j(n+1) = \frac{\sum_k p_{kj}(1+\beta_{kj}(n))^{1+\gamma}/w_k(n)}{1-\sum_i(\beta_{ij}(n))^2},
\end{aligned} \tag{38}$$

where the arguments of the $\beta$'s indicate the order of the iterations. The damping parameter $\lambda$ (typically chosen $0.4\div 0.5$) helps with convergence. To ensure appropriate accuracy for solutions with

β's close to zero or unity we also insert a normalization step after Equations (37) but prior to Equations (38), making the following two transformations consequently,

$$(a) \quad \forall(i,j): \quad \beta_{ij} \to \beta_{ij} / \sum_k \beta_{ik},$$
$$(b) \quad \forall(i,j): \quad \beta_{ij} \to \beta_{ij} / \sum_k \beta_{kj}.$$

The Algorithm is sensitive to initial values for β and $w$. To ensure convergence, we initiate the Algorithm with the output of the MF scheme (which converges much more easily) described in Appendix C.2, that is, $\beta(0) = \beta_{MF}$ and $w(0) = v_{MF}$. Numerical experiments show that this procedure converges to a stationary point of the fractional FE (13). We also verified that the iterative scheme designed for $\gamma < 0$ converges in the $\gamma > 0$ case, even thought the former scheme is obviously faster.

Note that FFE-based version of the standard sum-product Algorithm (SPA) can be developed. It is also natural to expect, in view of the general convexity of the fractional FE discussed in the main body of the text, that there exists a provably convergent version of the SPA. It will be important to design such a convergent γ-SPA in the future and to compare its practical performance against one of the heuristics described above.

## Appendix E. BP Gives Lower Bound on the Permanent

Here we give our version of the proof of the lower bound (16). First of all, in the case when the Bethe FE reaches its minimum in the interior of the domain, that is, at $\beta \in \mathcal{B}_p$, Equation (16) follows directly from the main result of Watanabe and Chertkov (2010), that is, Equation (7), and Schijver's inequality (17). Therefore, according to explanations of Section 3.1.1, we only need to analyze the case when the minimum of the Bethe FE is a partially resolved solution, with a β which can be split by appropriate permutations of rows and columns of the matrix into a perfect matching block (corresponding to a corner of the respective projected polytope), the block with all elements smaller than unity and nonzero unless the respective element of $p$ is zero (thus lying in the interior of the respective subspace), and all cross elements of β (between the blocks) equal to zero. Then, $Z_{o-BP}$ for such a partially resolved solution is split into the product of two contributions, $Z_{o-BP} = Z_{pm} \cdot Z_{int}$, where $Z_{pm}$ corresponds to the perfect matching block, and $Z_{int}$ corresponds to the interior block. In fact, $Z_{pm}$ is equal to the weighted perfect matching block of $p$ and $-\log(Z_{int})$ corresponds to the minimum of the Bethe FE computed for the interior block of $p$. On the other hand the full partition function, $Z$, can be bounded from below by the product $Z \geq Z_1 \cdot Z_2$, where $Z_1$ and $Z_2$ are permanents of the first and second blocks of the original matrix $p$. (Thus contributions of all the cross-terms of $p$ into $Z$ are ignored.) However, $Z_1 \geq Z_{pm}$, as counting only one perfect matching (and ignoring others), and $Z_2 \geq Z_{int}$ in accordance to what was already shown above for any minimum of Bethe FE achieved in the interior of the respective domain.

## Appendix F. Pruning of the Matrix

Computing the permanent of sufficiently dense matrices exactly with the ZDD approach explained in Appendix G is infeasible for $n > 30$. To overcome this difficulty we choose to sparsify dense matrices generated in one of our experimental ensembles, removing their less significant entries in the following steps. First, we use LP, described in Appendix A.2, to find the permutation corresponding to the maximum perfect matching. To avoid getting a zero permanent in the result, we include

all components of the maximum perfect matching permutation in the pruned matrix. Second, we consider every other entry of the matrix (not contributing to the maximum perfect matching) and keep it in the matrix only if it is included in a perfect matching which is close to the maximum perfect matching, that is, the two permutations share all but a few of their entries and the ratio of their weighted contributions (in the permanent) is larger than a pre-defined value. Then, we act according to either of the two strategies, both of which are explored in this manuscript. One strategy is to include all permutations whose products are more than a given fraction of the main permutation. This method will tend to reduce the fluctuations in the error of the pruned matrix (i.e., will reduce the variation in $Z_{\text{pruned}}/Z_{\text{original}}$). The other method is to always prune a set fraction of entries from the matrix, and prune them in order of decreasing value as determined by the above criterion. This method will reduce the fluctuations in the runtime of the Algorithm.

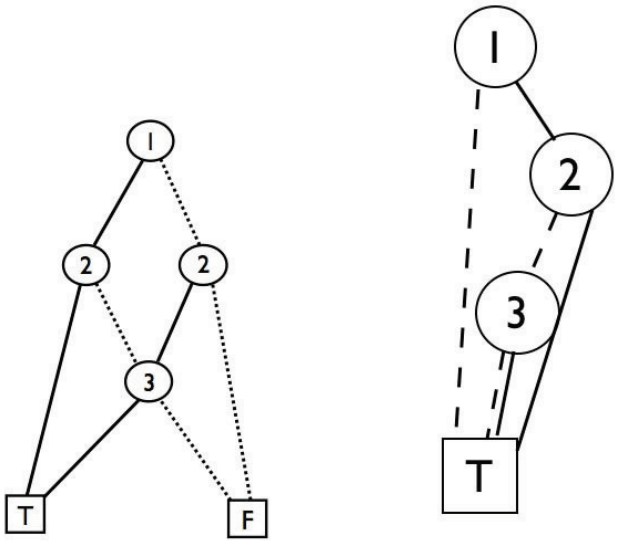
## Appendix G. Zero-suppressed Binary Decision Diagrams Method

Zero-suppressed binary decision diagrams, or ZDDs, are a tool useful for representing combinatorial problems. The concept was introduced by Shin-Ichi Minato in 1993 (Minato, 1993). The idea of ZDD is as follows: if one defines a combinatorial problem to be a function of many variables, each taking values in $\{0, 1\}$, with the value of the function itself being also in $\{0, 1\}$, then those sequences of inputs that lead to unity can be thought of as "solutions" to the problem. Furthermore, each solution can be described in terms of the input variables within it that are equal to unity. The problem, then, can be described as being a "family of sets," or set of sets, where the family is of all solutions to the problem and each set within the family is the set of input variables whose value is 1 in that solution.

To give an example of the "family of sets" concept, consider the $\overline{\text{XOR}}$ function, which returns 1 if and only if the inputs are equal. This function can also be represented as the family of sets $\{\emptyset, \{1,2\}\}$, where 1 and 2 correspond to inputs 1 and 2 to the function, because if the function is to have value 1 then either both inputs must be equal to 1 or neither must be. Once this has been understood, it is best to see the ZDD as nothing more than a concise representation of this family of sets, since the family can get quite cumbersome for problems with many solutions and many variables. Note that this system of representing problems provides the greatest improvement when there are few solutions, and when the solutions themselves are sparse, since the family of sets is then small. Correspondingly, ZDDs are most efficient under these conditions.

The actual format of a ZDD is that of a directed tree of nodes, with each node having a directed edge to two other nodes. Each edge emanating from a node has an identity, in that it is either a high "HI" branch or a low "LO" branch, and of the two edges emanating from each node, there must be exactly one "HI" branch and one "LO" branch. Each node also has an identity, a number from 1 to $n$ if there are $n$ inputs to the combinatorial problem. The tree must contain one or two special nodes, or "sinks", one of which is the "true" sink, and optionally the "false" sink. We also introduce the conventions that nodes can only point to nodes of higher identity than themselves and that no two nodes can be identical in both their identity and their LO and HI pointers.

Each node in a ZDD represents a choice about the variable the node identifies. If one begins at the top node of a ZDD, taking the HI branch represents including the variable represented by the node's identity in a prospective solution, and taking the LO branch represents not including that variable. If a LO or HI branch points to the true sink, that implies that a solution is reached if and only if all variables with identity greater than the current node identity are not included. If a



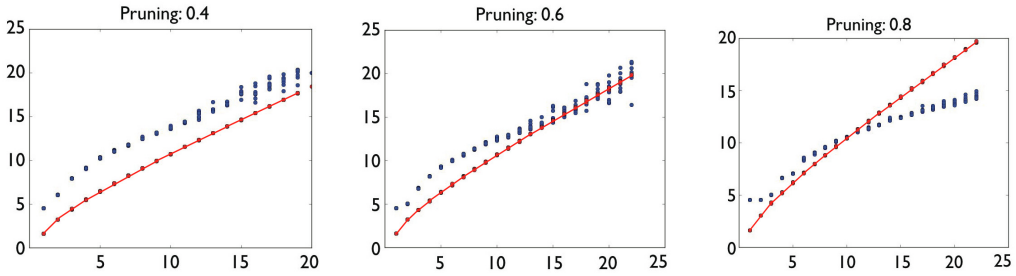

(a) (b)

Figure 13: Two simple ZDD diagrams discussed in the text.

LO branch points to the false sink, that implies that no solution is possible given the choices made previously. Interestingly, the constraints introduced in the paragraph previous to this one imply that a HI branch can never point to the False sink.

ZDDs are best understood with examples. The first example, also illustrated in Figure 13(a), is of the ZDD for the exactly-two function of three variables, in other words, the function that returns 1 if exactly two of its three inputs have value 1 and 0 otherwise. It can also be described as the family of sets $\{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$. Here, a dotted line denotes a LO branch and a normal line denotes a HI branch. Furthermore, the T and F symbols denote the true and false sinks, respectively, and the numbers inside each node refer to the nodes' identities (the variables that they represent). Our second example, shown in Figure 13(b), represents the family of sets $\{\emptyset, \{1\}, \{1,2\}, \{1,3\}\}$. Note the absence of a False sink. Note, also, the fact that a node's HI and LO branches need not point to the different locations. A more in-depth exploration of the ZDD concept can be found in Knuth (2009).

Once the basic concept of ZDDs is introduced, one can use it for solving various combinatorial problems, for example, to represent a permanent as a ZDD in order to use the method. When we apply ZDDs to the computations of permanent, we classify each entry of the matrix as either zero or nonzero. Then, we define a variable for each nonzero entry in the matrix. Each solution of our resulting ZDD will represent a possible permutation, meaning a set of entries in the matrix such that exactly one entry in each row and column is included in the set. There is a recursive Algorithm, suggested in Knuth (2009), that allows for efficient counting of the solutions of the ZDD. The Algorithm is simple: the number of solutions of a ZDD rooted at a node is equal to the sum of the numbers of solutions of the ZDD rooted at the HI and LO children of that node. The True sink is defined as having 1 solution, and the False sink as having 0. Note that the number of solutions of




(a) pruning 40% (b) pruning 60% (c) pruning 80%

Figure 14: Scatter plot of the number of memory accesses required to exactly compute the permanent of a sparse matrix for instances with different degree of pruning. Red and blue dots mark results of the Ryser formula and of the ZDD-based method, respectively. The numbers of memory accesses required to compute Ryser's formula at each trial are closely clumped, and so we link their results by a red line. The ZDD method's performance is less predictable, so we do not draw a line through its scatter plot.

a ZDD representing a matrix is equal to the permanent of the corresponding to $0-1$ matrix, with each 1 corresponding a nonzero entry.

In order to find the permanent of matrices that are not 0-1 matrices, only a small modification is necessary. Instead of purely counting solutions of the ZDD, we do a weighted count, where the weighted number of solutions of a ZDD rooted at a node is equal to the value of the corresponding matrix entry times the weighted number of solutions at the HI child added to the weighted number of solutions at the LO child. In other words, if we are considering a node $n$ with children HI and LO whose identity corresponds to a matrix entry of nonzero value $v$, then

$$\text{WeightedCount}(n) = v \cdot \text{WeightedCount}(\text{HI}) + \text{WeightedCount}(\text{LO}).$$

The WeightedCount of the root node of the ZDD will be equal to the permanent of the corresponding matrix.

This leaves the question of how to build the ZDD from the matrix. This is done using Knuth's "melding" Algorithm. The Algorithm is somewhat complex and will not be described here, but it is described in detail in Knuth (2009). The melding Algorithm is an efficient and systematic method for constructing larger ZDD out of the logical combination of smaller ones. The smallest ZDD being melded together using Knuth's Algorithm are ZDD representing the "exactly-one" constraint for each row and column of the matrix; in other words, they are constraints requiring exactly one matrix entry in every row and column to be included in a permutation which will be a "solution" to our problem.

## Appendix H. Comparison of Ryser's Formula with the ZDD-based Method

As part of our experiments we compared the speed of Ryser's formula with the speed of the ZDD-based method by counting memory accesses in each of the two Algorithms in order to fairly compare them. We found that the values we got for memory accesses were strongly correlated with the actual speed of the Algorithm. We found that for very dense matrices, Ryser's formula is faster,

but for sparser matrices the ZDD-based method is faster. We performed experiments with matrices that were 20%, 40%, and 60% sparse in order to get a good idea of the point where the ZDD-based method starts outperforming Ryser's formula. (Naturally, with no pruning, Ryser's formula outperforms the ZDD-based method significantly.)

As can be seen from Figure 14, the ZDD-based method begins outperforming Ryser's formula when matrices are around 60% sparse.

## References

M. Bayati and C. Nair. A rigorous proof of the cavity method for counting matchings. In *Proc. 44th Allerton Conf. on Communications, Control, and Computing*, 2006.

M. Bayati, D. Shah, and M. Sharma. Max-product for maximum weight matching: Convergence, correctness, and LP duality. *IEEE Transactions on Information Theory*, 54(3):1241–1251, 2008.

D.P. Bertsekas. Auction algorithms for network flow problems: a tutorial introduction. *Comput. Optimiz. Applic.*, 1:7–66, 1992.

H.A. Bethe. Statistical theory of superlattices. *Proceedings of Royal Society of London A*, 150:552, 1935.

G. Birkhoff. Three observations on linear algebra. *Univ. Nac. Tacuman Rev. Ser. A*, 5:147–151, 1946.

M. Chertkov. Exactness of belief propagation for some graphical models with loops. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10016 (12pp), 2008. URL `http://stacks.iop.org/1742-5468/2008/P10016`.

M. Chertkov and V. Y. Chernyak. Loop calculus in statistical physics and information science. *Phys. Rev. E*, 73(6):065102, Jun 2006a. doi: 10.1103/PhysRevE.73.065102.

M. Chertkov and V.Y. Chernyak. Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(06):P06009, 2006b. URL `http://stacks.iop.org/1742-5468/2006/i=06/a=P06009`.

M. Chertkov, L. Kroc, and M. Vergassola. Belief propagation and beyond for particle tracking. *arxiv*, abs/0806.1199, 2008.

M. Chertkov, L. Kroc, F. Krzakala, M. Vergassola, and L. Zdeborova. Inference in particle tracking experiments by passing messages between images. *Proceedings of the National Academy of Science*, 107:7663–7668, April 2010. doi: 10.1073/pnas.0910994107.

B. Cseke and T. Heskes. Properties of Bethe Free Energies and message passing in Gaussian models. *J. Artif. Intell. Res. (JAIR)*, 41:1–24, 2011.

K. Drescher, R. E. Goldstein, N. Michel, M. Polin, and I. Tuval. Direct measurement of the flow field around swimming microorganisms. *Phys. Rev. Lett.*, 105(16):168101, Oct 2010. doi: 10.1103/PhysRevLett.105.168101.

P.V. Egorychev. Proof of the van der Waerden conjecture for permanents. *Siberian Mathematical Journal*, 22(6):854–859, 1981. URL http://www.springerlink.com/content/k692377516k1x778/.

G. M. Engel and H. Schneider. Inequalities for determinants and permanents. *Linear and Multilinear Algebra*, 1:187–201, 1973.

D.I. Falikman. Proof of the van der Waerden conjecture regarding the permanent of a doubly stochastic matrix. *Mathematical Notes*, 29(6):475–479, 1981. URL http://www.springerlink.com/content/h41162g677317110/.

R.G. Gallager. *Low-Density Parity-Check Codes*. MIT Press, Cambridge, MA, 1963.

J. S. Guasto, K. A. Johnson, and J. P. Gollub. Oscillatory flows induced by microorganisms swimming in two dimensions. *Phys. Rev. Lett.*, 105(16):168102, Oct 2010. doi: 10.1103/PhysRevLett.105.168102.

L. Gurvits. Van der Waerden/Schrijver-Valiant like conjectures and stable (aka hyperbolic) homogeneous polynomials: one theorem for all. *Electronic Journal of Combinatorics*, 15:R66, 2008. URL http://www.emis.ams.org/journals/EJC/Volume_15/PDF/v15i1r66.pdf.

L. Gurvits. Unharnessing the power of Schrijver's permanental inequality. *ArXiv e-prints*, June 2011.

B. Huang and T. Jebara. Approximating the permanent with belief propagation, arxiv:0908.1769, 2009. URL http://arxiv.org/abs/0908.1769.

M. Huber and J. Law. Fast approximation of the permanent for very dense problems. *in: SODA 08: Proc. 19th ACM-SIAM Sympos. on Discrete Algorithms*, page 681689, 2008.

M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *J. ACM*, 51(4):671–697, 2004. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/1008731.1008738.

D. Kőnig. *Theorie der Endlichen und Unendlichen Graphen*. Akademische Verlags Gesellschaft, Leipzig, 1936.

P. Knopp and R. Sinkhorn. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific J. Math.*, 21(2):343–348, 1967.

D. E. Knuth. *The Art of Computer Programming, Volume 4, Fascicle 1: Bitwise Tricks & Techniques; Binary Decision Diagrams*. Addison-Wesley Professional, 12th edition, 2009. ISBN 0321580508, 9780321580504.

H. W. Kuhn. The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly*, 2:8397, 1955.

M. Laurent and A. Schrijver. On Leonid Gurvits's proof for permanents. *American Mathematical Monthly*, 117(10), 2010. URL http://homepages.cwi.nl/~lex/files/perma5.pdf.

N. Linial, A. Samorodnitsky, and A. Wigderson. A deterministic strongly polynomial algorithm for matrix scaling and approximate permanents. *Combinatorica*, 20(4):545–568, 2000.

S. Minato. Zero-suppressed BDDs for set manipulation in combinatorial problems. In *Design Automation, 1993. 30th Conference on*, pages 272 – 277, 1993. doi: 10.1109/DAC.1993.203958.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco: Morgan Kaufmann Publishers, Inc., 1988.

H.A. Peierls. Ising's model of ferromagnetism. *Proceedings of Cambridge Philosophical Society*, 32:477–481, 1936.

H. J. Ryser. Combinatorial mathematics. *The Carus Mathematical Monographs*, 14, 1963.

S. Sanghavi, D. Malioutov, and A. Willsky. Belief propagation and lp relaxation for weighted matching in general graphs. *Information Theory, IEEE Transactions on*, 57(4):2203 –2212, april 2011. ISSN 0018-9448. doi: 10.1109/TIT.2011.2110170.

A. Schrijver. Counting 1-factors in regular bipartite graphs. *Journal of Combinatorial Theory*, 72: 122–135, 1998. doi: 10.1006/jctb.1997.1798.

TheCodeProject. `http://www.codeproject.com/KB/applications/RyserPermanent.aspx`. Computes permanent of a matrix with Ryser's algorithm.

L.G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8: 189201, 1979.

B. L. van der Waerden. *[Aufgabe] 45, Jahresbericht der Deutschen Mathematiker-Vereinigung*, 35: 117, 1926.

J. von Neumann. A certain zero-sum two-person game equivalent to an optimal assignment problem. *Ann. Math. Studies*, 28:5–12, 1953.

P.O. Vontobel. The Bethe permanent of a non-negative matrix. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, pages 341 –346, 29 2010-oct. 1 2010. doi: 10.1109/ALLERTON.2010.5706926.

P.O. Vontobel. The bethe permanent of a non-negative matrix. *IEEE Trans. Inf. Theory*, 59(3): 1866–1901, 2013.

Y. Watanabe and M. Chertkov. Belief propagation and loop calculus for the permanent of a non-negative matrix. *Journal of Physics A: Mathematical and Theoretical*, 43(24):242002, 2010. URL `http://stacks.iop.org/1751-8121/43/i=24/a=242002`.

W. Wiegerinck and T. Heskes. Fractional belief propagation. *Advances in Neural Information Processing Systems 15*, 12:438–445, 2003.

A.B. Yedidia. Counting independent sets and kernels of regular graphs. *arxiv*, abs/0910.4664, 2009.

J.S. Yedidia, W.T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282 – 2312, July 2005. ISSN 0018-9448. doi: 10.1109/TIT.2005.850085.

Journal of Machine Learning Research 14 (2013) 2067-2118 Submitted 7/12; Revised 4/13; Published 7/13

# Construction of Approximation Spaces for Reinforcement Learning

**Wendelin Böhmer** WENDELIN@NI.TU-BERLIN.DE
*Neural Information Processing Group*
*Technische Universität Berlin*
*Marchstrasse 23, Berlin 10587, Germany*

**Steffen Grünewälder** STEFFEN@CS.UCL.AC.UK
*Centre for Computational Statistics and Machine Learning*
*University College London*
*London WC1E 6BT, United Kingdom*

**Yun Shen** YUN@NI.TU-BERLIN.DE
*Neural Information Processing Group*
*Technische Universität Berlin*
*Marchstrasse 23, Berlin 10587, Germany*

**Marek Musial** MUSIAL@CS.TU-BERLIN.DE
*Robotics Group*
*Technische Universität Berlin*
*Einsteinufer17, Berlin 10587, Germany*

**Klaus Obermayer** OBY@NI.TU-BERLIN.DE
*Neural Information Processing Group*
*Technische Universität Berlin*
*Marchstrasse 23, Berlin 10587, Germany*

**Editor:** Sridhar Mahadevan

## Abstract

Linear *reinforcement learning* (RL) algorithms like *least-squares temporal difference learning* (LSTD) require *basis functions* that span *approximation spaces* of potential value functions. This article investigates methods to construct these bases from samples. We hypothesize that an ideal approximation spaces should encode *diffusion distances* and that *slow feature analysis* (SFA) constructs such spaces. To validate our hypothesis we provide theoretical statements about the LSTD value approximation error and induced metric of approximation spaces constructed by SFA and the state-of-the-art methods *Krylov bases* and *proto-value functions* (PVF). In particular, we prove that SFA minimizes the average (over all tasks in the same environment) bound on the above approximation error. Compared to other methods, SFA is very sensitive to sampling and can sometimes fail to encode the whole state space. We derive a novel *importance sampling* modification to compensate for this effect. Finally, the LSTD and *least squares policy iteration* (LSPI) performance of approximation spaces constructed by Krylov bases, PVF, SFA and PCA is compared in benchmark tasks and a visual robot navigation experiment (both in a realistic simulation and with a robot). The results support our hypothesis and suggest that (i) SFA provides *subspace-invariant* features for MDPs with *self-adjoint* transition operators, which allows strong guarantees on the approximation error, (ii) the modified SFA algorithm is best suited for LSPI in both discrete and continuous state spaces and (iii) approximation spaces encoding diffusion distances facilitate LSPI performance.

**Keywords:** reinforcement learning, diffusion distance, proto value functions, slow feature analysis, least-squares policy iteration, visual robot navigation

©2013 Wendelin Böhmer, Steffen Grünewälder, Yun Shen, Marek Musial and Klaus Obermayer.

## 1. Introduction

*Reinforcement learning* (RL, Sutton and Barto, 1998; Bertsekas and Tsitsiklis, 1996) provides a framework to autonomously learn *control policies* in stochastic environments and has become popular in recent years for controlling robots (e.g., Abbeel et al., 2007; Kober and Peters, 2009). The goal of RL is to compute a policy which selects *actions* that maximize the *expected future reward* (called *value*). An agent has to make these decisions based on the *state* $\boldsymbol{x} \in \mathcal{X}$ of the system. The state space $\mathcal{X}$ may be finite or continuous, but is in many practical cases too large to be represented directly. *Approximated RL* addresses this by choosing a function from *function set* $\mathcal{F}$ that resembles the true value function. Many function sets $\mathcal{F}$ have been proposed (see, e.g., Sutton and Barto, 1998; Kaelbling et al., 1996, for an overview). This article will focus on the space of *linear functions* with $p$ non-linear *basis functions* $\{\phi_i(\cdot)\}_{i=1}^p$ (Bertsekas, 2007), which we call *approximation space* $\mathcal{F}_\phi$.

The required basis functions $\phi_i(\cdot)$ are usually defined by hand (e.g., Sutton, 1996; Konidaris et al., 2011) and a bad choice can critically impede the accuracy of both the value estimate and the resulting control policy (see, e.g., Thrun and Schwartz, 1993). To address this issue, a growing body of literature has been devoted to the *construction* of basis functions and their theoretical properties (Mahadevan and Maggioni, 2007; Petrik, 2007; Parr et al., 2007; Mahadevan and Liu, 2010; Sun et al., 2011). Recently, the unsupervised method *slow feature analysis* (SFA, Wiskott and Sejnowski, 2002) has been proposed in this context (Legenstein et al., 2010; Luciw and Schmidhuber, 2012). This article presents a theoretical analysis of this technique and compares it with state-of-the-art methods. We provide theoretical statements for two major classes of automatically constructed basis functions (*reward-based* and *subspace-invariant* features, Parr et al., 2008) with respect to the induced Euclidean metric and the approximation error of *least-squares temporal difference learning* (LSTD, Bradtke and Barto, 1996). We also prove that under some assumptions SFA minimizes an average bound on the approximation error of all tasks in the same environment and argue that no better solution based on a single training sequence exists.

In practical applications (such as robotics) the state can not always be observed directly, but may be deducted from *observations*[1] $\boldsymbol{z} \in \mathcal{Z}$ of the environment. *Partial observable Markov decision processes* (POMDPs, Kaelbling et al., 1998) deal with the necessary inference of hidden states from observations. POMDPs are theoretically better suited, but become quickly infeasible for robotics. In contrast, this article focuses on another obstacle to value estimation: the *metric* associated with observation space $\mathcal{Z}$ can influence basis function construction. We assume for this purpose a unique one-to-one correspondence[2] between states $\boldsymbol{x} \in \mathcal{X}$ and observations $\boldsymbol{z} \in \mathcal{Z}$. To demonstrate the predicted effect we evaluate construction methods on a robotic visual navigation task. The observations are first-person perspective images, which exhibit a very different Euclidean metric than the underlying state of robot position and orientation. We hypothesize that *continuous SFA* (RSK-SFA, Böhmer et al., 2012) is not severely impeded by the change in observation metric and substantiate this in comparison to *continuous proto value functions* (PVF, Mahadevan and Maggioni, 2007) and *kernel PCA* (Schölkopf et al., 1998). We also confirm theoretical predictions that SFA is sensitive to the sampling policy (Franzius et al., 2007) and derive an *importance sampling* modification to compensate for these imbalances.

1. In this article the set of all possible observations $\mathcal{Z}$ is assumed to be a manifold in vector space $\mathbb{R}^d$.
2. This makes $\mathcal{Z}$ an isomorphism of $\mathcal{X}$, embedded in $\mathbb{R}^d$. The only difference is the associated *metric*. In the following we will continue to discriminate between $\mathcal{X}$ and $\mathcal{Z}$ for illustrative purposes.

### 1.1 Approximation Spaces

State spaces $\mathcal{X}$ can be finite or continuous.[3] We define the corresponding observation space $\mathcal{Z}$ to be isomorphic, but it may be governed by a very different *metric*. For example: finite $\mathcal{X}$ are usually equipped with a *discrete metric*, in which all states have equal distance to each other. Isomorphic observations $\boldsymbol{z} \in \mathcal{Z} \subset \mathbb{R}^d$, on the other hand, might be equipped with an Euclidean metric in $\mathbb{R}^d$ instead. To approximate the *value function* $V(\cdot) : \mathcal{Z} \to \mathbb{R}$, one aims for a function $f(\cdot) \in \mathcal{F}$ that minimizes the *approximation error* w.r.t. some norm $\|V - f\|$. This article is focusing on the Euclidean $L_2$ norm[4] (see Section 2 for details), which depends on the metric's distance function. Besides different approximation errors, *generalization* to unseen states will also be very different in these spaces. This raises the question which metric is best suited to approximate value functions. Values are defined as the expected sum of *future* rewards. States with similar futures will therefore have similar values and are thus close by under an ideal metric. *Diffusion distances* compare the probabilities to end up in the same states (see, e.g., Coifman et al., 2005, and Section 4.1). It sands therefore to reason that a diffusion metric facilitates value approximation.

This article is using the term *approximation space* $\mathcal{F}_{\boldsymbol{\phi}}$ for the set of linear functions with $p$ non-linear basis functions $\phi_i : \mathcal{Z} \to \mathbb{R}$, $\mathcal{F}_{\boldsymbol{\phi}} := \left\{ f(\cdot) = \boldsymbol{w}^\top \boldsymbol{\phi}(\cdot) \,|\, \boldsymbol{w} \in \mathbb{R}^p \right\}$. Function approximation can be essentially performed by an inverse of the covariance matrix (see Section 2.3) and value estimation can be guaranteed to converge (Bertsekas, 2007). Nonetheless, the choice of basis functions $\boldsymbol{\phi} : \mathcal{Z} \to \mathbb{R}^p$ and thus approximation space $\mathcal{F}_{\boldsymbol{\phi}}$ will strongly affect approximation quality and generalization to unseen samples. An ideal approximation space should therefore (i) be able to approximate the value function well and (ii) be equipped with a Euclidean metric in $\{\boldsymbol{\phi}(\boldsymbol{z}) \,|\, \boldsymbol{z} \in \mathcal{Z}\}$ that resembles a diffusion metric. Approximation theory provides us with general functional bases that allow arbitrarily close approximation of continuous functions and thus fulfill (i), for example polynomials or a Fourier basis (Konidaris et al., 2011). However, those bases can usually not be defined on high-dimensional observation spaces $\mathcal{Z}$, as they are prone to the *curse of dimensionality*.

A straightforward approach to basis construction would extract a low-dimensional manifold of $\mathcal{Z}$ and construct a general function base on top of it. This can be achieved by manifold extraction (Tenenbaum et al., 2000; Jenkins and Mataric, 2004) or by computer vision techniques (e.g., Visual SLAM, Smith et al., 1990; Davison, 2003), which require extensive knowledge of the latent state space $\mathcal{X}$. Some approaches construct basis functions $\boldsymbol{\phi}(\cdot)$ directly on the observations $\boldsymbol{z} \in \mathcal{Z}$, but are either restricted to linear maps $\boldsymbol{\phi}(\cdot)$ (PP, Sprague, 2009) or do not generalize to unseen samples (ARE, Bowling et al., 2005). None of the above methods extracts $\mathcal{X}$ in a representation that encodes a diffusion metric.

Recent analysis of the approximation error has revealed two opposing approaches to basis construction: *reward-based* and *subspace-invariant* features (Parr et al., 2008). The former encode the propagated reward function and the latter aim for eigenvectors of the transition matrix. Section 3.1 provides an overview of the reward-based *Krylov bases* (Petrik, 2007), *Bellman error basis functions* (BEBF, Parr et al., 2007), *Bellman average reward bases* (BARB, Mahadevan and Liu, 2010) and *Value-function of the Bellman error* bases (V-BEBF, Sun et al., 2011). All of these algorithms are defined exclusively for finite state spaces. The encoded metric is investigated in Section 4.1. *Proto-value functions* (PVF, Mahadevan and Maggioni, 2007, and Section 3.3) are the state-of-the-

3. This article does not discuss discrete countable infinite state spaces, which are isomorphisms to $\mathbb{N}$.

4. Other approaches are based on the $L_\infty$ norm (Guestrin et al., 2001; Petrik and Zilberstein, 2011) or the $L_1$ norm (de Farias and Roy, 2003). However, all norms eventually depend on the metric of $\mathcal{X}$ or $\mathcal{Z}$.

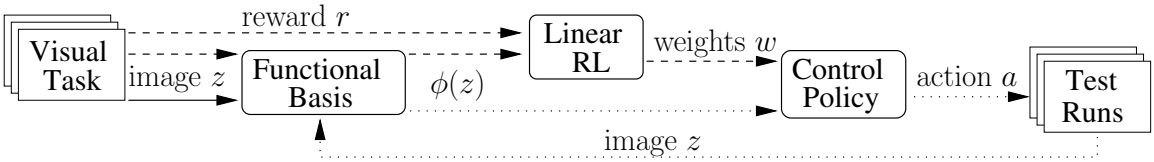



Figure 1: Scheme of a general RL architecture for visual tasks. First (solid arrow) one or many visual tasks generate images $z$ to train a *representation* $\phi : \mathcal{Z} \to \mathbb{R}^p$, which is a *functional basis* in the true state space $\mathcal{X}$. Given such a basis (dashed) one task is used to train a *control policy* with a *linear RL* algorithm. In the verification phase (dotted) the trained control policy generates multiple test trajectories.

art subspace-invariant feature construction method. In finite state spaces PVF are the eigenvectors to the smallest eigenvalues of the normalized graph Laplacian of an undirected graph representing the *transition possibility* (not *probability*) between states. As the calculation requires no knowledge of the reward, this technique has proven useful to transfer knowledge between different tasks in the same environment (Ferguson and Mahadevan, 2006; Ferrante et al., 2008). In Section 4.3 we will explain this observation by defining the class of learning problems for which this transfer is nearly optimal. To cope with continuous state or observation spaces, there also exists an extension based on the PVF of a k-nearest neighbors graph and Nyström approximation between the graph nodes (Mahadevan and Maggioni, 2007). However, as this approach is based on neighborhood relationships in $\mathcal{Z}$, the solution will not preserve diffusion distances.

An extension preserving these distances are Laplacian eigenmaps (Belkin and Niyogi, 2003) of the *transition operator*. Recently Sprekeler (2011) has shown that *slow feature analysis* (SFA, Wiskott and Sejnowski, 2002, and Section 3.4) approximates Laplacian eigenmaps. In the limit of an infinite training sequence, it can be shown (under mild assumptions) that the resulting non-linear SFA features span a Fourier basis in the unknown state space $\mathcal{X}$ (Wiskott, 2003). Franzius et al. (2007) show additionally that the *order* in which the basis functions are encoded is strongly dependent on the *relative velocities* in different state dimensions. This can lead to an insufficient approximation for *low dimensional*, but has little effect on *high dimensional* approximation spaces. Section 3.5 addresses this problem with an *importance sampling* modification to SFA.

### 1.2 Visual Tasks

Most benchmark tasks in RL have either a finite or a continuous state space with a well behaving Euclidean metric.[5] Theoretical statements in Section 4 predict that SFA encodes diffusion distances, which are supposed to facilitate generalization. Testing this hypothesis requires a task that can be solved either with a well behaving true state $x \in \mathcal{X}$ or based on observations $z \in \mathcal{Z} \subset \mathbb{R}^d$ with a disadvantageous metric. Value functions approximated w.r.t. a diffusion metric, that is, with SFA features, should provide comparable performance in both spaces. Based on a method that encodes only Euclidean distances in $\mathcal{Z}$ (e.g., PCA), on the other hand, the performance of the approximated value functions should differ.

5. An example for finite state spaces is the *50-state chain* (Section 5.2). The *puddle-world task* (Section 5.3) and the *mountain-car task* (not evaluated) have been defined with continuous and with discrete state spaces (Boyan and Moore, 1995; Sutton, 1996). Both continuous tasks are defined on well-scaled two dimensional state spaces. Euclidean distances in these spaces resemble diffusion distances closely.

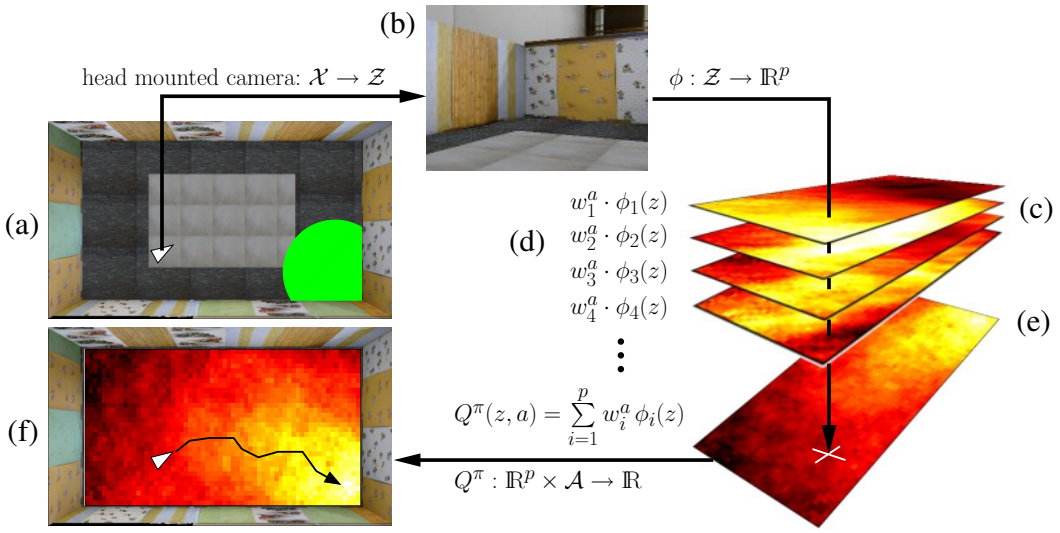



Figure 2: The figure shows the visual control process to guide a robot into the circular goal area. At its position (a), the robot observes an image with its head mounted camera (b). A function $\phi(\cdot)$, generated by one of the discussed unsupervised methods, maps the image into a $p$-dimensional feature space (c). For each action $a \in \mathcal{A}$, these features are weighted by the LSPI parameter vector $\boldsymbol{w}^a \in \mathbb{R}^p$ (d), giving rise to the Q-value function $Q^{\boldsymbol{\pi}}(\cdot,\cdot)$ (e). The control always chooses the action $a$ with the highest Q-value (f).

Applied to visual input, for example camera images, this class of problems is called *visual tasks*. Setting problems of *partial observability* aside, the true state $\boldsymbol{x}$ is usually assumed to be sufficiently *represented* by a set of hand-crafted features of $\boldsymbol{z}$. However, there is no straightforward way to extract the state reliably out of visual data without introducing artificial markers to the environment. Current approaches to visual tasks aim thus to learn a feature mapping $\boldsymbol{\phi} : \mathcal{Z} \to \mathbb{R}^p$ from observations $\boldsymbol{z}$, without loosing too much information about the true state $\boldsymbol{x}$ (see Jodogne and Piater, 2007, for an overview). Figure 1 shows a sketch of a general RL architecture to solve visual tasks with linear RL methods. Here we first learn an *image representation* $\boldsymbol{\phi} : \mathcal{Z} \to \mathbb{R}^p$ from experience (solid arrow), collected in one or more visual tasks within the same environment. To learn a *control policy* (dashed arrows) the agent treats the representation $\boldsymbol{\phi}(\boldsymbol{z}) \in \mathbb{R}^p$ of each observed image $\boldsymbol{z} \in \mathcal{Z}$ as the representation of the corresponding state $\boldsymbol{x} \in \mathcal{X}$. A linear RL algorithm can estimate future *rewards* $r \in \mathbb{R}$ by approximating the linear *Q-value function* $Q^{\boldsymbol{\pi}} : \mathbb{R}^p \times \mathcal{A} \to \mathbb{R}$ with *weight vector* $\boldsymbol{w} \in \mathbb{R}^{p|\mathcal{A}|}$. The control policy always chooses the *action* $a \in \mathcal{A}$ with the highest Q-value predicted by $Q^{\boldsymbol{\pi}}$ and can be verified by independent test runs from random start positions (dotted arrows). For example, in the context of navigation, Lange and Riedmiller (2010) employed a *deep auto-encoder* (Hinton and Osindero, 2006), Legenstein et al. (2010) hierarchical nonlinear *slow feature analysis* (SFA, Wiskott and Sejnowski, 2002) and Luciw and Schmidhuber (2012) *incremental SFA* (Kompella et al., 2012) to represent the underlying state space. The control problem was subsequently solved by different *approximate RL* algorithms. All above works verified

their approaches on a very regular observation space $\mathcal{Z}$ by providing the agent with a bird's eye view of an artificial world, in which a set of pixels determines the agents position uniquely.

To yield a less ideal observation space $\mathcal{Z}$, the visual navigation task in Section 5.4 observes first-person perspective images[6] instead. Figure 2 shows the control loop of the robot. The true state $\boldsymbol{x}$ is the robot's position at which an image $\boldsymbol{z}$ is taken by a head-mounted camera. $\mathcal{X}$ is continuous and in principle the actions $a \in \mathcal{A}$ should be continuous too. However, selecting continuous actions is not trivial and for the sake of simplicity we restricted the agent to three discrete actions: move forward and turn left or right.

### 1.3 Contributions

The contributions of this article are threefold:

1. We provide theoretical statements about the encoded *diffusion metric* (Section 4.1) and the LSTD value *approximation error* (Section 4.2) of both reward-based (Krylov bases) and subspace-invariant (SFA) features. We also prove that SFA minimizes an average bound on the approximation error of a particular set of tasks (Section 4.3). We conclude that *SFA can construct better approximation spaces for LSTD than PVF* and demonstrate this on multiple discrete benchmark tasks (Sections 5.1 to 5.3).

2. We investigate the role of the metric in approximation space $\mathcal{F}_\phi$ on a visual robot navigation experiment, both in a realistic simulation and on a robot (Sections 5.4 to 5.7). We demonstrate than SFA can sometimes fail to encode the whole state space due to its dependence on the sampling policy and address this problem with a novel *importance sampling* modification to the SFA algorithm.

3. We compare the performance of approximation spaces constructed by Krylov bases, PVF, SFA and PCA for *least-squares policy iteration* (LSPI, Lagoudakis and Parr, 2003). Results suggest that (i) the modified SFA algorithm is best suited for LSPI in both discrete and continuous state spaces and (ii) approximation spaces that encode a diffusion metric facilitate LSPI performance.

Both theoretical and empirical results leave room for interpretation and unresolved issues for future works. Section 6 discusses open questions as well as potential solutions. Finally, the main results and conclusions of this article are summarized in Section 7.

## 2. Reinforcement Learning

In this section we review *reinforcement learning* in potentially continuous state spaces $\mathcal{X}$, which require a slightly more complicated formalism than used in standard text books (e.g., Sutton and Barto, 1998). The introduced notation is necessary for Section 4 and the corresponding proofs in Appendix A. However, casual readers familiar with the RL problem can skip this section and still comprehend the more practical aspects of the article.

There exist many linear RL algorithms one could apply to our experiment, like *temporal difference learning* (TD($\lambda$), Sutton and Barto, 1998) or *Q-learning* (Watkins and Dayan, 1992). We

6. Rotating a camera by some degrees represents only a minor change in its orientation and therefore in $\mathcal{X}$, but shifts all pixels and can lead to very large Euclidean distances in $\mathcal{Z}$. Moving slightly forward, on the other hand, changes only the pixels of objects close by and thus yields a much smaller distance in $\mathcal{Z}$.

chose here the *least-squares policy iteration* algorithm (LSPI, Lagoudakis and Parr, 2003, see Section 2.4), because the underlying *least-squares temporal difference* algorithm (LSTD, Bradtke and Barto, 1996, see Section 2.3) is the most sample effective unbiased value estimator (Grünewälder and Obermayer, 2011). For practical implementation we consider *sparse kernel methods*, which are introduced in Section 2.5.

### 2.1 The Reinforcement Learning Problem

We start with the definition of a *Markov decision process* (MDP). Let $\mathcal{B}(\mathcal{X})$ denote the collection of all *Borel sets* of set $\mathcal{X}$. A *Markov decision process* is a tuple $(\mathcal{X}, \mathcal{A}, P, R)$. In our setup, $\mathcal{X}$ is a *finite* or *compact continuous*[7] state space and $\mathcal{A}$ the finite[8] action space. The *transition kernel*[9] $P : \mathcal{X} \times \mathcal{A} \times \mathcal{B}(\mathcal{X}) \to [0,1]$ represents the probability $P(A|x,a)$ to end up in set $A \in \mathcal{B}(\mathcal{X})$ after executing action $a$ in state $x$. $R : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \times \mathcal{B}(\mathbb{R}) \to [0,1]$ is a distribution over *rewards*: $R(B|x,a,y)$ is the probability to receive a reward within set $B \in \mathcal{B}(\mathbb{R})$ after a transition from state $x$ to state $y$, executing action $a$. In our context, however, we will be content with the mean *reward function* $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$, defined as $r(x,a) = \int_{\mathbb{R}} \int_{\mathcal{X}} r R(dr|x,a,y) P(dy|x,a), \forall x \in \mathcal{X}, a \in \mathcal{A}$. A control *policy* $\pi : \mathcal{X} \times \mathcal{B}(\mathcal{A}) \to [0,1]$ is a conditional distribution of actions given states. The goal of *reinforcement learning* is to find a policy that maximizes the *value* $V^\pi(x)$ at each state $x$, that is the expected sum of discounted future rewards

$$V^\pi(x_0) := \mathbb{E}\left[ \sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) \;\middle|\; \begin{array}{rcl} a_t & \sim & \pi(\cdot|x_t) \\ x_{t+1} & \sim & P(\cdot|x_t, a_t) \end{array} \right], \quad \forall x_0 \in \mathcal{X}.$$

Here the *discount factor* $\gamma \in [0,1)$ determines the relative importance of short term to long term rewards.[10] The value function can also be expressed *recursively*:

$$V^\pi(x) = \int r(x,a)\, \pi(da|x) + \gamma \iint V^\pi(y) P(dy|x,a)\, \pi(da|x), \quad \forall x \in \mathcal{X}.$$

In finite state (and action) spaces this equation can be solved by dynamic programming. Note that for fixed $V^\pi(\cdot)$ the equation is linear in the policy $\pi(\cdot|\cdot)$ and vice versa, allowing an *expectation maximization* type algorithm called *policy iteration* (PI, Sutton and Barto, 1998) to find the best policy. To allow for continuous state spaces, however, we need to translate this formalism into a Hilbert space.

### 2.2 MDP in Hilbert Spaces

For the sake of feasibility we will restrict our discussion to value functions $v^\pi \in L^2(\mathcal{X}, \xi)$ from the space of *square-integrable functions* on $\mathcal{X}$, endowed with probability measure $\xi : \mathcal{B}(\mathcal{X}) \to [0,1]$, $\int \xi(dx) = 1$. This Hilbert space contains $\xi$-measurable functions and should suffice for all

7. Compact state spaces $\mathcal{X}$ are necessary for *ergodicity*, see Footnote 12 on Page 2074. All finite $\mathcal{X}$ are compact.
8. For generality we maintain the notation of continuous compact action spaces as long as possible.
9. Following probability theory, a *kernel* denotes here a *conditional measure* over some set, in this article $\mathcal{X} \times \mathcal{A}$ or just $\mathcal{X}$. If this measure over the whole set is always one then it is called a *transition* or *Markov kernel*. Note that the *Radon-Nikodym derivative* of a kernel w.r.t. the uniform measure is called a *kernel function* in integral calculus. Note also the difference to *positive semi-definite kernels* in the context of RKHS (see Section 3.2).
10. In classical decision theory, $\gamma$ can be interpreted as the continuous version of a maximal search depth in the decision tree. Alternatively, one can see $\gamma^t$ as shrinking certainty about predicted rewards.

continuous setups. The induced *inner product* and *norm* are

$$\langle f,g\rangle_\xi = \int f(x)\,g(x)\,\xi(dx) \quad \text{and} \quad \|f\|_\xi = \langle f,f\rangle_\xi^{1/2}, \quad \forall f,g \in L^2(\mathcal{X},\xi)\,.$$

For a fixed policy $\pi$, this yields the *transition operator*[11] $\hat{P}^\pi : L^2(\mathcal{X},\xi) \to L^2(\mathcal{X},\xi)$,

$$\hat{P}^\pi[f](x) := \iint f(y)\,P(dy|x,a)\,\pi(da|x)\,, \quad \forall x \in \mathcal{X}\,, \quad \forall f \in L^2(\mathcal{X},\xi)\,.$$

The operator is called *ergodic* if every Markov chain sampled by the underlying transition kernel $P$ and policy $\pi$ is ergodic.[12] This is a convenient assumption as it implies the existence of a *steady state distribution* $\xi$, which we will use as measure of $L^2(\mathcal{X},\xi)$. This also implies

$$\xi(B) = \iint P(B|x,a)\,\pi(da|x)\,\xi(dx)\,, \quad \forall B \in \mathcal{B}(\mathcal{X})\,.$$

Under the assumption that all rewards are *bounded*, that is, $|r(x,a)|^2 < \infty \Rightarrow \exists r^\pi \in L^2(\mathcal{X},\xi)$ : $r^\pi(x) := \int r(x,a)\,\pi(da|x), \forall x \in \mathcal{X}, \forall a \in \mathcal{A}$, we can define the *Bellman operator* in $L^2(\mathcal{X},\xi)$

$$\hat{B}^\pi[f](x) := r^\pi(x) + \gamma \hat{P}^\pi[f](x)\,, \quad \forall x \in \mathcal{X}\,, \quad \forall f \in L^2(\mathcal{X},\xi)\,,$$

which performs *recursive value propagation*. This is of particular interest as one can show[13] that $\hat{B}^\pi[f]$ is a contract mapping in $\|\cdot\|_\xi$ and an infinite application starting from any function $f \in L^2(\mathcal{X},\xi)$ converges to the *true* value function $v^\pi \in L^2(\mathcal{X},\xi)$.

### 2.3 Least-squares Temporal Difference Learning

Infinitely many applications of the Bellman operator $\hat{B}^\pi[\cdot]$ are not feasible in practice. However, there exist an efficient solution if one restricts oneself to an approximation from $\mathcal{F}_\phi = \{f(\cdot) = w^\top \phi(\cdot)\,|\,w \in \mathbb{R}^p\} \subset L^2(\mathcal{X},\xi)$. For linearly independent basis functions $\phi_i \in L^2(\mathcal{X},\xi)$ the projection of any function $f \in L^2(\mathcal{X},\xi)$ into $\mathcal{F}_\phi$ w.r.t. norm $\|\cdot\|_\xi$ can be calculated by the linear *projection operator* $\hat{\Pi}_\xi^\phi : L^2(\mathcal{X},\xi) \to L^2(\mathcal{X},\xi)$,

$$\hat{\Pi}_\xi^\phi[f](x) := \sum_{j=1}^p \overbrace{\sum_{i=1}^p \langle f,\phi_i\rangle_\xi\,(\mathbf{C}^{-1})_{ij}}^{w_j \in \mathbf{R}}\,\phi_j(x)\,, \quad C_{ij} := \langle \phi_i,\phi_j\rangle_\xi\,, \quad \forall x \in \mathcal{X}\,, \quad \forall f \in L^2(\mathcal{X},\xi)\,.$$

Instead of infinitely many alternating applications of $\hat{B}^\pi$ and $\hat{\Pi}_\xi^\phi$, one can directly calculate the fixed point $f^\pi \in \mathcal{F}_\phi$ of the combined operator

$$f^\pi \stackrel{!}{=} \hat{\Pi}_\xi^\phi\big[\hat{B}^\pi[f^\pi]\big] \qquad \Rightarrow \qquad w^\pi = \Big(\underbrace{\langle \phi, \phi - \gamma \hat{P}^\pi[\phi]\rangle_\xi}_{\mathbf{A}^\pi \in \mathbf{R}^{p\times p}}\Big)^\dagger \underbrace{\langle \phi, r^\pi\rangle_\xi}_{b^\pi \in \mathbf{R}^p}\,,$$

11. Every *kernel* $A : \mathcal{X} \times \mathcal{B}(\mathcal{X}) \to [0,\infty)$ induces a linear operator $\hat{A} : L^2(\mathcal{X},\xi) \to L^2(\mathcal{X},\xi), \hat{A}[f](x) := \int A(dy|x)\,f(y), \forall x \in \mathcal{X}, \forall f \in L^2(\mathcal{X},\xi)$, which in this article bears the same name with a hat.
12. A Markov chain is called *ergodic* if it is *aperiodic* and *positive recurrent*: if there is a nonzero probability to break any periodic cycle and if any infinite sequence eventually must come arbitrarily close to every state $x \in \mathcal{X}$. This is a property of the transition kernel rather than the policy. If *one* policy that assigns a nonzero probability to each action yields ergodic Markov chains, then *every* such policy does. Of course this does not hold for deterministic policies.
13. In a straightforward extension of the argument for finite state spaces (Bertsekas, 2007, Chapter 6).

where $\boldsymbol{w}^{\boldsymbol{\pi}} \in \mathbb{R}^p$ denotes the corresponding parameter vector of fixed point $f^{\boldsymbol{\pi}}(x) = (\boldsymbol{w}^{\boldsymbol{\pi}})^\top \boldsymbol{\phi}(x)$, $\forall x \in \mathcal{X}$, $(\mathbf{A}^{\boldsymbol{\pi}})^\dagger$ denotes the *Moore-Penrose pseudo-inverse* of matrix $\mathbf{A}^{\boldsymbol{\pi}}$ and we wrote $(\langle \boldsymbol{\phi}, \boldsymbol{\phi} \rangle_\xi)_{ij} = \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle_\xi$ for convenience.

The stochastic matrices $\mathbf{A}^{\boldsymbol{\pi}}$ and $\boldsymbol{b}^{\boldsymbol{\pi}}$ can be bias-free estimated given a set of transitions $\left\{ \boldsymbol{\phi}(x_t) \xrightarrow{r_t} \boldsymbol{\phi}(x'_t) \right\}_{t=1}^n$ of start states $x_t \sim \xi(\cdot)$, executed actions $a_t \sim \boldsymbol{\pi}(\cdot|x_t)$, corresponding successive states $x'_t \sim P(\cdot|x_t, a_t)$ and received rewards $r_t \sim R(\cdot|x_t, a_t, x'_t)$. The resulting algorithm is known as *least-squares temporal difference learning* (LSTD, Bradtke and Barto, 1996). It can be shown that it converges[14] in $\|\cdot\|_\xi$ norm (Bertsekas, 2007). Note that for this property the samples must be drawn from steady state distribution $\xi$ and policy $\boldsymbol{\pi}$, usually by a long Markov chain executing $\boldsymbol{\pi}$.

Moreover, Tsitsiklis and Van Roy (1997) have proven that in $\|\cdot\|_\xi$ norm the error between true value function $v^{\boldsymbol{\pi}} \in L^2(\mathcal{X}, \xi)$ and approximation $f^{\boldsymbol{\pi}} \in \mathcal{F}_{\boldsymbol{\phi}}$ is bounded[15] by

$$\left\| v^{\boldsymbol{\pi}} - f^{\boldsymbol{\pi}} \right\|_\xi \leq \frac{1}{\sqrt{1-\gamma^2}} \left\| v^{\boldsymbol{\pi}} - \hat{\Pi}_\xi^{\boldsymbol{\phi}} \left[ v^{\boldsymbol{\pi}} \right] \right\|_\xi .$$

In Section 4 we will improve upon this bound significantly for a special case of SFA features. We will also show that for a specific class of tasks the basis functions $\boldsymbol{\phi}_i(\cdot)$ extracted by SFA minimize a mean bound on the right hand side of this equation, in other words minimize the mean approximation error over all considered tasks.

### 2.4 Least-squares Policy Iteration

Estimating the value function does not directly yield a control policy. This problem is tackled by *least-squares policy iteration* (LSPI, Lagoudakis and Parr, 2003), which alternates between *Q-value estimation* (the expectation step) and *policy improvement* (the maximization step). At iteration $i$ with current policy $\boldsymbol{\pi}_i$, the *Q-value function* $Q^{\boldsymbol{\pi}_i} : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is defined as the value of state $x \in \mathcal{X}$ conditioned on the next action $a \in \mathcal{A}$:

$$Q^{\boldsymbol{\pi}_i}(x,a) := r(x,a) + \gamma \int V^{\boldsymbol{\pi}_i}(y) \, P(dy|x,a) = r(x,a) + \gamma \iint Q^{\boldsymbol{\pi}_i}(y,b) \, \boldsymbol{\pi}_i(db|y) \, P(dy|x,a) .$$

Note that *Q-value estimation* is equivalent to *value estimation* in the space of twice integrable functions over the space of state-action pairs $\mathcal{X} \times \mathcal{A}$ endowed with probability measure $\mu(B,A) := \int_B \boldsymbol{\pi}_i(A|x) \, \xi(dx)$, $\forall (B,A) \in \mathcal{B}(\mathcal{X}) \times \mathcal{B}(\mathcal{A})$, that is, $L^2(\mathcal{X} \times \mathcal{A}, \mu)$. The corresponding transition operator $\hat{P}_Q^{\boldsymbol{\pi}_i} : L^2(\mathcal{X} \times \mathcal{A}, \mu) \to L^2(\mathcal{X} \times \mathcal{A}, \mu)$ is

$$\hat{P}_Q^{\boldsymbol{\pi}_i}[f](x,a) := \iint f(y,b) \, \boldsymbol{\pi}_i(db|y) \, P(dy|x,a) , \quad \forall f \in L^2(\mathcal{X} \times \mathcal{A}, \mu) , \quad \forall x \in \mathcal{X} , \quad \forall a \in \mathcal{A} .$$

The greedy policy $\boldsymbol{\pi}_{i+1}$ in the $i$'th *policy improvement* step will for each state $x$ draw one of the actions with the highest Q-value, that is, $a^{\boldsymbol{\pi}_i}(x) \sim \arg\max_{a \in \mathcal{A}} Q^{\boldsymbol{\pi}_i}(x,a)$, and stick with it:

$$\boldsymbol{\pi}_{i+1}(a|x) := \begin{cases} 1 & ,\text{if} \quad a = a^{\boldsymbol{\pi}_i}(x) \\ 0 & ,\text{else} \end{cases} , \quad \forall x \in \mathcal{X} , \quad \forall a \in \mathcal{A} .$$

14. In a straightforward extension of the argument for finite state spaces (Bertsekas, 2007, Chapter 6).
15. Besides this bound in the weighted $L_2$ norm there exists a multitude of bounds in $L_\infty$ and sometimes $L_1$ norm. See Petrik and Zilberstein (2011) for a recent overview.

In finite state-action spaces, this procedure will provably converge to a policy that maximizes the value for all states (Kaelbling et al., 1996).

To cope with continuous state and/or action spaces, LSPI employs the LSTD algorithm to estimate approximated Q-value functions $f^{\pi_i} \in \mathcal{F}_\phi$, where the basis functions $\phi_i \in L^2(\mathcal{X} \times \mathcal{A}, \mu)$ are defined over state-action pairs rather than states alone. In difference to value estimation, any experienced set of transitions $\{(x_t, a_t) \xrightarrow{r_t} x'_t\}_{t=1}^n$ yields the necessary information for Q-value estimation with arbitrary policies $\pi_i$, in other words the LSTD training set

$$\left\{ \phi(x_t, a_t) \xrightarrow{r_t} \int \phi(x'_t, a)\, \pi_i(da|x'_t) \right\}_{t=1}^n .$$

However, convergence guarantees hold *only* when $\mu$ is the steady state distribution of $P_Q^{\pi_i}$, which usually only holds in the first iteration. Although it can thus not be guaranteed, empirically LSPI fails only for large function spaces $\mathcal{F}_\phi$ and $\gamma$ close to 1. In Section 5, Figure 8, we demonstrate this at the example of well and poorly constructed basis functions.

The easiest way to encode $p$ state-action pairs for finite action spaces $\mathcal{A}$ is to use an arbitrary $q := p/|\mathcal{A}|$ dimensional state encoding $\phi : \mathcal{X} \to \mathbb{R}^q$ and to extend it by $\bar{\phi}(x,a) := \phi(x) e_a^\top$, $\forall x \in \mathcal{X}, \forall a \in \mathcal{A}$, where $e_a \in \mathbb{R}^{|\mathcal{A}|}$ is a column vector of length 1 which is 0 everywhere except in one dimension uniquely associated with action $a$. The resulting $q \times |\mathcal{A}|$ matrix $\bar{\phi}(x,a)$ can be treated as set of $p$ state-action basis functions.

### 2.5 Reproducing Kernel Hilbert Spaces

Although $L^2(\mathcal{X}, \xi)$ is a very powerful tool for analysis, it has proven problematic in machine learning (Wahba, 1990; Schölkopf and Smola, 2002). Many algorithms employ instead the well behaving *reproducing kernel Hilbert spaces* $\mathcal{H}_\kappa \subset L^2(\mathcal{Z}, \xi)$ (RKHS, see, e.g., Schölkopf and Smola, 2002). A RKHS is induced by a *positive semi-definite kernel function* $\kappa : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$; the set $\{\kappa(\cdot, x) \mid x \in \mathcal{Z}\}$ is a full (but not orthonormal) basis of $\mathcal{H}_\kappa$. The inner product of two kernel functions in $\mathcal{H}_\kappa$ can be expressed as a kernel function itself. Take the example of the *Gaussian kernel* used in this article:

$$\langle \kappa(\cdot, x), \kappa(\cdot, y) \rangle_{\mathcal{H}_\kappa} \quad = \quad \kappa(x, y) \quad := \quad \exp\left(-\tfrac{1}{2\sigma^2} \|x - y\|_2^2\right), \qquad \forall x, y \in \mathcal{Z}.$$

Due to compactness of $\mathcal{Z}$, all continuous functions $f$ in $L^2(\mathcal{Z}, \xi)$ can be approximated arbitrarily well in $L_\infty$ (supremum) norm by functions from $\mathcal{H}_\kappa$.

Naive implementation of the *kernel trick* with $n$ observed samples $\{z_t\}_{t=1}^n$ induces a computational complexity of $\mathcal{O}(n^3)$ and a memory complexity of $\mathcal{O}(n^2)$. For large $n$ it can thus be necessary to look for approximate solutions in the subspace spanned by some *sparse subset* $\{s_i\}_{i=1}^m \subset \{z_t\}_{t=1}^n$, $m \ll n$, and thus $f(\cdot) = \sum_{i=1}^m \alpha_i \kappa(\cdot, s_i) \in \mathcal{H}_\kappa$, $\alpha \in \mathbb{R}^m$ (*projected process* matrix sparsification, Rasmussen and Williams, 2006). If subset and approximation space are chosen well, the LSTD solution $f^\pi \in \mathcal{F}_\phi$ can be approximated well too:

$$f^\pi \quad \in \quad \underbrace{\mathcal{F}_\phi \quad \subset \quad \mathcal{F}_{\{\kappa(\cdot, s_i)\}_{i=1}^m}}_{\text{approximation space}} \qquad \underset{\text{subset selection}}{\subset} \qquad \underbrace{\mathcal{H}_\kappa \quad \subset \quad L^2(\mathcal{Z}, \xi)}_{\text{all continuous functions}} .$$

However, finding a suitable subset is not trivial (Smola and Schölkopf, 2000; Csató and Opper, 2002). We employ the *matching pursuit for maximization of the affine hull* algorithm (MP-MAH,

Böhmer et al., 2012) to find a uniformly distributed subset. Section 5.5 empirically evaluates the effect of this choice.

A MDP with *discrete state space* $\mathcal{Z} := \{x_i\}_{i=1}^d$ can also be embedded as a RKHS. The kernel $\kappa(x_i, x_j) = \delta_{ij}$ induces the *discrete metric*, where $\delta_{ij}$ is the *Kronecker delta*. In this metric every state is an *open set* and thus Borel sets, integrals, ergodicity and all other concepts in this section can be extended to discrete state spaces. The discrete metric does not allow generalization to neighboring states, though. In this case the "sparse" subsets of the kernel algorithms discussed in Section 3 must contain *all* states, that is, $\{s_i\}_{i=1}^m = \{x_i\}_{i=1}^d$, which restricts the formalism to *finite* or *compact continuous* state spaces.

## 3. Basis Function Construction

Solving an MDP with the methods discussed in Section 2 requires the projection into an approximation space $\mathcal{F}_\phi = \{f(\cdot) = \boldsymbol{w}^\top \boldsymbol{\phi}(\cdot) \,|\, \boldsymbol{w} \in \mathbb{R}^p\} \subset L^2(\mathcal{Z}, \xi)$. The discussed algorithms make it necessary to specify the involved basis functions $\phi_i(\cdot) \in L^2(\mathcal{Z}, \xi), \forall i \in \{1, \dots, p\}$, *before* training, though. As the true value function $v^\pi \in L^2(\mathcal{Z}, \xi)$ is initially unknown, it is not obvious how to pick a basis that will eventually approximate it well. Classical choices (like Fourier bases, Konidaris et al., 2011) are known to approximate *any* continuous function arbitrarily well in the limit case. However, if applied on high-dimensional observations, for example, $\boldsymbol{z} \in \mathbb{R}^d$, the number of required functions $p$ scales exponentially with $d$. It would therefore be highly advantageous to exploit knowledge of task or observation space and *construct* a low dimensional basis.

In this context, recent works have revealed two diametrically opposed concepts (Parr et al., 2008). Expressed in the notation of Section 2, the *Bellman error* of the fixed point solution $f^\pi(\cdot) = (\boldsymbol{w}^\pi)^\top \boldsymbol{\phi}(\cdot) \overset{!}{=} \hat{\Pi}_\xi^\phi [\hat{B}^\pi [f^\pi]] \in \mathcal{F}_\phi$ can be separated into two types of error functions,

$$\hat{B}^\pi [f^\pi] - f^\pi \quad = \quad \underbrace{\left(\hat{I} - \hat{\Pi}_\xi^\phi\right)[r^\pi]}_{\Delta^r \in L^2(\mathcal{Z}, \xi)} \quad + \quad \gamma \sum_{i=1}^p w_i^\pi \underbrace{\left(\hat{I} - \hat{\Pi}_\xi^\phi\right)\left[\hat{P}^\pi [\phi_i]\right]}_{\Delta_i^\phi \in L^2(\mathcal{Z}, \xi)},$$

the *reward error* $\Delta^r \in L^2(\mathcal{Z}, \xi)$ and the *per-feature errors* $\Delta_i^\phi \in L^2(\mathcal{Z}, \xi)$. Correspondingly, there have been two opposing approaches to basis function construction in literature:

1. *Reward-based features* encode the reward function and how it propagates in time. $\Delta^r$ is thus zero everywhere, but $\Delta_p^\phi$ can still induce Bellman errors.

2. *Subspace-invariant features* aim for eigenfunctions of transition operator $\hat{P}^\pi$ to achieve no per-feature errors. $\Delta^r$, however, can still induce Bellman errors.

This article focuses on subspace-invariant feature sets, but reward-based features are introduced for comparison in Section 3.1. As a baseline which encodes distances in $\mathcal{Z}$ but does not attempt subspace invariance, we introduce *principal component analysis* (PCA, Section 3.2). We continue with *proto value functions* (PVF, Section 3.3), which are the current state of the art in subspace-invariant features. *Slow feature analysis* (SFA, Section 3.4) has only recently been proposed to generate basis functions for RL. Section 4 analyzes the properties of both reward-based and subspace-invariant features in detail and Section 5 empirically compares all discussed algorithms in various experiments.

### 3.1 Reward-based Basis Functions

The true value function $v^\pi \in L^2(\mathcal{Z},\xi)$ is defined as $v^\pi(z) = \sum_{t=0}^\infty \gamma^t (\hat{P}^\pi)^t[r](z), \forall z \in \mathcal{Z}$, where $(\hat{P}^\pi)^t$ refers to $t$ consecutive applications of operator $\hat{P}^\pi$ and $(\hat{P}^\pi)^0 := \hat{I}$. This illustrates the intuition of *Krylov bases* (Petrik, 2007):

$$\phi_i^K := (\hat{P}^\pi)^{i-1}[r] \in L^2(\mathcal{Z},\xi), \qquad \Phi_k^K := \{\phi_1^K, \ldots, \phi_k^K\}.$$

These bases are natural for *value iteration*, as the value functions of all iterations can be exactly represented (see also Corollary 10, Page 2087). However, the transition operator must be approximated from observations and the resulting basis $\Phi_k^K$ is not orthonormal. If employed, a projection operator must thus compute an expensive inverse (see Section 2.3). *Bellman error basis functions* (BEBF, Parr et al., 2007) rectify this by defining the $(k+1)$'th feature as the Bellman error of the fixed point solution $f^k \in \mathcal{F}_{\Phi_k^B}$ with $k$ features:

$$\phi_{k+1}^B := \hat{B}^\pi[f^k] - f^k \in L^2(\mathcal{Z},\xi), \quad f^k \overset{!}{=} \hat{\Pi}_\xi^{\Phi_k^B}\left[\hat{B}^\pi[f^k]\right] \in \mathcal{F}_{\Phi_k^B}, \quad \Phi_k^B := \{\phi_1^B, \ldots, \phi_k^B\}.$$

BEBF are orthogonal, that is, $\langle \phi_i^B, \phi_j^B \rangle_\xi = \varepsilon \delta_{ij}, \varepsilon > 0$, and scaling $\varepsilon$ to 1 yields an orthonormal basis. Parr et al. (2007) have shown that Krylov bases and BEBF span the same approximation space $\mathcal{F}_{\Phi_k^K} = \mathcal{F}_{\Phi_k^B}$. Both approaches require many features if $\gamma \to 1$.

Mahadevan and Liu (2010) have extended BEBF to *Bellman average reward bases* (BARB) by including the *average reward* $\rho$ as the first feature. This is motivated by *Drazin bases* and has been reported to reduce the number of required features for large $\gamma$. Recently, Sun et al. (2011) have pointed out that given some basis $\Phi_k$, the *best* $k+1$'th basis function is always the fixed point solution with the current Bellman error, that is, the next BEBF $\phi_{k+1}^B$, as reward. Adding the resulting *Value function of the Bellman error* (V-BEBF) to the current basis can represent the true value exactly. However, the approach has to be combined with some feature selection strategy, as finding the V-BEBF fixed point is just as hard.

All above algorithms are exclusively defined on discrete MDPs. Although an extension to general RKHSs seems possible, it is not the focus of this article. However, to give readers a comparison of available methods we will evaluate orthogonalized Krylov bases (which are equivalent to BEBF) on discrete Benchmark tasks in Sections 5.2 and 5.3.

### 3.2 Principal Component Analysis

To provide a baseline for comparison, we introduce *principal component analysis* (PCA, Pearson, 1901). As PCA does not take any transitions into account, the extracted features must therefore encode Euclidean distances in $\mathcal{Z}$. PCA aims to find subspaces of maximal variance, which are spanned by the eigenvectors to the *p largest* eigenvalues of the data covariance matrix. One interpretation of PCA features $\phi : \mathcal{Z} \to \mathbb{R}^p$ is an optimal encoding of the centered data $\{z_t\}_{t=1}^n \subset \mathcal{Z} \subset \mathbb{R}^d$ w.r.t. linear *least-squares* reconstruction, that is the optimization problem

$$\inf_{\phi \in (\mathcal{F}_{\text{lin}})^p} \underbrace{\inf_{f \in (\mathcal{F}_\phi)^d} \tilde{\mathbb{E}}_t\left[\left\|z_t - f(z_t)\right\|_2^2\right]}_{\text{least-squares reconstruction in } \mathcal{F}_\phi},$$

where $\tilde{\mathbb{E}}_t[\cdot]$ is the empirical expectation operator w.r.t. all indices $t$ and $\mathcal{F}_{\text{lin}}$ the set of linear functions in $\mathbb{R}^d$.

In the interest of a more efficient encoding one can extend the function set $\mathcal{F}_{\text{lin}}$. A popular example are *reproducing kernel Hilbert spaces*, introduced in Section 2.5. The resulting algorithm is called *kernel PCA* (Schölkopf et al., 1998) and performs an eigenvalue decomposition of the centered *kernel matrix* $K_{ij} = \kappa(z_i, z_j)$. The eigenvectors $v^i \in \mathbb{R}^n$ to the $p$ largest eigenvalues are the coefficients to the feature maps:

$$\phi_i(z) := \sum_{t=1}^{n} v_t^i \, \kappa(z, z_t), \quad \forall z \in \mathcal{Z}.$$

The classical algorithm (Schölkopf et al., 1998) is severely limited by a *computational complexity* of $\mathcal{O}(n^3)$ and a *memory complexity* of $\mathcal{O}(n^2)$. It can thus be necessary to approximate the solution by using a *sparse kernel matrix* of a subset of the data (*projected process*, Rasmussen and Williams, 2006), that is, $K_{it} = \kappa(s_i, z_t)$, with $\{s_i\}_{i=1}^m \subset \{z_t\}_{t=1}^n$, $m \ll n$. The eigenvectors $v^i \in \mathbb{R}^m$ of $\frac{1}{n}\mathbf{K}\mathbf{K}^\top$ determine the coefficients of the *sparse kernel PCA* features. If a large enough subset is distributed uniformly in $\mathcal{Z}$, the approximation is usually very good.

### 3.3 Proto-value Functions

In finite state spaces $\mathcal{Z}$, $|\mathcal{Z}| < \infty$, proto-value functions (PVF, Mahadevan and Maggioni, 2007) are motivated by *diffusion maps* on graphs (Coifman et al., 2005). For this purpose a *connection graph* is constructed out of a Markov chain $\{z_t\}_{t=1}^n \subset \mathcal{Z}$: for the first observed transition from state $x$ to $y$, the corresponding entry of connection matrix $\mathbf{W}$ is set $W_{xy} := 1$. All entries of non-observed transitions are set to zero. As diffusion maps require undirected graphs, this matrix must be symmetrized by setting $\mathbf{W} \leftarrow \frac{1}{2}(\mathbf{W} + \mathbf{W}^\top)$. PVF are the eigenvectors to the $p$ smallest eigenvalues of the *normalized graph Laplacian* $\mathbf{L} := \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{W})\,\mathbf{D}^{-1/2}$, where $D_{xy} = \delta_{xy} \sum_{z=1}^{|\mathcal{Z}|} W_{xz}, \forall x, y \in \mathcal{Z}$, and $\delta_{xy}$ is the Kronecker delta. Section 4.1 shows that this approach, also known as *spectral encoding* (Belkin and Niyogi, 2003), yields approximation spaces in which Euclidean distances are equivalent to diffusion distances on the connection graph. Note, however, that these are not exactly the diffusion distances of the transition kernel, as the transition *possibility* rather than *probability* is encoded in matrix $\mathbf{W}$. Section 4.2 discusses this difference.

For infinite observation spaces $\mathcal{Z}$ PVF are also defined by connection graphs. However, in difference to the finite case, the construction of this graph is not straightforward. Mahadevan and Maggioni (2007) proposed a symmetrized *k-nearest neighbors graph* $\mathbf{W}$ out of a random[16] set $\{s_j\}_{j=1}^m \subset \{z_t\}_{t=1}^n$, $m \ll n$. Each node $s_i$ is only connected with the $k$ nearest nodes $\{s'_j\}_{j=1}^k \subset \{s_j\}_{j=1}^m$ (w.r.t. the Euclidean norm in $\mathcal{Z}$), with weights determined by a Gaussian kernel $\kappa(\cdot,\cdot)$ with width-parameter $\sigma$,

$$W_{ij} := \kappa(s_i, s_j) = \exp\left(-\tfrac{1}{2\sigma^2}\|s_i - s_j\|_2^2\right).$$

After symmetrization the PVF $\hat{\phi}_i$ at the nodes $s_j$ are calculated. A Nyström extension approximates the PVF for all samples $z$ by calculating the mean over the weighted PVF of the $k$ nodes $\{s'_j\}_{j=1}^k \subset \{s_j\}_{j=1}^m$ closest to $z$,

$$\phi_i(z) := \sum_{j=1}^{k} \frac{\kappa(z, s'_j)}{\sum_{l=1}^{k} \kappa(z, s'_l)} \, \hat{\phi}_i(s'_j), \quad \forall z \in \mathcal{Z}.$$

16. Ideally the nodes are uniformly drawn w.r.t. the true diffusion metric, in other words uniformly in $\mathcal{X}$. If nodes are drawn randomly or uniformly in $\mathcal{Z}$, this difference can lead to a significant deviation in the number of transitions between nodes and the resulting diffusion distances thus deviate as well.

Note that these features are no longer based on the *transitions* of the observed Markov chain, but on Euclidean distances in $\mathcal{Z}$.

### 3.4 Slow Feature Analysis

The unsupervised learning method *slow feature analysis* (SFA, Wiskott and Sejnowski, 2002) aims for a set of mappings $\boldsymbol{\phi} : \mathcal{Z} \to \mathbb{R}^p$ such that the values $\phi_i(\boldsymbol{z}_t)$ change slowly over an observed Markov chain $\{\boldsymbol{z}_t\}_{t=1}^n \subset \mathcal{Z}$. The objective (called *slowness* $\mathcal{S}$) is defined as the *expectation of the squared discrete temporal derivative*:

$$\inf_{\boldsymbol{\phi} \in (\mathcal{F})^p} \quad \sum_{i=1}^p \mathcal{S}(\phi_i) \quad := \quad \sum_{i=1}^p \tilde{\mathbb{E}}_t[\dot{\phi}_i^2(\boldsymbol{z}_t)] \quad \text{(slowness)}\,.$$

To ensure each slow feature encodes unique information and can be calculated in an iterative fashion, the following constraints must hold $\forall i \in \{1, \ldots, p\}$ :

$$\begin{aligned} \tilde{\mathbb{E}}_t[\phi_i(\boldsymbol{z}_t)] &= 0 && \text{(zero mean)}, \\ \tilde{\mathbb{E}}_t[\phi_i^2(\boldsymbol{z}_t)] &= 1 && \text{(unit variance)}, \\ \forall j \neq i : \tilde{\mathbb{E}}_t[\phi_i(\boldsymbol{z}_t)\phi_j(\boldsymbol{z}_t)] &= 0 && \text{(decorrelation)}, \\ \forall j > i : \mathcal{S}(\phi_i) &\leq \mathcal{S}(\phi_j) && \text{(order)}\,. \end{aligned}$$

The principle of slowness has been used for a long time in the context of neural networks (Földiák, 1991). Kompella et al. (2012) have proposed an incremental online SFA algorithm. Recently several groups have attempted to use SFA on a random walk of observations to generate basis functions for RL (Legenstein et al., 2010; Luciw and Schmidhuber, 2012).

Although formulated as a linear algorithm, SFA was originally intended to be applied on the space of polynomials like quadratic (Wiskott and Sejnowski, 2002) or cubic (Berkes and Wiskott, 2005). The polynomial expansion of potentially high dimensional data, however, spans an impractically large space of coefficients. Hierarchical application of quadratic SFA has been proposed to solve this problem (Wiskott and Sejnowski, 2002; Legenstein et al., 2010). Although proven to work in complex tasks (Franzius et al., 2007), this approach involves a multitude of hyper-parameters and no easy way to counteract inevitable over-fitting is known so far.

An alternative to polynomial expansions are *sparse kernel methods* (see Section 2.5). We summarize in the following the *regularized sparse kernel SFA* (RSK-SFA, Böhmer et al., 2012) which we have used in our experiments. For a given sparse subset $\{\boldsymbol{s}_i\}_{i=1}^m \subset \{\boldsymbol{z}_t\}_{t=1}^n$, the algorithm determines the mapping $\boldsymbol{\phi} : \mathcal{Z} \to \mathbb{R}^p$ in 3 steps:

i Fulfilling the zero mean constraint directly on sparse kernel matrix $K_{it} := \boldsymbol{\kappa}(\boldsymbol{s}_i, \boldsymbol{z}_t)$, that is, $\mathbf{K}' := (\mathbf{I} - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^\top)\mathbf{K}(\mathbf{I} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top)$, where $\mathbf{1}_k \in \mathbb{R}^k$ is a column vector of ones.

ii Fulfilling unit variance and decorrelation constraints by performing an eigenvalue decomposition $\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top := \frac{1}{n}\mathbf{K}'\mathbf{K}'^\top$ and projecting $\mathbf{K}'' := \boldsymbol{\Lambda}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{K}'$.

iii Minimize the objective by another eigenvalue decomposition $\mathbf{R}\boldsymbol{\Sigma}\mathbf{R}^\top := \frac{1}{n-1}\dot{\mathbf{K}}''\dot{\mathbf{K}}''^\top$, where $\dot{K}''_{it} := K''_{it+1} - K''_{it}$. Grouping the kernel functions of the sparse subset into one multivariate function $\boldsymbol{k} : \mathcal{Z} \to \mathbb{R}^m$ with $k_i(\boldsymbol{z}) := \boldsymbol{\kappa}(\boldsymbol{z}, \boldsymbol{s}_i)$, $\forall \boldsymbol{z} \in \mathcal{Z}$, the solution is given by

$$\boldsymbol{\phi}(\boldsymbol{z}) = \mathbf{A}^\top \boldsymbol{k}(\boldsymbol{z}) - \boldsymbol{c}, \quad \forall \boldsymbol{z} \in \mathcal{Z}$$
$$\text{with } \mathbf{A} := (\mathbf{I} - \mathbf{1}_m\mathbf{1}_m^\top)\mathbf{U}\boldsymbol{\Lambda}^{-\frac{1}{2}}\mathbf{R}, \text{ and } \boldsymbol{c} := \tfrac{1}{n}\mathbf{A}\mathbf{K}\mathbf{1}_n\,.$$

Böhmer et al. (2012) have demonstrated the numerical instability of this algorithm in face of insufficient sparseness and introduced a *regularization term* $\|\phi_i\|_{\mathcal{H}_\kappa}$ to the objective to stabilize the solution. In our experiments we did not face this problem, and regularization is thus omitted here.

### 3.5 Relative Velocities and Modified SFA

In the limit of an infinite Markov chain in $\mathcal{Z}$ and some mild assumptions[17] on the transition kernel in $\mathcal{X}$, the slowest possible mappings can be calculated analytically (Wiskott, 2003; Franzius et al., 2007). As the discrete temporal derivative is specified by the transition kernel in $\mathcal{X}$, the analytical solutions have domain $\mathcal{X}$ as well. Note, however, that the same transition kernel yields the same feature maps in $\mathcal{X}$, independent[18] of the actual observation space $\mathcal{Z}$. Section 4.1 demonstrates that these solutions are endowed with the diffusion metric of the symmetrized transition kernel, not unlike PVF in finite state spaces. Sprekeler (2011) has recently shown that in this case SFA solutions are (almost) equivalent to PVF. Note also that SFA encodes the actual transition *probabilities*, which requires more samples to converge than the transition *possibilities* encoded by PVF.

The analytically derived SFA features of Franzius et al. (2007) are of particular interest to the visual navigation experiment (Section 5.4 and Figure 2, Page 2071), as they assume the same setup. The solution is a *Fourier basis* on domain $\mathcal{X} := [0, L_x] \times [0, L_y] \times [0, 2\pi)$,

$$\phi_{\iota(i,j,l)}(x,y,\theta) = \begin{cases} \sqrt[3]{2}\cos(\frac{i\pi}{L_x}x)\cos(\frac{j\pi}{L_y}y)\sin(\frac{l+1}{2}\theta), & l \text{ odd} \\ \sqrt[3]{2}\cos(\frac{i\pi}{L_x}x)\cos(\frac{j\pi}{L_y}y)\cos(\frac{l}{2}\theta), & l \text{ even} \end{cases}, \quad \forall (x,y,\theta) \in \mathcal{X},$$

where $\iota : (\mathbb{N} \times \mathbb{N} \times \mathbb{N} \setminus \{(0,0,0)\}) \to \mathbb{N}^+$ is an index function, which depends on the *relative velocities* in two spatial dimensions $x$ and $y$, and the robot's orientation $\theta$. It can occur that SFA features have the same slowness, in which case the solution is no longer unique. For example, if $\phi_{\iota(1,0,0)}$ and $\phi_{\iota(0,1,0)}$ have the same slowness, then $\mathcal{S}(\phi_{\iota(1,0,0)}) = \mathcal{S}(\phi_{\iota(0,1,0)}) = \mathcal{S}(a\phi_{\iota(1,0,0)} + b\phi_{\iota(0,1,0)})$ holds as long as $a^2 + b^2 = 1$. This corresponds to an arbitrary rotation in the subspace of equally slow features. However, if we are interested in the space spanned by all features *up to a certain slowness*, every rotated solution spans the same approximation space $\mathcal{F}_\phi$.

The order $\iota(\cdot,\cdot,\cdot)$ of the analytical SFA features derived by Franzius et al. (2007, see above) depend strongly on the *relative velocities* in the state dimensions. For example, crossing the room in our experiment in Section 5.4 requires 10 movements, during which feature $\phi_{\iota(1,0,0)}$ will run through half a cosine wave. In as little as 4 rotations, on the other hand, feature $\phi_{\iota(0,0,1)}$ registers the same amount of change. Sampled evenly by a random policy, the first SFA features will therefore *not* encode the robot's orientation, which can critically impair the value representation in low dimensional approximation spaces. This article proposes a simple modification to the RSK-SFA algorithm to adjust the relative velocities by means of *importance sampling*.

Let $\{(z_t, a_t)\}_{t=0}^n$ denote a training sequence sampled by policy $\pi$ with a steady state distribution $\xi$, which induces the joint distribution $\mu(B,A) = \int_B \pi(A|z)\,\xi(dz), \forall B \in \mathcal{B}(\mathcal{Z}), \forall A \in \mathcal{B}(\mathcal{A})$. To switch to another policy $\tau$ and state distribution $\zeta$, that is, the joint distribution $\eta(B,A) = \int_B \tau(A|z)\,\zeta(dz), \forall B \in \mathcal{B}(\mathcal{Z}), \forall A \in \mathcal{B}(\mathcal{A})$, one can weight each transition with the *Radon-Nikodym*

17. In this case decorrelated Brownian motion in a multivariate state space $\mathcal{X}$ with independent boundary conditions for each dimension. Examples are rectangles, cubes, tori or spheres of real coordinates.

18. In line with SFA literature, this article does not discuss *partial observability* of the state. In other words, we assume there exist an *unknown* one-to-one mapping of states $x \in \mathcal{X}$ to observations $z \in \mathcal{Z}$.

*derivative* $\frac{d\eta}{d\mu}$. This yields the modified SFA optimization problem

$$\begin{aligned} \inf_{\phi\in(\mathcal{F})^p} \quad & \sum_{i=1}^{p} \hat{\mathbb{S}}(\phi_i,\eta) \quad := \quad \sum_{i=1}^{p} \tilde{\mathbb{E}}_t\left[\tfrac{d\eta}{d\mu}(z_t,a_t)\,\dot{\phi}_i^2(z_t)\right] \\ \text{s.t.} \quad & \begin{aligned} \tilde{\mathbb{E}}_t\left[\tfrac{d\eta}{d\mu}(z_t,a_t)\,\phi_i(z_t)\right] &= 0 \\ \tilde{\mathbb{E}}_t\left[\tfrac{d\eta}{d\mu}(z_t,a_t)\,\phi_i(z_t)\,\phi_j(z_t)\right] &= \delta_{ij} \end{aligned} \quad , \quad \forall i,j\in\{1,\ldots,p\}\,. \end{aligned}$$

However, there is no indication which distribution $\zeta$ and policy $\tau$ ensure a balanced encoding.

We propose here a simple heuristic for situations in which the actions affect only mutually independent subspaces of $\mathcal{X}$. In our robot navigation experiment, for example, rotations do not influence the robot's spatial position nor do the movements influence it's orientation. As optimal SFA features in the spatial subspace are significantly slower (see above), the first features will encode this subspace exclusively. This can be counteracted by setting $\zeta := \xi$ and defining $\frac{d\tau}{d\pi}(z,a) := \vartheta(a), \forall z\in\mathcal{Z}, \forall a\in\mathcal{A}$, where $\vartheta : \mathcal{A}\to\mathbb{R}^+$ weights each action independent of the current state. In practice, weights $\vartheta(a)$ need to be adjusted by hand for each action $a\in\mathcal{A}$: the *higher* $\vartheta(a)$, the *weaker* the influence of the corresponding subspace of $\mathcal{X}$ onto the first features. Only the last step (iii) of RSK-SFA has to be modified by redefining $\dot{K}''_{it} \leftarrow \vartheta^{\frac{1}{2}}(a_t)\,\dot{K}''_{it}$. Figure 9, Page 2101, demonstrates the effect of this modification.

## 4. Theoretical Analysis

This section analyzes the theoretical properties of *reward-based* and *subspace-invariant* features w.r.t. value function approximation. The employed formalism is introduced in Section 2. If not stated otherwise, the features are assumed to be optimized over $L^2(\mathcal{Z},\xi)$ and based on an infinite ergodic Markov chain. Proofs to all given lemmas and theorems can be found in Appendix A.

At the heart of function approximation lies the concept of *similarity*. Similar states will have similar function values. Usually this similarity is given by a metric on the observation space. Deviations of the function output from this metric must be compensated by the optimization algorithm. However, value function approximation allows for *explicit* specification of the required similarity. The definition of the value assigns similar function output to states with (i) similar immediate rewards and (ii) similar futures. As discussed in Section 3, *reward-based features* focus on encoding (i), whereas *subspace-invariant features* focus on (ii). Section 4.1 analyzes how SFA encodes similar futures as *diffusion distances* and shows some restrictions imposed onto the class of subspace-invariant features. The ramifications of these restrictions onto value function approximation are discussed in Section 4.2.

This article also presents a second, novel perspective onto value function approximation. The MDP one will face is usually not known before learning and the construction of a suitable basis is very expensive. Instead of approximating a particular MDP at hand, one could focus on a complete set $\mathcal{M}$ of *anticipated* MDPs. An *optimal* approximation space should be able to approximate any MDP $m\in\mathcal{M}$ if encountered. In difference to the classical analysis put forward by Petrik (2007), Parr et al. (2007) and Mahadevan and Maggioni (2007), this point of view puts emphasis on the reuse of prior experience, as investigated in *transfer learning* (Taylor and Stone, 2009; Ferguson and Mahadevan, 2006; Ferrante et al., 2008). Section 4.3 defines a criterion of *optimal features* for some anticipated set $\mathcal{M}$. Under some assumptions on $\mathcal{M}$, we prove that SFA optimizes a bound on

this criterion and argue that there can be no better bound based on a single Markov chain. Section 4.4 provides a summarizing conclusion and further implications can be found in Section 6.

### 4.1 Diffusion Metric

Values of observed states $x, y \in \mathcal{Z}$ depend less on their Euclidean distance in $\mathcal{Z}$ than on common *future states*. PVF are thus based on *diffusion distances* $d_t(x,y)$ of a graph representing the symmetrized *transition possibilities* $T_{xy} := W_{xy}/(\sum_{z\in\mathcal{Z}} W_{xz})$ between discrete states (see Section 3.3 or Mahadevan and Maggioni, 2007):

$$d_t^2(x,y) = \sum_{z\in\mathcal{Z}} \xi_z \left( (\mathbf{T}^t)_{xz} - (\mathbf{T}^t)_{yz} \right)^2 ,$$

where $\xi \in \mathbb{R}^{|\mathcal{Z}|}$ are arbitrary[19] non-negative weights and $\mathbf{T}^t$ denotes the $t$'th power of matrix $\mathbf{T}$. These diffusion distances are equal to Euclidean distances in a space spanned by the eigenvectors $\phi_i \in \mathbb{R}^{|\mathcal{Z}|}$ and eigenvalues $\lambda_i \in \mathbb{R}$ of connectivity matrix $\mathbf{T}$ (e.g., for general similarity matrices see Coifman et al., 2005):

$$d_t(x,y) = \left\| \psi^t(x) - \psi^t(y) \right\|_2 , \qquad \psi_i^t(x) := \lambda_i^t \phi_{ix} , \qquad \forall t \in \mathbb{N} .$$

An extension to potentially continuous observation (or state) spaces $\mathcal{Z}$ with ergodic transition kernels $P^\pi : \mathcal{Z} \times \mathcal{B}(\mathcal{Z}) \to [0,1]$ is not trivial. Mean-squared differences between distributions are not directly possible, but one can calculate the difference between *Radon-Nikodym derivatives*.[20] Due to ergodicity the derivative always exists for finite sets $\mathcal{Z}$, but for continuous $\mathcal{Z}$ one must exclude transition kernels that are not absolutely continuous.[21]

**Assumption 1** *If $\mathcal{Z}$ is continuous, the transition kernel knows no finite set of future states.*

$$P(B|z,a) \quad = \quad 0 , \qquad \forall B \in \left\{ B \in \mathcal{B}(\mathcal{Z}) \,\middle|\, |B| < \infty \right\} , \quad \forall z \in \mathcal{Z} , \quad \forall a \in \mathcal{A} .$$

This can always be fulfilled by adding a small amount of *continuous noise* (e.g., Gaussian) to each transition. Let in the following $(P^\pi)^t(\cdot,x) : \mathcal{B}(\mathcal{Z}) \to [0,1]$ denote the state distribution after $t$ transitions, starting at state $x \in \mathcal{Z}$. Note that under Assumption 1 the Radon-Nikodym derivative w.r.t. steady state distribution $\xi$ is[22] $\frac{d(P^\pi)^t(\cdot|x)}{d\xi} \in L^2(\mathcal{Z},\xi), \forall t \in \mathbb{N} \setminus \{0\}$.

**Definition 1** *The diffusion distance $d_t : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}^+$ based on ergodic transition kernel $P^\pi$ with steady state distribution $\xi$ is defined as*

$$d_t(x,y) \; := \; \left\| \mu_x^t - \mu_y^t \right\|_\xi , \qquad \mu_x^t \; := \; \tfrac{d(P^\pi)^t(\cdot|x)}{d\xi} \in L^2(\mathcal{Z},\xi) , \qquad \forall x,y \in \mathcal{Z} , \quad \forall t \in \mathbb{N} \setminus \{0\} .$$

19. In our context these weights are the equivalent to the steady state distribution and thus named the same.
20. If Radon-Nikodym derivative $\frac{d\zeta}{d\xi}$ exists then $\int \xi(dz) \frac{d\zeta}{d\xi}(z) f(z) \; = \; \int \zeta(dz) f(z) , \; \forall f \in L^2(\mathcal{Z},\xi)$.
21. The Radon-Nikodym derivative $\frac{d\zeta}{d\xi}$ exists if distribution $\zeta$ is *absolutely continuous* w.r.t. steady state distribution $\xi$, that is if $\xi(B) = 0 \; \Rightarrow \; \zeta(B) = 0, \; \forall B \in \mathcal{B}(\mathcal{Z})$. If there exists a finite Borel set $B \in \mathcal{B}(\mathcal{Z})$ with $\zeta(B) > 0$, however, the derivative must not exist as $\xi(B) = 0$ can hold for ergodic Markov chains.
22. Assumption 1 guarantees the Radon-Nikodym derivative exists in the space of integrable functions $L^1(\mathcal{Z},\xi)$, but by compactness of $\mathcal{Z}$ the derivative is also in $L^2(\mathcal{Z},\xi) \subset L^1(\mathcal{Z},\xi)$ (Reed and Simon, 1980).

The projection methods discussed in Section 2 are based on *Euclidean distances* in the approximation space $\mathcal{F}_\phi$. These spaces are invariant to scaling of the basis functions. Given a particular scaling, however, diffusion distances can equal Euclidean distances in $\mathcal{F}_\phi$. In this case we say that the basis functions *encode* this distance.

**Definition 2** *Basis functions $\phi_i \in L^2(\mathcal{Z},\xi), i \in \{1,\dots,p\}$, are said to "encode" a distance function $d : \mathcal{Z}\times\mathcal{Z} \to \mathbb{R}^+$ if there exists a scaling vector $\varrho \in (\mathbb{R}^+)^p$ such that*

$$d(x,y) \quad = \quad \sqrt{\sum_{i=1}^{p} \varrho_i \Big(\phi_i(x) - \phi_i(y)\Big)^2} \quad = \quad \Big\|\phi(x) - \phi(y)\Big\|_\varrho , \qquad \forall x,y \in \mathcal{Z}\,.$$

If the analogy between value function generalization and diffusion distances is correct, one should aim for a set of basis functions that at least approximates an encoding of diffusion distances $d_t(\cdot,\cdot)$, if possible for all forecast parameters $t \in \mathbb{N}\setminus\{0\}$ at once.

**Lemma 3** *Let $\xi$ denote the steady state distribution of ergodic transition kernel $P^\pi$, which has a self-adjoint transition operator $\hat{P}^\pi = (\hat{P}^\pi)^* : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$. The corresponding diffusion distance equals the Euclidean distance in the space spanned by $\psi_i^t(\cdot) := \lambda_i^t \phi_i(\cdot), \forall i \in \mathbb{N}$, where $\lambda_i \in \mathbb{R}$ and $\phi_i \in L^2(\mathcal{Z},\xi)$ are the eigenvalues and eigenfunctions of $\hat{P}^\pi$, that is*

$$d_t(x,y) = \|\psi^t(x) - \psi^t(y)\|_2\,, \quad \forall x,y \in \mathcal{Z}, \forall t \in \mathbb{N}\setminus\{0\}\,.$$

**Proof** *see Appendix A, Page 2108.* ■

Note that the full set of eigenfunctions $\phi_i$ encodes *all* diffusion distances $d_t(\cdot,\cdot), \forall t \in \mathbb{N}\setminus\{0\}$. Lemma 3 shows that the above relationship between diffusion and Euclidean distances in the eigenspace of the transition operator $\hat{P}^\pi : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$ also holds, but only if this operator is *self-adjoint*.[23] This does not hold for most transition operators, however. Their eigenfunctions do not have to be orthogonal or even be real-valued functions, analogous to complex eigenvectors of asymmetric matrices. Using these eigenfunctions, on the other hand, is the declared intent of *subspace-invariant features* (see Section 3). Constructing real-valued basis functions with zero *per-feature error* thus does not seem generally possible.

In this light one can interpret the *symmetrized transition possibilities* encoded by PVF as a self-adjoint approximation of the *transition probabilities* of $P^\pi$. This raises the question whether better approximations exist.

**Lemma 4** *Let $P^\pi$ be an ergodic transition kernel in $\mathcal{Z}$ with steady state distribution $\xi$. The kernel induced by adjoint transition operator $(\hat{P}^\pi)^*$ in $L^2(\mathcal{Z},\xi)$ is $\xi$-almost-everywhere an ergodic transition kernel with steady state distribution $\xi$.*
**Proof** *see Appendix A, Page 2109.* ■

To obtain a self-adjoint transition kernel, Lemma 4 shows that the kernel of the adjoint operator $(\hat{P}^\pi)^*$ is a transition kernel as well. Intuitively, when $\hat{P}^\pi$ causes all water to flow downhill, $(\hat{P}^\pi)^*$ would cause it to flow the same way uphill. Note the difference to an *inverse* transition kernel, which could find new ways for the water to flow uphill. Although this changes the transition dynamics,

23. Each linear operator $\hat{A} : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$ has a unique *adjoint operator* $\hat{A}^*$ for which holds: $\langle f, \hat{A}[g]\rangle_\xi = \langle \hat{A}^*[f], g\rangle_\xi, \forall f,g \in L^2(\mathcal{Z},\xi)$. The operator is called *self-adjoint*, if $\hat{A} = \hat{A}^*$.

one can construct a *symmetrized transition operator* $\hat{P}_s^{\pi} := \frac{1}{2}\hat{P}^{\pi} + \frac{1}{2}(\hat{P}^{\pi})^*$ as a self-adjoint approximation of $\hat{P}^{\pi}$. Estimating $\hat{P}_s^{\pi}$ may take more samples than the connection graph $\mathbf{T}$ constructed by PVF, but it stands to reason that $\hat{P}_s^{\pi}$ is a better approximation to $\hat{P}^{\pi}$. This intuition is put to the test in Section 5.1. We find indeed that SFA features have on average smaller *per-feature errors* than PVF. For purely random transition kernels the advantage of SFA is minuscule, but the when $\hat{P}^{\pi}$ resembles a self-adjoint operator the difference is striking (see Figure 3 on Page 2092). The goal of encoding diffusion distances based on $P^{\pi}$ appears thus best served by the eigenfunctions of the symmetrized transition operator $\hat{P}_s^{\pi}$. Lemma 5 shows[24] that in the limit of an infinite training sequence, SFA extracts these eigenfunctions in the order of their largest eigenvalues:

**Lemma 5** *In the limit of an infinite ergodic Markov chain drawn by transition kernel $P^{\pi}$ in $\mathcal{Z}$ with steady state distribution $\xi$ holds $\mathcal{S}(f) = 2\langle f, (\hat{I} - \hat{P}^{\pi})[f]\rangle_{\xi}, \forall f \in L^2(\mathcal{Z}, \xi)$.*
**Proof** *see Appendix A, Page 2109.* ■

Note that the first, constant eigenfunction of $\hat{P}_s^{\pi}$ is not extracted, but has no influence on the encoded distance. Encoding any diffusion distance $d_t(\cdot,\cdot)$ would therefore need a potentially infinite number of SFA features. As the influence of each feature shrinks exponentially with the forecast parameter $t$, however, the encoding can be approximated well by the first $p$ SFA features. Except for $t = 0$, this approximation is optimal in the least-squares sense. Note also that for fixed $p$ the approximation quality *increases* with $t$. Predictions based on SFA features will therefore be more accurate in the long term than in the short term.

**Theorem 6** *SFA features $\{\phi_i\}_{i=1}^{\infty}$ simultaneously encode all diffusion distances $d_t(\cdot,\cdot), \forall t \in \mathbb{N} \setminus \{0\}$, based on the symmetrized transition kernel $P_s^{\pi} = \frac{1}{2}P^{\pi} + \frac{1}{2}(P^{\pi})^*$. The first $p$ SFA features are an optimal $p$-dimensional least-squares approximation to this encoding.*
**Proof** *The theorem follows directly from Definition 2 and Lemmas 3, 4, 5 and 15.* ■

A similar proposition can be made for PVF features and diffusion distances based on *transition possibilities*. The connection to reward-based features (Section 3.1) is less obvious. Concentrating naturally on immediate and short term reward, these basis functions depend on the *reward function* at hand. It is, however, possible to show the encoding of diffusion distances *on average*, given the reward function is drawn from a *white noise functional*[25] $\rho$.

**Theorem 7** *On average over all reward functions $r^{\pi} : \mathcal{Z} \to \mathbb{R}$ drawn from a white noise functional $\rho$, the squared norm of a Krylov basis $\{\phi_i^K\}_{i=1}^{p}$ from an ergodic transition kernel $P^{\pi}$ encodes squared diffusion distances based on $\hat{P}^{\pi}$ up to horizon $p-1$, that is*

$$d_t^2(x,y) \;=\; \mathbb{E}\left[\left\|\phi^K(x) - \phi^K(y)\right\|_{\varrho}^2 \,\middle|\, r^{\pi} \sim \rho\right], \quad \forall x,y \in \mathcal{Z},\ \exists \varrho \in (\mathbb{R}^+)^p,\ \forall t \in \{1,\ldots,p-1\}.$$

**Proof** *see Appendix A, Page 2110.* ■

Although Krylov bases are different for each reward function $r^{\pi} \in L^2(\mathcal{Z}, \xi)$ and the employed squared distances diverge slightly from Definition 1, Theorem 7 implies that on average they encode

24. Lemma 5 shows that the SFA optimization problem is equivalent to $\inf_{\phi} \langle \phi, (\hat{I} - \hat{P}^{\pi})[\phi]\rangle_{\xi} \equiv \sup_{\phi} \langle \phi, \hat{P}^{\pi}[\phi]\rangle_{\xi} = \sup_{\phi} \langle \phi, \hat{P}_s^{\pi}[\phi]\rangle_{\xi}$, due to the symmetry of the inner product.
25. A white noise functional is the Hilbert space equivalent to a Gaussian normal distribution (Holden et al., 2010). In our context it suffices to say that $\mathbb{E}[\langle f, r^{\pi}\rangle_{\xi}^2 \,|\, r^{\pi} \sim \rho] = \langle f, f\rangle_{\xi}, \forall f \in L^2(\mathcal{Z}, \xi)$.

diffusion distances up to time horizon $p-1$. The same results hold for BEBF (Petrik, 2007) and with minor modifications for BARB bases (Mahadevan and Liu, 2010).

In conclusion, reward-based features encode diffusion distances exactly up to some time horizon, whereas SFA and PVF approximate an encoding for *all* possible distances. So far the only connection to value function approximation is the intuition of a generalizing metric. However, in the next subsection we show striking parallels between diffusion distances and approximation errors.

### 4.2 Value Function Approximation

The analysis in Section 4.1 revealed a critical problem for the construction of *subspace-invariant features* (Parr et al., 2008): eigenfunctions of the transition operator $\hat{P}^{\pi}$ are not necessarily orthogonal and real-valued. Constructing a real-valued, orthonormal basis of subspace-invariant features is thus only possible in some rare cases of self-adjoint transition operators. Both SFA and PVF substitute therefore a "similar" self-adjoint transition operator for $\hat{P}^{\pi}$. SFA employs the *symmetrized* operator $\hat{P}^{\pi}_s := \frac{1}{2}\hat{P}^{\pi} + \frac{1}{2}(\hat{P}^{\pi})^*$ and PVF assigns equal probability to *all possible* neighbors[26] $\hat{T}^{\pi}$. Analytical comparison of the quality of these approximations is difficult, however.

On the other hand, the class of MDPs for which SFA features are subspace-invariant *contains* the class for which PVF are. To see this, imagine a transition kernel $T^{\pi}$ for which PVF are subspace-invariant, which implies that for each state there exists a uniform distribution to end up in the set of its neighbors, with symmetric neighborhood relationships. As $\hat{T}^{\pi}$ is thus self-adjoint, any ergodic Markov chain from this kernel will yield subspace-invariant SFA features. Reversely, one can construct a transition kernel $P^{\pi}$ with a self-adjoint transition operator but without uniform transition probabilities. PVF would no longer correspond to eigenfunctions of $\hat{P}^{\pi}$ and would thus not be subspace-invariant. SFA can in this sense be seen as a generalization of PVF.

Within the class of MDPs with self-adjoint transition operators, however, one can make some strong claims regarding *value function approximation* with LSTD (Section 2.3).

**Lemma 8** *Let $\{\phi_i\}_{i=1}^{p}$ denote any p SFA features from a MDP with self-adjoint transition operator, then the LSTD fixed point $f^{\pi} = \hat{\Pi}^{\phi}_{\xi}\left[\hat{B}^{\pi}[f^{\pi}]\right]$ and the projection of true value function $v^{\pi} = \hat{B}^{\pi}[v^{\pi}]$ coincide, that is*

$$f^{\pi}(x) \;=\; \hat{\Pi}^{\phi}_{\xi}[v^{\pi}](x) \;=\; \sum_{i=1}^{p} \langle r^{\pi}, \phi_i \rangle_{\xi}\, \tau_i\, \phi_i(x)\,, \quad \forall x \in \mathcal{Z}, \quad \tau_i := \left(1-\gamma+\tfrac{\gamma}{2}\mathcal{S}(\phi_i)\right)^{-1}.$$

**Proof** *see Appendix A, Page 2110.* ■

Lemma 8 implies that for SFA features of symmetric transition models, the bound of Tsitsiklis and Van Roy (1997, introduced in Section 2.3) can be dramatically improved:

**Corollary 9** *The approximation error of the LSTD solution $f^{\pi}$ to the true value $v^{\pi}$ for MDPs with self-adjoint transition operators using any corresponding SFA features $\{\phi_i\}_{i=1}^{p}$ is*

$$\left\| v^{\pi} - f^{\pi} \right\|_{\xi} \quad = \quad \left\| v^{\pi} - \hat{\Pi}^{\phi}_{\xi}[v^{\pi}] \right\|_{\xi}.$$

26. In the discrete case these are all observed transitions, in the continuous case neighborhood relationships are based on similarities in observation space $\mathcal{Z}$ (Mahadevan and Maggioni, 2007).

An analogous proposition can be made for PVF, but for a smaller subset of MDPs. Equivalent fixed point solutions for $p$ reward-based features, on the other hand, do not appear generally possible as the behavior beyond $p-1$ time steps is not encoded (see Theorem 7). However, it is easy to see that *finite horizon* solutions can be computed *exactly* by projected value iteration (finite application of the projected Bellman operator, see, e.g., Bertsekas, 2007).

**Corollary 10** *Finite horizon value functions* $v_h^{\pi} := (\hat{B}^{\pi})^h[r] = (\hat{\Pi}_{\xi}^{\phi^K}[\hat{B}^{\pi}])^h[r]$ *can be computed exactly up to horizon* $h = p-1$ *by projected value iteration with Krylov base* $\{\phi_i^K\}_{i=1}^p$.

The conclusions from Theorems 6 & 7 and Corollaries 9 & 10 are very similar: SFA/PVF optimally approximate/generalize *infinite-horizon* value functions for a subset of possible MDPs, whereas reward-based features represent/generalize *finite-horizon* value functions exactly without any such restrictions.

There have also been attempts to join both types of basis functions by selecting the subspace-invariant feature most similar to the current Bellman error (Petrik, 2007; Parr et al., 2008). To motivate this, Parr et al. (2007) gave a lower bound for the approximation-bound improvement if a BEBF feature $\phi_p^B$ is added to the set $\Phi_{p-1}^B := \{\phi_i^B\}_{i=1}^{p-1}$:

$$\left\|v^{\pi} - \hat{\Pi}_{\xi}^{\Phi_{p-1}^B}[v^{\pi}]\right\|_{\xi} - \left\|v^{\pi} - \hat{\Pi}_{\xi}^{\Phi_p^B}[v^{\pi}]\right\|_{\xi} \quad \geq \quad \left\|v^{\pi} - f^{(p-1)}\right\|_{\xi} - \left\|v^{\pi} - \hat{B}\left[f^{(p-1)}\right]\right\|_{\xi},$$

where $f^{(p-1)} \in \mathcal{F}_{\Phi_{p-1}^B}$ is the LSTD fixed point solution based on $\Phi_{p-1}^B$. One can observe that for each added feature the bound shrinks by the Bellman error function $\phi_p^B$. The PVF feature with the highest correlation to $\phi_p^B$ is thus a good subspace-invariant choice.

We can provide an even stronger assertion about the approximation error of SFA features here. Theorem 11 does not rely on the knowledge of a current LSTD solution, but on the similarity of reward function $r^{\pi}$ and SFA feature $\phi_p$. Given SFA features and reward, the basis can thus be selected *before* training begins.

**Theorem 11** *Let* $\xi$ *be the steady state distribution on* $\mathcal{Z}$ *of a MDP with policy* $\pi$ *and a self-adjoint transition operator in* $L^2(\mathcal{Z},\xi)$. *Let further* $\Phi_p = \{\phi_i\}_{i=1}^p$ *be any set of p SFA features and* $v^{\pi} \in L^2(\mathcal{Z},\xi)$ *the true value of the above MDP. The improvement of the LSTD solution* $f^{(p)} := \hat{\Pi}_{\xi}^{\Phi_p}\left[\hat{B}^{\pi}[f^{(p)}]\right]$ *by including the p'th feature is bounded from below by*

$$\left\|v^{\pi} - f^{(p-1)}\right\|_{\xi} - \left\|v^{\pi} - f^{(p)}\right\|_{\xi} \quad \geq \quad \frac{1-\gamma}{2}\,\frac{\langle r^{\pi},\phi_p\rangle_{\xi}^2}{\|r^{\pi}\|_{\xi}}\,\tau_p^2, \qquad \tau_p := \left(1-\gamma+\tfrac{\gamma}{2}\mathcal{S}(\phi_p)\right)^{-1}.$$

**Proof** *see Appendix A, Page 2111.* ■

The bound improves with the similarity to reward function $r^{\pi} \in L^2(\mathcal{Z},\xi)$. The factor $\tau_p$, defined in Lemma 8, is inversely related to the slowness of the feature $\phi_p$. In $L^2(\mathcal{Z},\xi)$ we can guarantee[27] for

27. Lemma 5, Page 2085, shows that the slowness of eigenfunction $\phi_p$ of self-adjoint $P^{\pi}$ is related to the corresponding eigenvalue $\lambda_p$ by $\mathcal{S}(\phi_p) = 2(1-\lambda_p)$. Eigenvalues can be negative, but since $\lim_{p\to\infty}|\lambda_p| = 0$, every *finite* set of SFA features for *infinite* state/observation spaces will correspond to nonnegative eigenvalues only. In finite state spaces or in general RKHS with finite support, for example, in all sparse kernel algorithms, one can only guarantee $0 < \mathcal{S}(\phi_p) \leq 4$ and $\lim\limits_{p\to\infty}\lim\limits_{\gamma\to 1}\tau_p = \frac{1}{2}$.

infinite state/observation spaces that $\mathcal{S}(\phi_1) > 0$ and $\lim_{p\to\infty} \mathcal{S}(\phi_p) = 2$ that

$$\lim_{\gamma\to 0} \tau_p \quad = \quad 1\,, \qquad \lim_{\gamma\to 1} \tau_p \quad = \quad \frac{2}{\mathcal{S}(\phi_p)} \quad \geq \quad 1\,, \qquad \lim_{p\to\infty} \lim_{\gamma\to 1} \tau_p \quad = \quad 1\,.$$

One could use this bound to select the best feature set for a given MDP, similar to *matching pursuit* approaches (Mallat and Zhang, 1993). However, this is beyond the scope of this article and left for future works.

### 4.3 Encoding Anticipated Value Functions

The last subsection analyzed the properties of SFA for value function approximation of an MDP at hand. Constructed features still need to be represented somehow, for example with a RKHS or some larger set of given basis functions. Reward-based features like BEBF can reduce the value estimation time by incrementally increasing the feature set by the current Bellman error. Strictly, there is no reason to *remember* those features, though. One could instead simply remember the current value estimate. And since the features depend on the reward function, reusing them to solve another MDP is out of the question.

Subspace-invariant features, on the other hand, do not depend on the reward function but are very expensive to construct. This raises the question of *when* these features are actually computed. For example, constructing $p$ RSK-SFA features based on a Markov chain of $n$ observations with a sparse subset of $m$ support observations yields a computational complexity of $\mathcal{O}(m^2 n)$. Sparse kernel LSTD (Xu, 2006) alone exhibits the same complexity but makes use of the full span of the sparse subset, instead of only a $p$-dimensional subspace thereof. Computing features and the value function at the same time therefore does not yield any computational advantage.

Alternatively, one could rely on previously experienced "similar" MDPs to construct the basis functions (*transfer learning*, Taylor and Stone, 2009). Ferguson and Mahadevan (2006) and Ferrante et al. (2008) followed this reasoning and constructed PVF out of experiences in MDPs with the same transition, but different reward model. This section aims to analyze this transfer effect without going into the details of how to choose the MDPs to learn from.

Instead of defining "similar" MDPs, we ask how well one can approximate all value functions for a set of *anticipated tasks* $\mathcal{M}$. The set of all value functions one might encounter during *value iteration* is huge. For LSTD, however, one only has to consider fixed points $f^\pi \in L^2(\mathcal{Z}, \xi)$ of the combined operator $\hat{\Pi}_\xi^\phi \left[\hat{B}^\pi[\cdot]\right]$ (see Section 2.3). Note that there is a unique fixed point $f^\pi$ for every policy $\pi$ from the set of allowed policies $\Omega$, for example all deterministic policies. Moreover, Tsitsiklis and Van Roy (1997) have derived the upper bound

$$\left\| v^\pi - f^\pi \right\|_\xi \leq \frac{1}{\sqrt{1-\gamma^2}} \left\| v^\pi - \hat{\Pi}_\xi^\phi \left[ v^\pi \right] \right\|_\xi ,$$

which means that the approximation error (left hand side) is bounded by the projection error of *true* value function $v^\pi \in L^2(\mathcal{Z}, \xi)$ onto $\mathcal{F}_\phi$ (right hand side). It stands to reason that a set of basis functions which minimizes the right hand side of this bound for all tasks in $\mathcal{M}$ and policies from $\Omega$ according to their occurrence can be called *optimal* in this sense.

**Definition 12** *A set of p basis functions* $\{\phi_i\}_{i=1}^p \subset L^2(\mathcal{Z},\xi)$ *is called "optimal" w.r.t. the distributions* $\rho : \mathcal{B}(\mathcal{M}) \to [0,1]$ *and* $\omega : \mathcal{B}(\Omega) \to [0,1]$*, if they are a solution to*

$$\inf_{\phi \in (L^2(\mathcal{Z},\xi))^p} \mathbb{E}\left[ \underbrace{\left\| v_m^\pi - \hat{\Pi}_\xi^\phi \left[ v_m^\pi \right] \right\|_\xi^2}_{\textit{(bound)}} \;\middle|\; \begin{array}{l} m \sim \rho(\cdot) \\ \pi \sim \omega(\cdot) \end{array} \right].$$

The expectation integrates over all true value functions $v_m^\pi$ which are determined by all policies $\pi \in \Omega$, drawn from some distribution $\omega : \mathcal{B}(\Omega) \to [0,1]$ (e.g., uniform) and all tasks $m$ drawn from distribution[28] $\rho : \mathcal{B}(\mathcal{M}) \to [0,1]$. A similar definition of optimality has been proposed in the context of *shaping functions*[29] (Snel and Whiteson, 2011). Other definitions of optimality are discussed in Section 6.2.

The optimization problem in Definition 12 has no general analytic solution. In particular there is no solution for one MDP and all policies, which would be ideal for policy iteration (e.g., LSPI, Section 2.4). There is another special case, however, which demonstrates the setting under which SFA extracts nearly optimal approximation spaces.

For a fixed policy $\pi \in \Omega$ and task $m \in \mathcal{M}$ one can calculate the exact value function $v^\pi \in L^2(\mathcal{Z},\xi)$. Let $(\hat{I} - \gamma \hat{P}^\pi)^{-1}$ denote the inverse operator[30] to $(\hat{I} - \gamma \hat{P}^\pi)$, then

$$v^\pi \quad \stackrel{!}{=} \quad \hat{B}^\pi[v^\pi] \quad = \quad r^\pi + \gamma \hat{P}^\pi[v^\pi] \quad = \quad (\hat{I} - \gamma \hat{P}^\pi)^{-1}[r^\pi].$$

Substituting this into Definition 12, one can give an analytic solution $\phi_i \in L^2(\mathcal{Z},\xi), i \in \{1,\ldots,p\}$, for all tasks within the same environment,[31] that is, $\mathcal{M} := \{(\mathcal{X},\mathcal{A},P,r) \mid r \sim \rho\}$, restricted to the sampling policy $\pi$, that is, $\Omega := \{\pi\}$. The key insight is that the only allowed difference between tasks is the expected reward function $r^\pi : \mathcal{Z} \to \mathbb{R}$. If we do not constrain the possible reward functions (e.g., all states are possible goals for navigation), their statistics $\rho$ can be described as a *white noise functional* in $L^2(\mathcal{Z},\xi)$ (Holden et al., 2010, see also Footnote 25 on Page 2085).

**Theorem 13** *For any infinite ergodic Markov chain with steady state distribution* $\xi$ *over state space* $\mathcal{Z}$*, SFA selects features from function set* $\mathcal{F} \subset L^2(\mathcal{Z},\xi)$ *that minimize an upper bound on the optimality criterion of Definition 12 for sampling policy* $\pi$ *and discount factor* $\gamma > 0$*, under the assumption that the mean-reward functions* $r^\pi : \mathcal{Z} \to \mathbb{R}$ *are drawn from a white noise functional in* $L^2(\mathcal{Z},\xi)$*.*

**Proof:** see Appendix A, Page 2111. ■

The main result of Theorem 13 is that under the above assumptions, SFA approximates the *optimal basis functions* of Definition 12. To be exact, the SFA objective minimizes a *bound* on the optimality criterion. A closer look into the the proof of Theorem 13 on Page 2111 shows that the exact solution in Definition 12 requires a bias-free estimation of the term $\left\| (\hat{P}^\pi)^*[\phi_i] \right\|_\xi^2$, which is impossible without *double sampling* (see, e.g., Sutton and Barto, 1998). We argue therefore that *SFA constitutes the best approximation to optimal features one can derive using a single Markov chain*. Note that unlike the results in Sections 4.1 and 4.2, this conclusion is not restricted to self-adjoint transition operators.

28. To define a proper distribution $\rho$ one must formally define all anticipated MDPs in $\mathcal{M}$ over the union of all involved state-action spaces. See Snel and Whiteson (2011) for an example of such an approach.
29. In the context of value iteration, shaping functions are equivalent to an initialization of the value function.
30. The existence of such an operator is shown in Lemma 14, Page 2112.
31. With the same state (observation) space $\mathcal{Z}$, action space $\mathcal{A}$ and transition kernel $P$. This class of tasks is also called *variable-reward* tasks (Mehta et al., 2008) and applies for example when a flying robot needs to execute different maneuvers, but is constraint by the same aerodynamics (Abbeel et al., 2007).

### 4.4 Conclusion

Although no direct relationship has been proven, this section has provided evidence for a strong connection between diffusion distances and value function approximation. Our analysis suggests that *reward-based* features can represent finite-horizon value functions exactly. They do not generalize to different MDPs or policies and can thus as well be forgotten after the value estimate is updated. *Subspace-invariant* features, on the other hand, approximate infinite-horizon values optimally if the transition kernel adheres to some restrictions. Moreover, we argue that even in the general case, SFA provides the best possible construction method based on a single Markov chain. Computational complexity prevents this class of features from performing cost-efficient dimensionality reduction for LSTD, though. On the other hand, subspace-invariant features provide on average an optimal basis for *all* reward functions within the same environment. This optimality is still restricted to the sampling policy $\pi$, but SFA features should have an advantage over PVF here, as they are subspace-invariant for a much *larger* class of MDPs.

Using such features effectively for transfer learning or within policy iteration requires them to perform well for other policies $\pi'$, in other words to induce little *per-feature errors* when applied to $\hat{P}^{\pi'}$. In the absence of theoretical predictions a uniform sampling-policy $\pi$ appears to be a reasonable choice here. Note that depending on the transition kernel $P^{\pi}$, this can still yield an arbitrary steady state distribution $\xi$. PVF features are in the limit not affected by $\xi$ and *importance sampling* should be able to compensate the dependence of SFA (see Section 3.5). Theoretical statements about the influence of sampling policy and steady state distribution on SFA and PVF per-feature errors with other policies, however, are beyond the scope of this article and left for future works. See Section 6.1 for a discussion.

*Although SFA is more sensitive to the sampling policy than PVF, the presented analysis suggests that it can provide better approximation spaces for value estimation, that is, LSTD.*

## 5. Empirical Analysis

This section empirically evaluates the the construction of approximation spaces in light of the theoretical predictions of Section 4. Our analysis focuses on three questions:

1. How well does LSTD estimate the value function of a given Markov chain?

2. How good is the performance of policies learned by LSPI based on a random policy?

3. How does this performance depend on the approximation space metric?

We start with evidence for the relationship (hypothesized in Section 4.1) between *per-feature errors* (see Section 3) and how *self-adjoint* a transition operator is. Furthermore, Section 4.2 predicts that the set of MDPs for which PVF are *subspace-invariant* is a subset of the respective set of SFA. To test both possibilities we evaluated the first two questions on two discrete benchmark tasks: the *50-state chain* in Section 5.2 and the more complicated *puddle-world task* in Section 5.3. The third question can not be answered with a discrete metric. To test the influence of the observation space metric, we designed a simple but realistic robot navigation task with continuous state and observation spaces (Section 5.4). A robot must navigate into a goal area, using first-person video images as observations. Results are presented in Sections 5.5, 5.6 and 5.7.

### 5.1 Comparison of Subspace Invariance

In Section 4.1 we stated our intuition that the *symmetrized transition operator* of SFA approximates the true transition operator better than the *neighborhood operator* of PVF. Here we want to substantiate this intuition by testing the *per-feature errors* of theoretical SFA and PVF solutions applied to randomly generated MDPs.

#### 5.1.1 THEORETICAL FEATURES

Discrete MDPs allow an exact calculation of the objectives described in Section 3. To test the limit case of an infinite Markov chain one can generate *theoretical* features, which is straight forward[32] for SFA. These features depend on *steady state distribution* (s.s.d.) $\xi$. Changing $\xi$ has a surprising effect on per-feature errors, in particular if one assumes a uniform $\xi$. To demonstrate this effect, all experiments with theoretical features also include this case.

Theoretical PVF, on the other hand, require a proper definition of *neighborhood*. As all states could be connected by the transition kernel, *transition possibility* (as in Section 3.3) is not always an option. We followed the *k-nearest neighbor* approach of Mahadevan and Maggioni (2007) instead and defined[33] the $k$ most probable transitions as neighbors.

For each feature $\phi_i$ from a set $\Phi_p := \{\phi_1, \ldots, \phi_p\}$ one can calculate the *per feature error* $\Delta_i^{\Phi_p} \in L^2(\mathcal{X}, \xi)$ (see Section 3). To measure how how strongly $\Phi_p$ diverges from *subspace invariance*, we add the norms of all $p$ error functions together, that is, $\sum_{i=1}^{p} \|\Delta_i^{\Phi_p}\|_\xi$. This yields a measure of subspace invariance for each set of $p$ features. To compare construction methods we also calculated the mean of the above measure over $p \in \{1, 2, \ldots, 100\}$.

#### 5.1.2 SUBSPACE-INVARIANCE AND SELF-ADJOINT TRANSITION OPERATORS

To investigate whether SFA or PVF features approximate arbitrary transition models better, we tested the per-feature errors $\Delta_i^{\Phi_p}$ of random MDPs. 100 MDPs with $d = 100$ states each were created. Each state is connected with 10 random future states and each connection strength is uniformly i.i.d. drawn. The resulting matrix is converted into a probability matrix $\mathbf{P}^{\boldsymbol{\pi}}$ by normalization. SFA features are subspace-invariant for *self-adjoint* transition operators and PVF only for a subset thereof. As the above generated transition matrices are usually not self-adjoint, we repeatedly applied a *symmetrization* operator $\hat{G} : \mathbb{R}^{d \times d} \to \mathbb{R}^{d \times d}$ to each matrix $\mathbf{P}$, that is, $\hat{G}[\mathbf{P}]_{ij} := (P_{ij} + P_{ji})/(1 + \sum_k P_{jk})$. With each application the resulting transition matrices come closer and closer to be self-adjoint.

Figure 3 shows the measure for subspace invariance for theoretical PVF and SFA with both sampling distributions $\xi$. The left figure plots this measure against the feature set size $p$. One can observe that all methods show similar errors for the original asymmetric MDP (solid lines). Application of the symmetrization operator (dashed lines), on the other hand, yields a clear advantage for one SFA method. This becomes even more apparent in the right plot of Figure 3. Here the mean measure over all feature set sizes $p$ is plotted against the number of symmetrization operator applications. One can observe that (in difference to PVF) the per-feature errors of both SFA meth-

32. SFA minimizes the slowness, in the limit according to Lemma 5: $\mathcal{S}(\phi_i) = 2\langle \phi_i, (\hat{I} - \hat{P}^{\boldsymbol{\pi}})[\phi_i] \rangle_\xi$. Let $\mathbf{P}^{\boldsymbol{\pi}}$ be the transition matrix and $\boldsymbol{\Xi}$ a diagonal matrix of steady state distribution $\xi$, which is the left eigenvector to the largest eigenvalue of $\mathbf{P}^{\boldsymbol{\pi}}$. Expressing the objective in matrix notation, the theoretical SFA features are the eigenvectors to the smallest eigenvalues of the symmetric matrix $2\boldsymbol{\Xi} - \boldsymbol{\Xi}\mathbf{P}^{\boldsymbol{\pi}} - (\boldsymbol{\Xi}\mathbf{P}^{\boldsymbol{\pi}})^\top$.

33. We tested $k \in \{1, 2, 5, 10, 20, 50, 100\}$ and present here the best results for $k = 10$.

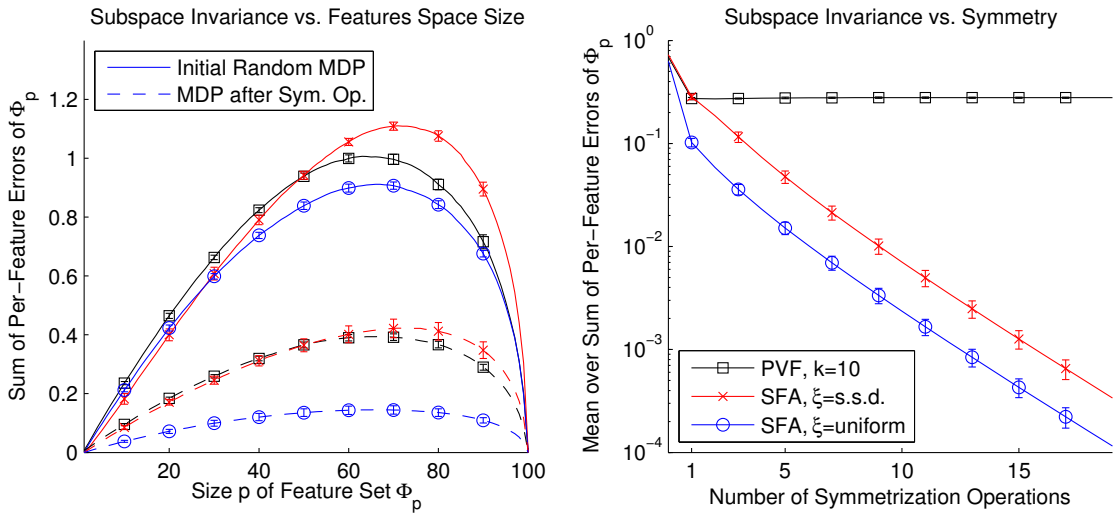



Figure 3: Subspace invariance of SFA and PVF features of random MDPs and symmetrized versions thereof. The left plot shows $\sum_{i=1}^{p} \|\Delta_i^{\Phi_p}\|_\xi$ for different feature set sizes $p$. The first symmetrization reduces per-feature errors of both methods, but all following symmetrization operations reduce only the error of SFA which is illustrated in the right plot. All means and standard deviations are w.r.t. 100 random MDPs. Note that the scale of per-feature errors differs between plots.

ods shrink the more self-adjoint the transition operator becomes. Also, SFA features with uniform distribution $\xi$ are roughly 3 times as subspace-invariant as original SFA features with steady state distribution $\xi$.

We conclude that PVF and SFA methods approximate arbitrary asymmetric MDPs equally well. However, the more self-adjoint the transition operator, the larger the advantage of SFA. Furthermore, SFA features based on a uniform distribution $\xi$ are on average more subspace-invariant than those based on the steady state distribution.

### 5.2 50-state Chain-Benchmark Task

First we investigate how well a basis constructed from a Markov chain can approximate the corresponding value function. The employed *50-state chain task* is based on a problematic 4-state MDP by Koller and Parr (2000) and has been extended in various variations by Lagoudakis and Parr (2003). Here we adopt the details from Parr et al. (2007). The task has a very similar transition- and neighborhood structure.

#### 5.2.1 TASK

50 states are connected to a chain by two actions: move left and right. Both have a 90% chance to move in their respective direction and a 10% chance to do the opposite. Non-executable transitions at both ends of the chain remain in their state. Only the 10th and 41th state are rewarded. Executing any action there yields a reward of $+1$. The task is to estimate the value function with a discount

factor $\gamma = 0.9$ from a Markov chain. Note that transition probabilities of a random policy equal the neighborhood relationships. SFA features for this policy should therefore equal PVF features.

#### 5.2.2 ALGORITHMS

We compare discrete versions of *slow feature analysis*, *proto value functions* and *Krylov bases* as described in Section 3. SFA feature sets also contained a constant feature. In this and the following experiments, higher Krylov bases have proven to be too similar for stable function approximation. We therefore orthonormalized each feature w.r.t. its predecessors, which solved the problem. After construction, the features were used to estimate the value function from the same training set with LSTD. Sampling influences value estimation here and to avoid the resulting bias we measure the difference (in some norm) to the LSTD solution with a *discrete* representation.

#### 5.2.3 RESULTS

To explore the effect of the sampling policy, we tested the (non-deterministic) uniform and the (deterministic) optimal policy. Figure 4 plots mean and standard deviation of the LSTD solution difference in $L_2$ norm w.r.t. 1000 trials and 4000 samples each. To make sure all states are visited, an optimal policy trial is sampled in 40 trajectories with random start states and 100 samples each. Training sets that did not visit all states were excluded. As reported by previous works (Petrik, 2007; Parr et al., 2008; Mahadevan and Liu, 2010), reward-based features like Krylov bases perform in this task much better than subspace-invariant features. Solutions with PVF and SFA features are virtually identical for the random policy, as the transition probabilities of SFA equal the neighborhood relations of PVF. There are distinguishable differences for the optimal policy, but one can hardly determine a clear victor. Using the $L_\infty$ norm for comparison (not shown) yields qualitatively similar results for the two feature spaces. Policy iteration did also not yield any decisive differences between SFA and PVF (not shown). In conclusion, the 50-state chain appears to belong to the class of MDPs for which features learned by SFA and PVF are not always identical, but *equally* able to estimate the value functions with LSTD.

### 5.3 Puddle-world–Benchmark Task

The puddle-world task was originally proposed by Boyan and Moore (1995), but details presented here are adapted from Sutton (1996). It is a continuous task which we discretize in order to compare reward-based features. This is a common procedure and allows to evaluate robustness by running multiple discretizations. In comparison to the 50-state chain the task is more complex and exhibits differing transition- and neighborhood-structures.

#### 5.3.1 TASK

The state space is a two dimensional square of side length 1. Four actions move the agent on average 0.05 in one of the compass directions. The original task was almost deterministic (Sutton, 1996) and to make it more challenging we increased the Gaussian noise to a standard deviation of 0.05. Centered in the $(1,1)$ corner is an absorbing circular goal area with radius 0.1. Each step that does not end in this area induces a punishment of $-1$. Additionally, there exist two puddles, which are formally two lines $(0.1,0.75) \longleftrightarrow (0.45,0.75)$ and $(0.45,0.4) \longleftrightarrow (0.45,0.8)$ with a radius of 0.1 around them. Entering a state less than 0.1 away from one of the center-lines is pun-

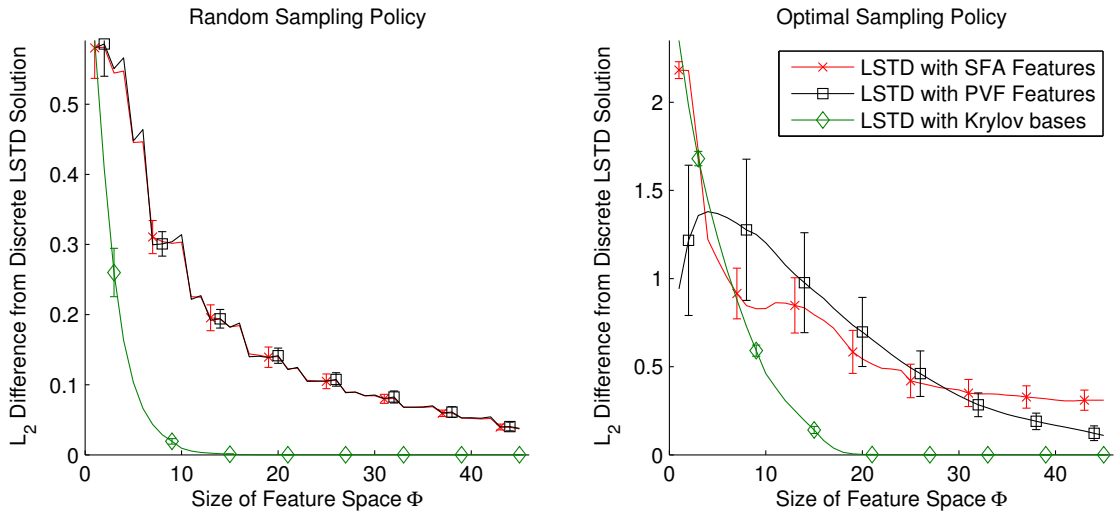



Figure 4: Difference in $L_2$ norm between approximated and discrete LSTD solutions vs. feature space size for random (left plot) and optimal (right plot) sampling policies in the 50-state chain. Means an standard deviations w.r.t. 1000 trials. $L_\infty$ differences decrease slower but show otherwise the same qualitative trends.

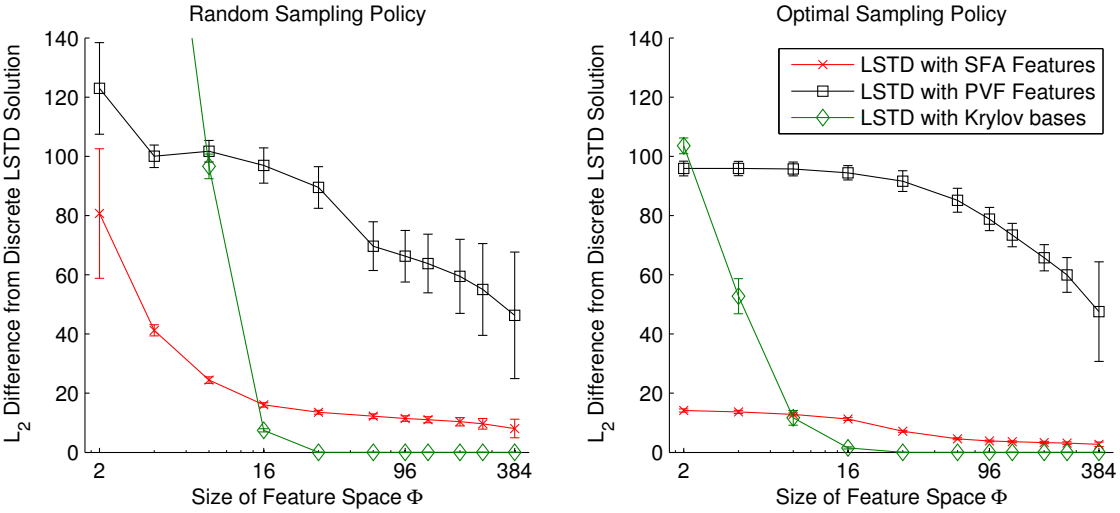



Figure 5: Difference in $L_2$ norm between approximated and discrete LSTD solutions vs. feature space size in puddle-worlds. Means and standard deviations are w.r.t. 10 training sets for each state space size $\{20\times 20, 25\times 25, \ldots, 50\times 50\}$, sampled with random (left plot) or optimal (right plot) policies. Note the logarithmic x-axis. Measuring the differences in $L_\infty$ norm yields the same qualitative trends.

ished with $-400$ times the distance into the puddle. The task is to navigate into the goal, collecting as little punishment as possible. Seven discretizations with horizontal and vertical side length of $\bar{s} \in \{20, 25, \ldots, 50\}$ states were tested. Transition probabilities of states $\boldsymbol{x}_i$ to states $\boldsymbol{x}_j$ are the normalized *continuous* probabilities (average movement $\boldsymbol{\mu}_a$ plus Gaussian noise with standard deviation $\tilde{\sigma}$), that is

$$P_{ij}^a := \frac{\exp(-\frac{1}{2\tilde{\sigma}^2}\|\boldsymbol{x}_i + \boldsymbol{\mu}_a - \boldsymbol{x}_j\|_2^2)}{\sum_{k=1}^{\bar{s}^2}\exp(-\frac{1}{2\tilde{\sigma}^2}\|\boldsymbol{x}_i + \boldsymbol{\mu}_a - \boldsymbol{x}_k\|_2^2)}, \qquad \forall i, j \in \{1, \ldots, \bar{s}^2\}, \quad \forall a \in \{1, \ldots, 4\}.$$

### 5.3.2 ALGORITHMS

We evaluated discrete versions of *slow feature analysis* (SFA), *proto value functions* (PVF) and orthonormalized *Krylov bases*. To test the influence of sampling on feature construction we also evaluated the theoretical features of all three algorithms[34] (see Section 5.1). The algorithms were trained with a long (uniform) random policy Markov chain $\{\boldsymbol{x}_t\}_{t=1}^n$ in both LSTD and LSPI evaluations. To see the effect of different policies on LSTD, we trained all three on the *optimal* policy as well. In this case the training set consists of randomly initialized trajectories of length 20 to overcome sampling problems. We chose $n = 80\,\bar{s}^2$, as large state spaces require more samples to converge.

All constructed approximation spaces were tested with LSTD and LSPI[35] under discount factor $\gamma = 0.99$ on the same training sequence $\{\boldsymbol{x}_t\}_{t=1}^n$ used in feature construction. Policy iteration ended if the value of *all* samples differed by no more than $10^{-8}$ or after 50 iterations otherwise. To investigate asymptotic behavior we also tested LSPI with all state-action pairs and *true* mean future states as training set, corresponding to the limit of an infinite Markov chain. Performance of a (deterministic) policy learned with LSPI is measured by the *mean accumulated reward* of 1000 trajectories starting at random states. A test trajectory terminates after 50 transitions or upon entering the goal area.

### 5.3.3 LSTD EVALUATION

We tested the LSTD approximation quality as in Section 5.2. Figure 5 shows the difference in $L_2$ norm between the LSTD solution based on the constructed features and a discrete state representation vs. the number of employed features $p \in \{2, 4, 8, 16, 32, 64, 96, 128, 192, 256, 384\}$. Means and standard deviations are w.r.t. 10 training sets for each state space size $\{20\times 20, 25\times 25, \ldots, 50\times 50\}$, sampled with random (left plot) or optimal (right plot) policies. Note in comparison to Figure 4 that the x-axis is logarithmic. Reward-based Krylov bases have a clear advantage for $p \geq 16$ features (and reach perfection for $p \geq 64$), similar to the 50-state chain task. Fewer SFA features, on the other hand, capture the value function much better. This has also been observed for large discount factors $\gamma$ when comparing BEBF and modified PVF (Mahadevan and Liu, 2010). The different encoding of diffusion distances (Theorems 6 & 7, Page 2085) provides a good explanation for this effect: Krylov bases represent finite-horizon value functions perfectly (Corollary 10, Page 2087),

34. We tested theoretic PVF with $k \in \{5, 10, 15, 20, 25, 50\}$ and chose $k = 10$. For larger $k$ we observed slowly degrading performance, which is more pronounced under realistic LSPI sampling.

35. As convergence to the optimal policy can not be guaranteed for LSPI, the eventual policy depends also on the initial (randomly chosen) policy $\boldsymbol{\pi}_0$. In difference to Lagoudakis and Parr (2003) we used throughout this article a *true* (non-deterministic) random policy $\boldsymbol{\pi}_0(a|\boldsymbol{z}) = \frac{1}{|\mathcal{A}|}, \forall \boldsymbol{z} \in \mathcal{Z}, \forall a \in \mathcal{A}$. This heuristic appeared to be more robust in large feature spaces.

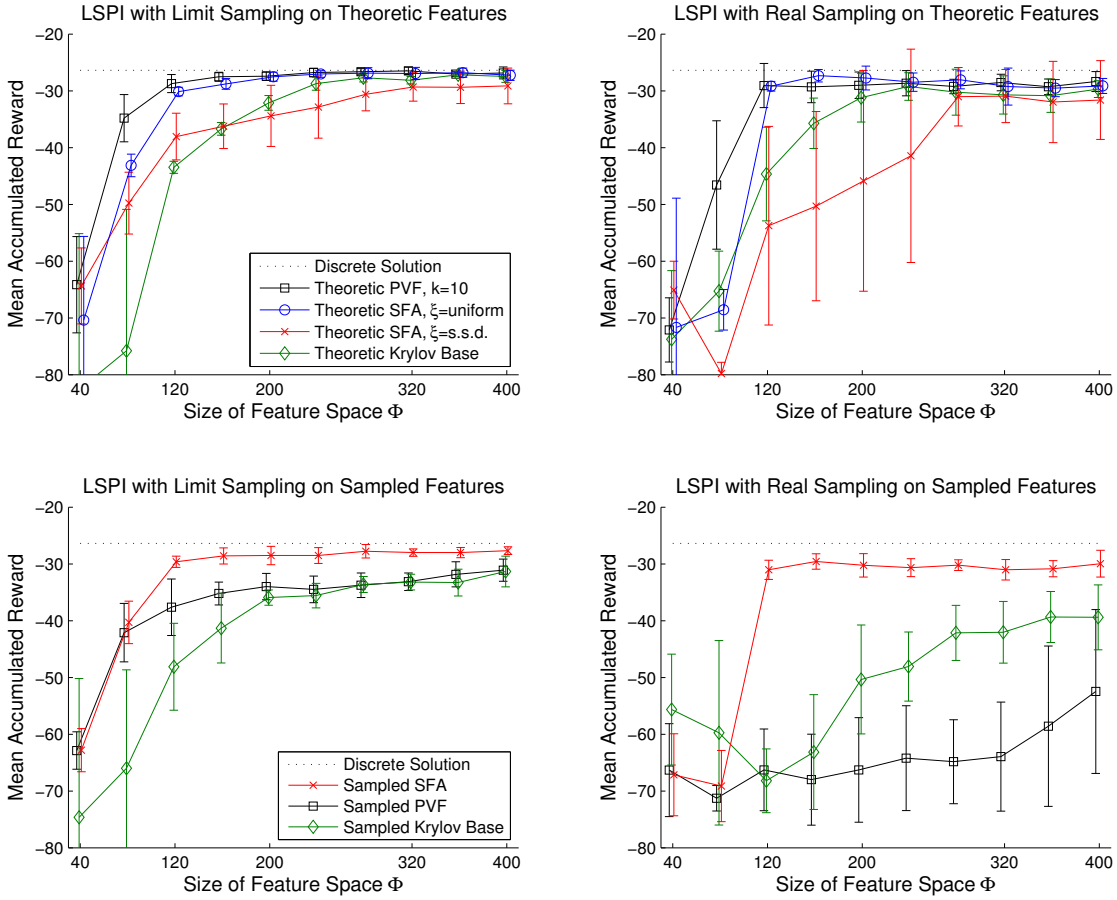



Figure 6: Performance of LSPI with optimal (left column) and realistic sampling (right column) in puddle-worlds. The upper row employed theoretical features and the lower row features constructed from samples. Means and standard deviations are w.r.t. state space sizes $\{20 \times 20, 25 \times 25, \ldots, 50 \times 50\}$.

whereas SFA represents the infinite-horizon values approximatively (Corollary 9, Page 2086). However, Figure 5 clearly supports our hypothesis (proposed in Section 4.4) that *SFA can provide better approximation spaces for value function approximation with LSTD than PVF*. With respect to the previous subsection one should extend this hypothesis by adding *when transition and neighborhood structures are dissimilar*.

### 5.3.4 LSPI EVALUATION

Figure 6 shows the LSPI performance with both theoretic (upper row) and sampled features (lower row). LSPI was trained with an optimal training set (left column) and a realistic sequence drawn by

a random policy (right column). The top-left plot is the most hypothetical and the bottom-right plot the most realistic scenario. Means and standard deviation are w.r.t. state space sizes.

First and foremost, note that the sampled SFA features (crosses, lower row) perform significantly (more than one standard deviation) better, except in the under-fitting regime of 80 features and less. They are also the only sampled features that are robust against LSPI sampling, which can be seen by similar performance in both bottom plots. In comparison with theoretic SFA features, their performance resembles the solution with uniform distribution $\xi$ (circles, upper row), which clearly outperforms SFA features based on the very similar steady state distribution $\xi$ (crosses, upper row). This is surprising, as the two distributions only differ at the borders of the square state space. Similar to Section 5.1, the uniform distribution appears nonetheless to have an advantage here.

Secondly, note that although theoretic PVF features (squares, upper row) rival the best of SFA, sampled PVF features (squares, lower row) are less successful. In the most realistic case the performance appears almost unaffected by additional features. This is consistent with our observations of the LSTD solutions in Figure 5, where sampled PVF features performed equally bad under both policies. Reward-based Krylov Features (diamonds), on the other hand, appear relatively stable through most settings and only cave in at the most realistic scenario. However, note that for LSPI *reward-based* features do not have any (empirical or theoretical) advantage over *subspace-invariant* features.

Although not exactly predicted by theory, we see this as evidence that *discrete SFA can construct better approximation spaces for LSPI than PVF or Krylov bases*. Theoretical PVF may rival SFA, but realistic sampling appears to corrupt PVF features. This advantage of SFA may be lost for other sampling policies, though. As an example, observe the strong influence of minor changes in the sampling distribution $\xi$ on the theoretical SFA solution.

### 5.4 Visual Navigation-Setup

We investigate our third question on Page 2090 with a simple but realistic continuous application task. Continuous state spaces impede the use of reward-based features, but allow an analysis of the presented metric by encoding observations with PCA. Our focal idea is to compare basis construction approaches based on different metrics in the *observation space* and the underlying *state space*. A *visual navigation task* guides a robot into a designated goal area. Observations are first-person perspective images from a head mounted video camera (see Figure 2, Page 2071, for a sketch of the control process). While robot coordinates come close to encoding diffusion distances of random policies, these observations clearly do not. A comparison between PCA and SFA features in these two cases can thus illuminate the role of the observation metric in the construction of continuous basis functions. Adequacy of SFA in this task is demonstrated with a real robot (Section 5.7) and extensively compared with PVF and PCA in a realistic simulation (Section 5.6). Section 5.5 provides an evaluation of the involved sparse subsets.

#### 5.4.1 ROBOT

We used the wheeled PIONEER 3DX robot (Figure 7a), equipped with a head mounted BUMBLEBEE camera for the experiments. The camera recorded mono RGB images from a first-person perspective of the environment in front of the robot with a 66° field of vision (Figure 7b). The robot was able to execute 3 commands: Move approximately 30cm forwards; turn approximately 45° left or right.

a) Robot b) Robot View c) Simulator View

Figure 7: a) PIONEER 3DX wheeled robot. b) A first-person perspective image from the robot camera. c) Corresponding image of the simulator.

#### 5.4.2 ENVIRONMENT

The robot had to navigate within a rectangular $3m \times 1.8m$ area surrounded by walls, approx. $1m$ in height. We covered the walls with different wallpaper to have a rich texture (sketched in Figure 10). The scenery was well-lit by artificial light. A camera installed at the ceiling allowed us to track the robot's position for analysis. We also ran simulations of the experiment for large scale comparison. Based on photographed textures, the JAVA3D engine rendered images from any position in the simulated environment. Those images were similar to the real experiment, but not photo-realistic (Figure 7c).

#### 5.4.3 TASK

Starting from a random start position, the robot has to execute a series of actions (move forward, turn left, turn right) that lead to an unmarked goal area in as few steps as possible without hitting the walls. Learning and control are based on the current camera image $\boldsymbol{z}_t$ and a corresponding reward signal $r_t \in \{-1, 0, +1\}$ indicating whether the robot is in the goal area ($r_t = +1$), close to a wall ($r_t = -1$) or none of the above ($r_t = 0$).

#### 5.4.4 ALGORITHMS

The algorithms *sparse kernel principal component analysis* (SK-PCA, Section 3.2), *k-nearest-neighbor extension of proto value functions* (kNN-PVF, Section 3.3) and *regularized sparse kernel slow feature analysis* (RSK-SFA Section 3.4), were implemented[36] using a Gaussian kernel $\boldsymbol{\kappa}(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2), \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{Z}$. To ensure the respective *function sets* from which the basis functions were chosen are roughly equivalent, we set the Gaussian width parameter $\sigma = 5$ for all algorithms. Runtime deviated at most by a factor of 2, as SFA requires two eigenvalue decompositions. The importance sampling modification of SFA described in Section 3.5 assigned a weight[37] of $\vartheta(a_{move}) = 5$ to forward movements and $\vartheta(a_{turn}) = 1$ to rotations. This balanced out *relative velocities* and ensured that the first features encode both spatial and orientational information.

36. RSK-SFA and SK-PCA select functions from a RKHS, based on any positive semi-definite kernel, for example, the Gaussian kernel. kNN-PVF are based on a k-nearest-neighbors graph with edges weighted by a Gaussian kernel of the distance between nodes.
37. We tested other weights with less detail. The results appear stable around $\vartheta(a_{move}) = 5 \pm 1$ but exact statements require an order of magnitude more simulations than we were able to provide for this article.

All discussed algorithms also require a sparse subset of the data. For optimal coverage it appears straightforward to select this subset uniformly in observation space $\mathcal{Z}$. This can be achieved with the *matching pursuit for affine hull maximization* algorithm (MP-MAH, Böhmer et al., 2012). However, we show in Section 5.5 that this intuition is wrong and in fact one should instead select the subset uniformly distributed in the true state space $\mathcal{X}$, which can be achieved by applying MP-MAH on $\mathcal{X}$ instead of $\mathcal{Z}$. As $\mathcal{X}$ is usually not known explicitly, the comparison between algorithms in Section 5.6 is performed with randomly drawn subsets.

#### 5.4.5 SIMULATED EXPERIMENTS

Drawing actions uniformly, we generated 10 independent random walks with 35,000 samples each. The rendered images were brightness corrected and scaled down to $32 \times 16$ RGB pixels, yielding observations $\boldsymbol{z}_t \in \mathcal{Z} \subset [0,1]^{1536}$. Each training set was used to construct one feature space for each of the above algorithms with a sparse subset of 4000 samples (see Section 5.5). The resulting basis functions were applied on the corresponding training set.

The control policy was learned by LSPI on the first $p \in \{2,4,8,\ldots,2048\}$ constructed features $\phi_i : \mathcal{Z} \to \mathbb{R}$ and a constant feature $\phi_0(\boldsymbol{z}) = 1\,, \forall \boldsymbol{z} \in \mathcal{Z}$. The discount factor was $\gamma = 0.9$ and the goal area was located in the lower right corner with a radius of 50cm around x=260cm and y=40cm (see right plot of Figure 10). A distance to the walls of 40cm or less was punished.

The resulting policies were each tested on 200 test trajectories from random start positions. Navigation performance is measured[38] as *fraction of successful trajectories*, which avoid the wall and hit the goal in less than 50 steps.

#### 5.4.6 ROBOT EXPERIMENT

Running the robot requires a large amount of time and supervision, preventing a thorough evaluation. For RSK-SFA the continuous random walk video of approx. 10 hours length was sampled down to approx. 1 Hz and a sparse subset of 8,000 out of 35,000 frames was selected with the MP-MAH algorithm (Böhmer et al., 2012). The first 128 RSK-SFA and one constant feature were applied on a training set of 11,656 *transitions* sampled from the same video. LSPI was trained as in the simulator experiments and evaluated on each 20 test trajectories for the lower right and for a smaller center goal with a radius of 20cm (see Figure 10).

### 5.5 Visual Navigation-Sparse Subset Selection

The analysis of Böhmer et al. (2012) suggests that a sparse subset uniformly distributed in observation space $\mathcal{Z}$ improves the performance of RSK-SFA. We observed the same effect on the *slowness* (not shown), but interestingly not on the *navigation performance* of the respective LSPI solution. Figure 8 plots this performance against the number of features $p$ used for LSPI. Random subset selection (crosses) outperforms MP-MAH selection on $\mathcal{Z}$ with the correct kernel width (upward triangle) significantly. Moreover, random selection yields high performance reliably (small standard deviation), whereas subsets that are uniformly distributed in $\mathcal{Z}$ result in unpredictable behavior for large approximation spaces. Shrinking the kernel parameter $\sigma$ of MP-MAH (*not* of RSK-SFA) decreases the disadvantage as the algorithm converges theoretically to random selection in the limit

38. Other measures are possible. We tested "mean number of steps to goal" and also compared those to an almost optimal policy. However, the resulting plots were qualitatively so similar that we stuck to the simplest measure.

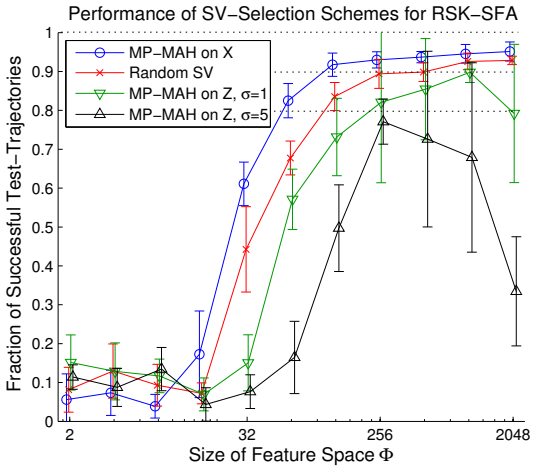



Figure 8: The influence of *sparse subset selection* for RSK-SFA on the performance of LSPI. Mean and standard deviation (over 10 independent training sets) of the navigation performance are plotted against the (logarithmic) number of features used in LSPI. Note that a subset equally distributed in the true state space $\mathcal{X}$ (circles) is most efficient and reliable, whereas for equal distribution in observation space $\mathcal{Z}$ (triangles) LSPI becomes unreliable in large approximation spaces $\mathcal{F}_\phi$.

$\sigma \to 0$ (downwards triangles). Using MP-MAH to select a subset uniformly distributed in the *true state space*[39] $\mathcal{X}$ (circles), however, demonstrates that this is not an over-fitting effect as the learned policies outperform those of random selection significantly. Section 6.3 attempts an explanation of these results and discusses some practical implications for sparse subset selection.

However, in practice $\mathcal{X}$ is usually not explicitly known. The main comparison in Section 5.6 is therefore performed with randomly drawn subsets. Note that our random walk sampled the state space $\mathcal{X}$ almost uniformly, which is the explanation for the good performance of randomly selected subsets. A random selection from biased random walks, for example, generated by other tasks, will severely decrease the navigation performance. We expect a similar effect in other sparse kernel RL methods (e.g., Engel et al., 2003; Xu, 2006).

### 5.6 Visual Navigation-Comparison of Algorithms

The left plot of Figure 9 shows a comparison of the effect of all discussed basis function construction algorithms on the control policy learned by LSPI. Note that the algorithms are based on the same randomly chosen sparse subset of 4000 samples: 4000 orthogonal features extracted by any algorithm span approximation space $\mathcal{F}_{\{\boldsymbol{\kappa}(\cdot,\boldsymbol{s}_i)\}_{i=1}^m}$. Therefore all algorithms perform equally well with enough ($p \geq 1024$) features. One can, on the other hand, observe that both the original RSK-SFA (crosses) and the modified algorithm (circles) outperform SK-PCA (triangles) and kNN-PVF[40]

39. Robot position and orientation coordinates for which the Euclidean metric resembles diffusion distances.
40. We tested the navigation performance based on kNN-PVF basis functions for the kNN parameters $k \in \{10, 25, 50\}$. As the results did not differ significantly, we omitted them here.

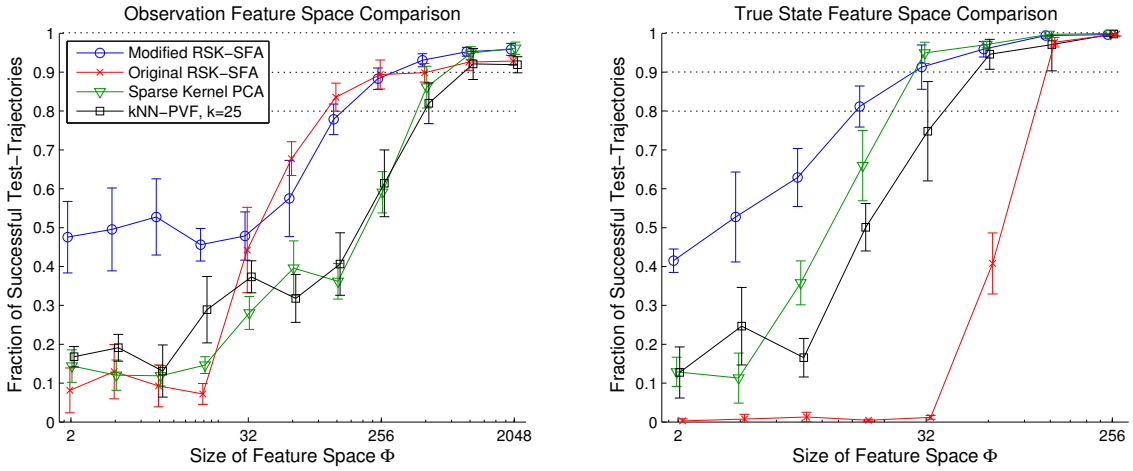



Figure 9: Comparison of subspace-invariant feature space construction algorithms on observations $\boldsymbol{z} \in \mathcal{Z}$ (camera images, left plot) and on the true state $\boldsymbol{x} \in \mathcal{X}$ (robot coordinates, right plot). Mean and standard deviation (over 10 independent training sets) of the navigation performance are plotted against the number of features used in LSPI. Note that the x-axis is logarithmic and differs between plots. The dotted lines indicate 80%, 90% and 100% performance levels.

(squares) significantly in medium sized feature spaces ($32 \leq p < 1024$ features). In particular the 80% and 90% levels of navigation performance are both reached with roughly a quarter of basis functions. We attribute this advantage to approximation spaces encoding diffusion distances rather than similarities in $\mathcal{Z}$.

Close inspection of the feature space revealed that the first RSK-SFA features encode spatial information only (not shown). This is due to the different velocities of rotations and movements of the robot in the true state space $\mathcal{X}$ (see Section 3.5 and Franzius et al., 2007). Consequentially, small feature spaces can not express policies involving rotation and thus perform poorly. The modified algorithm balances this handicap out and yields steady performance in small feature spaces ($2 \leq p < 32$ features). If the necessary orientational components are encoded in the original RSK-SFA basis functions ($p \geq 32$), both algorithms perform comparable as they span (almost) the same approximation space. We conclude that the modified RSK-SFA algorithm is the superior continuous basis function construction scheme, irrespective of feature space size $p$.

### 5.6.1 COMPARISON IN TRUE STATE SPACE $\mathcal{X}$

To confirm the above effect is due to the difference in the observed metric and the diffusion metric constructed by SFA, we run the same experiment with $\mathcal{Z} \approx \mathcal{X}$. For this purpose we applied all basis function construction algorithms on robot coordinates. The true state space $\mathcal{X}$ is supposed to be equipped with a diffusion metric, which should have a constant distance between successive states. We thus divided the spatial coordinates by an average movement of 30cm and the robots orientation by the average rotation of 45°. The kernel width was chosen $\sigma = 2$ to allow sufficient overlap.

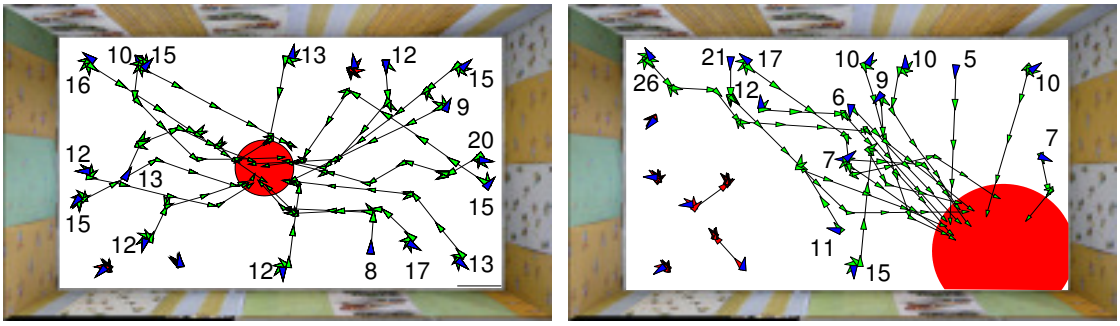



Figure 10: The recorded robot trajectories of a control policy learned by LSPI using RSK-SFA features. The plot shows 40 trajectories with random starting positions and orientation (dark triangles), aiming to hit the circular goal area. The numbers indicate the number of steps the robot required to reach the goal.

The right plot of Figure 9 shows the navigation performance of the learned LSPI policy. Due to the suitability of the observation space, a relative small number[41] of basis functions of $p \geq 128$ suffices for an almost perfect navigation performance with all algorithms. However, relative velocities influence the RSK-SFA solution as well, which renders the original algorithm (crosses) almost useless and demonstrates its sensitivity to the sampling policy. The modified algorithm (circles) performs best here as well, but demonstrates an advantage only in very small feature spaces ($2 \leq p < 32$). Together with the good performance of SK-PCA (triangles), which slightly outperforms the state-of-the-art method kNN-PVF (squares), this can be seen as evidence that methods based on Euclidean distances in $\mathcal{Z} \approx \mathcal{X}$, which are already close to diffusion distances, suffice to learn a good policy in this setup.

#### 5.6.2 CONCLUSION

The results presented in this subsection provide ample evidence for the hypothesis that *SFA can construct better approximation spaces for LSPI than PVF or PCA*, but also demonstrates its sensitivity to the sampling policy. We have empirically shown that the novel modified RSK-SFA algorithm outperforms all continuous basis function construction schemes reviewed for this article. The advantage to baseline method PCA vanishes when the observations already conform to a diffusion metric. This suggests that SFA performs essentially PCA based on diffusion rather than Euclidean distances. It also implies an answer to our third question on Page 2090: *LSPI performance is facilitated by approximation spaces that encode diffusion distances of a uniform random policy*.

### 5.7 Visual Navigation-Robot Demonstration

A thorough reproduction of the experiments from Section 5.6 on a real robot is beyond the scope of this article. Measuring the navigation performance of 200 test trajectories can not easily be automated and requires an enormous amount of supervision. However, we demonstrate how our

41. Note that, in difference to the space of images, this observation space is three dimensional. Instead of (at least potentially) dimensionality reduction, these basis functions form an overcomplete basis.

key method, that is SFA in the form of RSK-SFA, performs on a real robot. After 10 LSPI iterations with 128 RSK-SFA basis functions, the control policy achieved a success rate of 75% in two separate tasks depicted in Figure 10. 17 out of 20 test trajectories hit a centered goal within 20 steps (left plot) and 13 out of 20 trajectories reached the much farther goal area in the lower right corner (right plot). The failed trajectories made at most one forward move and then started to oscillate between right and left rotations. These oscillations also appeared often in successful trajectories whenever the robot switched from one action to another. This can be seen in the large number of steps some trajectories require to reach the goal area. We attribute these oscillations to approximation errors and noise in the basis functions. Whenever the policy changes from one action to another, the respective Q-values must be close-by and small deviations from the true value can drastically influence the greedy action selection. If the wrongfully chosen action is a rotation, the correct reaction would be to rotate back. We observed these never ending oscillations in simulations as well. The problem can usually be diminished by adding more basis functions. The fact that the robot was sometimes able to break the oscillation is an indicator for noise in images and motors. This behavior could have been easily avoided by introducing some simple heuristics, for example "never rotate back" or "in doubt select the previous action". As this article investigates approximation spaces rather than optimizing the visual task itself, we omitted those heuristics in our experiments. For practical implementations, however, they should be taken under consideration.

## 6. Discussion

This section discusses implications of the presented results and points out open questions and potential directions of future research.

### 6.1 SFA Features as Basis Functions for LSPI

Theoretical predictions in Section 4 cover value function estimation with LSTD for the current sampling policy. When LSPI changes this policy, all statements become strictly invalid. Nonetheless, the results in Sections 5.3 and 5.6 demonstrate the applicability of SFA features as basis functions for LSPI. Both experiments also show that an *importance sampling* modification to SFA can yield even better results. Maybe there exists a policy $\tau$ and a state-distribution $\zeta$ which are *optimal* (in the sense of Definition 12) for at least all deterministic policies encountered during LSPI. In the experiments this appeared to be a uniform policy and a uniform state-distribution, but we can not claim any generality based on the presented evidence alone. We still want to suggest a possible explanation:

Uniform $\tau$ and $\zeta$ might be an optimal training sets for LSPI. Koller and Parr (2000) proposed this in light of a pathological 4-state chain MDP (similar to Section 5.2). LSTD weights the importance of value approximation errors and predicts thus accurate Q-values according to $\zeta$. Using steady state distribution $\xi$ of some sampling policy $\pi$ implies that decisions at often visited states should be more reliable that those at seldom visited states. Take the optimal policy in a navigation task (like Section 5.3 or Section 5.4) as an example. Transition noise guarantees an ergodic Markov chain, but the steady state distribution will concentrate almost all mass around the goal. As a result, Q-values far away from the goal can be approximated almost arbitrarily bad and decisions become thus erratic. This is not the intended effect. To solve the task, one needs to control the approximation error of all states one will encounter until the task is complete. Without knowledge about certainty, every

decision along the way might be equally important. A uniform $\zeta$ reflects this. A similar argument can be made for a uniform $\tau$ in the context of policy iteration.

Future works might be able to identify the corresponding Radon-Nikodym derivatives $\frac{d\zeta}{d\xi} : \mathcal{Z} \to \mathbb{R}^+$ and $\frac{d\tau}{d\pi} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}^+$ and use *importance sampling* as discussed in Section 3.5. Until then the best approach appears to be sampling with a random policy and fine tuning with the modified SFA algorithm presented in this article.

### 6.2 Optimal Basis Functions

The analysis in Section 4.3 introduces the concept of *optimal basis functions* (Definition 12, Page 2088) to construct approximation spaces for LSTD with SFA. However, Theorem 13, Page 2089, implies that SFA features are only *optimal* for sampling policy $\pi$. This optimality is lost when policy iteration varies $\pi$, but optimizing the basis functions w.r.t. policy has no analytic solution and appears not feasible. Besides the question of feasibility, there are alternatives to the definition of *optimal basis functions* in Section 4.3. Here we suggest three possibilities:

1. Given one task $m = (\mathcal{X}, \mathcal{A}, P, R)$ only, the optimal basis functions for LSPI encode the true value function $v_m^\pi(\cdot)$ of $m$ with all possible policies $\pi \in \Omega$, that is

$$\inf_{\phi \in L^2(\mathcal{Z},\xi)} \mathbb{E}\left[ \left\| v_m^\pi - \hat{\Pi}_\xi^\phi [v_m^\pi] \right\|_\xi^2 \;\middle|\; \pi \sim \omega(\cdot) \right].$$

   As LSPI is based on a dictionary of transitions, however, the distribution $\xi$ of the weighted projection operator $\hat{\Pi}_\xi^\phi$ corresponds to the sampling distribution instead of steady state distribution of policy $\pi$.

2. Definition 12 minimizes the mean approximation error over all expected task. An alternative would be to minimize the *worst case* bound instead, that is

$$\inf_{\phi \in L^2(\mathcal{Z},\xi)} \left( \sup_{m \in \mathcal{M}, \pi \in \Omega} \left\| v_m^\pi - \hat{\Pi}_\xi^\phi [v_m^\pi] \right\|_\xi^2 \right).$$

3. The presented *weighted Euclidean norm projection* $\hat{\Pi}_\xi^\phi$ is the most commonly used choice. However, Guestrin et al. (2001) have proposed an efficient algorithm based on *supremum norm projection* $\hat{\Pi}_\infty^\phi$ for approximation of the updated value function $\hat{B}^\pi[v](\cdot)$. It is straightforward to derive a bound analogous[42] to Tsitsiklis and Van Roy (1997) and thus to define a matching *optimality criterion*

$$\inf_{\phi \in L^2(\mathcal{Z},\xi)} \left( \sup_{m \in \mathcal{M}, \pi \in \Omega} \left\| v_m^\pi - \hat{\Pi}_\infty^\phi [v_m^\pi] \right\|_\infty \right).$$

As with the criterion of Definition 12, it might not be feasible to solve these optimization problems in practice. Future works could find feasible approximations thereof, though.

42. One needs to show that $\hat{\Pi}_\infty^\phi[\cdot]$ is a non-expansion and $\hat{B}^\pi[\cdot]$ a contraction in $\|\cdot\|_\infty$ (Bertsekas, 2007).

### 6.3 Sparse Subset Selection

Results in Section 5.5 suggest that sparse subsets uniformly distributed in $\mathcal{X}$ (rather than $\mathcal{Z}$) support the best basis functions for LSPI. One possible explanation is the effect of the sparse subset distribution $\chi : \mathcal{B}(\mathcal{X}) \to [0,1]$ onto the objective. In limit of an infinite subset of an infinite Markov chain, extracted features $\phi_i(z) = \int \chi(ds)\, \alpha_i(s)\, \kappa(z,s) = \langle \alpha_i, \kappa(z,\cdot)\rangle_\chi, \forall z \in \mathcal{Z}$, are determined by coefficient functions $\alpha_i \in L^2(\mathcal{X},\chi)$. To determine these functions, RSK-SFA therefore approximates *generalized eigenfunctions* in $L^2(\mathcal{X},\chi)$, which are strongly affected by norm $\|\cdot\|_\chi$. Discrete SFA, on the other hand, represents every state by one unique variable, which corresponds to a uniform distribution $\chi$ in $\mathcal{X}$. It is easy to see that a uniform $\chi$ preserves the optimization problem best. Selecting a subset uniformly in $\mathcal{Z}$, however, does not generally yield a uniform $\chi$ in $\mathcal{X}$, due to the difference in Euclidean and diffusion distances. Results presented in Sections 5.3 and 5.6 demonstrate how sensitive the LSPI performance based on SFA features is to sampling policy and sampling state-distribution. Non-uniform $\chi$ will probably decrease performance similarly. It seem therefore reasonable to attribute the results of Section 5.5 to the above effect.

Nonetheless, this raises two question of practical concern:

1. how can we *select* sparse subsets uniformly in $\mathcal{X}$ and
2. how can we guarantee uniform *support*[43] in $\mathcal{Z}$?

The first question is for sparse kernelized RL algorithms (e.g., Engel et al., 2003; Xu, 2006) of utmost importance as the reported problem will most likely affect them as well. Future works must derive such an algorithm, maybe based on slowness or diffusion distances.

The related *radial basis function networks* (RBF, see, e.g., Haykin, 1998) have found an empirically answer to the second question: each support vector $s_i$ is assigned an individual kernel width $\sigma_i$ relative to the distance to its neighbors. For the Hilbert spaces of all kernels $\kappa(\cdot,\cdot)$ holds $\mathcal{H}_\kappa \subset L^2(\mathcal{Z},\xi)$ and one could thus perform inner products between Gaussian kernel functions of different width[44] in $L^2(\mathcal{Z},\xi)$. However, the math necessary to pose kernel SFA in this framework is quite advanced and calls for further research.

### 6.4 Visual Policies for Robots

Visual tasks are an interesting field of research as they expose elemental weaknesses in current RL approaches, for example the different Euclidean distances in observation and ideal approximation spaces discussed in this article. For applications in the field of robotics, however, the assumption of *isometry between observations and states* reaches its limits.

On the one hand, partial observability of the environment (POMDPs, Kaelbling et al., 1998) will jumble the observed transition structure and therefore all discussed feature construction methods.[45] This could be avoided by including partial *histories* of observations. For example, *predictive state representations* (PSR, Littman et al., 2001; Wingate, 2012) can construct sufficient statistics of

43. Areas in $\mathcal{Z}$ with less *support vectors* $s_i$ will have an overall lower output of kernel functions and will thus exhibit worse generalization. This effect can be quantized for sample $z \in \mathcal{Z}$ by the *approximation error* of the corresponding kernel function, that is, $\inf_{a \in \mathbb{R}^m} \|\kappa(\cdot,z) - \sum_i a_i \kappa(\cdot,s_i)\|_{\mathcal{H}}$, and is called the *support* of $z$.

44. Note that $\langle \kappa_a(\cdot,s_i), \kappa_b(\cdot,s_j)\rangle_\xi = \int \xi(dz)\, \kappa_a(z,s_i)\, \kappa_b(z,s_j)$ has an analytic solution if $\xi$ is the uniform distribution, because the product of two Gaussian functions is a Gaussian function as well.

45. Basis construction might work well in a POMDP if applied on *beliefs* instead of observations, though.

observation history to solve the task without extensive knowledge of the underlying POMDP. Extensions to continuous state and observation spaces are rare (Wingate and Singh, 2007) and linear approximation not straight forward. Additionally, any traditional metric over histories will probably not reflect *diffusion distances* very well and thus perform suboptimal in techniques like LSPI (see our conclusion in Section 5.6). Therefore, the potential of feature construction techniques like *optimal basis functions* (Section 4.3) for PSR appear tremendous and should be investigated further.

Setting partial observability aside, on the other hand, every natural variable influencing the image will be treated as part of state space $\mathcal{X}$, for example angle and brightness of illumination, non-stationary objects, the view out of the window, etc. The resulting state space $\mathcal{X}$ grows exponentially in the number of these independent variables, as every combination of variables is a unique state. Besides encoding mostly useless information, this yields two problems for the method presented in this article: (i) the whole state space $\mathcal{X}$ must be *sampled* by the RL agent, which will eventually take too much time, and (ii) the basis functions must *support* the whole space, for example, a subset for sparse kernel methods must uniformly cover $\mathcal{X}$ (see Sections 5.5 and 6.3), which will eventually require too many computational resources. Both problems can not be resolved by current kernel methods to construct basis functions and/or standard linear RL approaches. *Factored MDP* approaches in combination with *computer vision* methods have the potential to solve this dilemma, though.

Using an array of highly invariant image descriptors (e.g., SIFT, Lowe, 1999), object recognition and position estimates (e.g., SLAM, Smith et al., 1990; Davison, 2003), the observation space $\mathcal{Z}$ can become much more regular. Smart choices of descriptors, for example, reacting to the window frame and not the view outside, will even make them invariant to most state dimensions in $\mathcal{X}$. If the descriptors include short-term memory, then the presented method could even be applied in a POMDP setup. However, an application of standard kernel methods takes the similarity between all descriptors at once into account and would therefore still require sampling and support on the whole space $\mathcal{X}$. *Factorizing* basis functions $\phi_i(\cdot) = \prod_j \psi_{ij}(\cdot)$, on the other hand, have full domain $\mathcal{Z}$ but are a product of multiple functions $\psi_{ij}(\cdot)$ with a domain of only a few descriptors. Integrals over $\mathcal{Z}$ break down into the product of multiple low dimensional integrals, which would each only require limited amount of sampling and support. If those factorized basis functions approximate the *optimal basis functions* discussed in Section 4.3 sufficiently close, factored MDP algorithms can be applied (Koller and Parr, 1999; Guestrin et al., 2001; Hauskrecht and Kveton, 2003; Guestrin et al., 2004). Future works must develop both the factorizing basis function construction method and some adequate factored linear RL algorithm to exploit them.

## 7. Summary

This article investigates *approximation spaces* for value estimation, in particularly the role of the *metric* in these spaces. This is relevant because this metric influences the Euclidean $L_2$ approximation error minimized by *least-squares temporal difference learning* (LSTD). We hypothesize that an *ideal Euclidean metric for LSTD should encode diffusion distances*, which reflect similar futures analogous to values. Furthermore, *slow-feature analysis (SFA) constructs the best subspace-invariant approximation spaces for LSTD*. To verify these hypotheses we compare *Krylov bases*, *proto-value functions* (PVF), *principal component analysis* (PCA) and SFA (see Section 3) theoretically and experimentally. We also derive a novel *importance sampling* modification to the SFA

algorithm to compensate for sampling imbalances of SFA. The novel algorithm showed excellent performance in Section 5.

Our theoretical analysis in Section 4 compares Krylov bases with SFA. We argue that the latter is a generalization of PVF and can construct typical *subspace-invariant* approximation spaces. Our analysis yields impressive statements for MDPs which are actually subspace-invariant under PVF or SFA. For example, Corollary 9 (Page 2086) shows a dramatically improved bound on the LSTD approximation error and Theorem 11 (Page 2087) gives a lower bound on the improvement thereof by adding another feature. However, compatible MDPs are not very common: SFA features are subspace-invariant for all MDPs with a *self-adjoint* transition operator and PVF are for all MDPs with a transition kernel visiting all neighbors uniformly. Note that the latter set is a subset of former. We argue further that *real-valued* subspace-invariant features can only be obtained for MDPs with *self-adjoint* transition operators. Both SFA and PVF can thus be interpreted as self-adjoint approximations of arbitrary MDPs, as empirical results in Section 5.1 demonstrate. This interpretation is formally supported by Theorem 13 (Page 2089). It states that SFA minimizes a mean bound over all tasks in the *same environment*, which means an arbitrary but fixed transition kernel and all possible reward functions. However, all above results hold only for the sampling policy.

It is therefore an empirical question how the discussed approximation spaces will fare when *least-squares policy iteration* (LSPI) changes the policy. We ask in Section 5:

- *How well does LSTD estimate the value function of a given Markov chain?* We predicted in Section 4.4 and verified in Sections 5.1 to 5.3: "SFA can provide better approximation spaces for LSTD than PVF".
- *How good is the performance of policies learned by LSPI based on a random policy?* Our empirical conclusion of Sections 5.3 and 5.6 is "[Modified] SFA can construct better approximation spaces for LSPI than PVF".
- *How does this performance depend on the approximation space metric?* The connection between diffusion distance and approximation error suggested itself in Sections 4.1 and 4.2. We empirically verified in Section 5.6: "LSPI performance is facilitated by approximation spaces that encode diffusion distances of a uniform random policy" because "SFA essentially performs PCA based on diffusion distances".

We see both theoretical and empirical results as evidence supporting our hypotheses. There are still too many open questions to be certain, like the undesirable dependence on the sampling distribution and other issues discussed in Section 6. However, especially the good performance with LSPI inspires hope and calls for further research.

## Acknowledgments

We would like to thank Roland Vollgraf, Hannes Nickisch and Mathias Franzius for pointing us to SFA as a pre-processing method for video data. This work was funded by the *German science foundation* (DFG) within the SPP 1527 *autonomous learning*, the *German federal ministry of education and research* (grant 01GQ0850), the EPSRC grant #EP/H017402/1 (CARDyAL) and by the integrated graduate program on *human-centric communication* at Technische Universität Berlin.

## Appendix A. Proofs of Section 4

For an introduction into the terminology see Section 2. The equivalency sign $\equiv$ is used to indicate that two optimization problems are solved by the same function.

*Lemma 3 (repeated): Let $\xi$ denote the steady state distribution of ergodic transition kernel $P^\pi$, which has a self-adjoint transition operator $\hat{P}^\pi = (\hat{P}^\pi)^* : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$. The corresponding diffusion distance equals the Euclidean distance in the space spanned by $\psi_i^t(\cdot) := \lambda_i^t \phi_i(\cdot), \forall i \in \mathbb{N}$, where $\lambda_i \in \mathbb{R}$ and $\phi_i \in L^2(\mathcal{Z},\xi)$ are the eigenvalues and eigenfunctions of $\hat{P}^\pi$, that is*

$$d_t(x,y) = \|\psi^t(x) - \psi^t(y)\|_2\,, \quad \forall x,y \in \mathcal{Z}, \forall t \in \mathbb{N}\setminus\{0\}\,.$$

**Proof** The diffusion distance $d_t(x,y)$ between states $x$ and $y$ is defined as the mean squared difference of the probability distributions after $t$ steps (see Page 2084):

$$d_t(x,y) \quad := \quad \left\|\mu_x^t - \mu_y^t\right\|_\xi .$$

Under the formal restrictions mentioned of Assumption 1, Page 2083, $\mu_x^t \in L^2(\mathcal{Z},\xi), \forall t \in \mathbb{N}\setminus\{0\}$, and one can rewrite the inner product with arbitrary functions $f \in L^2(\mathcal{Z},\xi)$:

$$\langle \mu_x^t, f\rangle_\xi = \int \xi(dy)\left(\mu_x^t(y)\right) f(y) = \int (P^\pi)^t(dy|x)\, f(y) = (\hat{P}^\pi)^t[f](x)\,,$$

where $(\hat{P}^\pi)^t$ denotes $t$ successive applications of the transition operator $\hat{P}^\pi$ in $L^2(\mathcal{Z},\xi)$.

$$\begin{aligned} d_t^2(x,y) &= \langle \mu_x^t, \mu_x^t\rangle_\xi - 2\langle \mu_x^t, \mu_y^t\rangle_\xi + \langle \mu_y^t, \mu_y^t\rangle_\xi \\ \langle \mu_x^t, \mu_y^t\rangle_\xi &= \int (P^\pi)^t(dz|x)\,\mu_y^t(z) \quad = \quad (\hat{P}^\pi)^t\left[\mu_y^t\right](x)\,. \end{aligned}$$

$\hat{P}^\pi$ is specified to be self-adjoint and due to the *Hilbert-Schmidt theorem* (e.g., Theorem 4.2.23 in Davies, 2007) holds for eigenfunctions $\hat{P}^\pi[\phi_i](\cdot) = \lambda_i \phi_i(\cdot)$, and $\langle \phi_i, \phi_j\rangle_\xi = \delta_{ij}, \forall i,j \in \mathbb{N}$:

$$\hat{P}^\pi[f](x) = \sum_{i=0}^{\infty} \langle f, \phi_i\rangle_\xi\, \lambda_i\, \phi_i(x)\,, \qquad \forall x \in \mathcal{Z}, \quad \forall f \in L^2(\mathcal{Z},\xi)\,.$$

Applying this $t$ times, we can write

$$\begin{aligned} (\hat{P}^\pi)^t[\mu_y^t](x) &= \sum_{i=0}^{\infty} \phi_i(x)\,\lambda_i^t\,\langle \mu_y^t, \phi_i\rangle_\xi \;=\; \sum_{i=0}^{\infty} \phi_i(x)\,\lambda_i^t\,(\hat{P}^\pi)^t[\phi_i](y) \\ &= \sum_{i,j=0}^{\infty} \phi_i(x)\,\lambda_i^t\,\lambda_j^t\,\phi_j(y)\,\underbrace{\langle \phi_i, \phi_j\rangle_\xi}_{\delta_{ij}} \;=\; \psi^t(x)^\top \psi^t(y)\,. \end{aligned}$$

Therefore the diffusion distance $d_t(x,y)$ can be written as

$$d_t^2(x,y) = \psi^t(x)^\top\psi^t(x) - 2\psi^t(x)^\top\psi^t(y) + \psi^t(y)^\top\psi^t(y) = \|\psi^t(x) - \psi^t(y)\|_2^2\,.$$

■

*Lemma 4: Let $P^{\pi}$ be an ergodic transition kernel in $\mathcal{Z}$ with steady state distribution $\xi$. The kernel induced by adjoint transition operator $(\hat{P}^{\pi})^*$ in $L^2(\mathcal{Z},\xi)$ is $\xi$-almost-everywhere an ergodic transition kernel with steady state distribution $\xi$.*

**Proof** We first show that for *any* linear operator $\hat{A} : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$ with *kernel* $A : \mathcal{Z} \times \mathcal{B}(\mathcal{Z}) \to \mathbb{R}^+$, that is, $\hat{A}[f](x) = \int A(dy|x)\, f(y)$, *$\xi$-almost-everywhere* ($\xi$-a.e.) in $\mathcal{Z}$ holds

$$\underbrace{\langle f, \hat{A}[\mathbf{1}]\rangle_\xi = \langle f, \mathbf{1}\rangle_\xi, \quad \forall f \in L^2(\mathcal{Z},\xi)}_{(i)} \qquad \Leftrightarrow \qquad \int \underbrace{A(dy|x) = 1, \quad \xi\text{-a.e.}}_{(ii)}$$

where $\mathbf{1}(x) = 1, \forall x \in \mathcal{Z}$, is the constant function in $L^2(\mathcal{Z},\xi)$.

Let's assume the induction $(i) \Rightarrow (ii)$ is *not* true, that is, $(i)$ is true, but there exists a Borel set with non-zero measure $\xi$ of states $x \in \mathcal{Z}$ that violate $(ii)$. This set can be split up in $\int A(dy|x) > 1, \forall x \in B_+ \in \mathcal{B}(\mathcal{Z})$, and $\int A(dy|x) < 1, \forall x \in B_- \in \mathcal{B}(\mathcal{Z})$, with $\xi(B_+ \cup B_-) > 0$. Let furthermore $f \in L^2(\mathcal{Z},\xi)$ be defined as

$$f(x) = \begin{cases} 1 & ,\text{if } x \in B_+ \\ -1 & ,\text{if } x \in B_- \\ 0 & ,\text{otherwise} \end{cases},$$

which must adhere to claim $(i)$:

$$\begin{array}{rcl} \langle f, \hat{A}[\mathbf{1}]\rangle_\xi & = & \int_{B_+} \xi(dx) \int A(dy|x) - \int_{B_-} \xi(dx) \int A(dy|x) \\ & > & \int_{B_+} \xi(dx) - \int_{B_-} \xi(dx) \quad = \quad \langle f, \mathbf{1}\rangle_\xi . \end{array}$$

This is a contradiction and proves $(i) \Rightarrow (ii)$. The induction $(i) \Leftarrow (ii)$ is trivial, which proves $(i) \Leftrightarrow (ii)$.

Now we show that $(i)$ holds for $(\hat{P}^{\pi})^*$, which is the adjoint operator to $\hat{P}^{\pi}$.

$$\langle f, (\hat{P}^{\pi})^*[\mathbf{1}]\rangle_\xi = \langle \hat{P}^{\pi}[f], \mathbf{1}\rangle_\xi = \int \underbrace{\int \xi(dx)\, P^{\pi}(dy|x)}_{\xi(dy)\text{ (ergodicity)}} f(y) = \langle f, \mathbf{1}\rangle_\xi, \quad \forall f \in L^2(\mathcal{Z},\xi).$$

This proves that the kernel of $(\hat{P}^{\pi})^*$ is a *transition kernel* $\xi$-almost-everywhere in $\mathcal{Z}$, which means the kernel adheres to $(ii)$. Ergodicity can be proven using the same techniques as above:

$$\langle \mathbf{1}, \hat{A}[f]\rangle_\xi = \langle \mathbf{1}, f\rangle_\xi, \quad \forall f \in L^2(\mathcal{X},\xi) \qquad \Leftrightarrow \qquad \xi(B) = \int A(B|x)\, \xi(dx), \quad \forall B \in \mathcal{B}(\mathcal{X}).$$

$\langle \mathbf{1}, (\hat{P}^{\pi})^*[f]\rangle_\xi = \langle \mathbf{1}, f\rangle_\xi, \forall f \in L^2(\mathcal{X},\xi)$, and therefore the transition kernel of $(\hat{P}^{\pi})^*$ is ergodic with steady state distribution $\xi$. ■

*Lemma 5: In the limit of an infinite ergodic Markov chain drawn by transition kernel $P^{\pi}$ in $\mathcal{Z}$ with steady state distribution $\xi$ holds $\mathcal{S}(f) = 2\langle f, (\hat{I} - \hat{P}^{\pi})[f]\rangle_\xi, \forall f \in L^2(\mathcal{Z},\xi)$.*

**Proof** Due to a theorem of Jensen and Rahbek (2007) we can ensure that the *empirical mean* of functions over *sequences* of states converges in the limit to its expectation:

$$\begin{array}{rcl} \mathcal{S}(f) & = & \lim_{n\to\infty} \frac{1}{n} \sum_{t=0}^{n-1} \Big(f(x_{t+1}) - f(x_t)\Big)^2 = \iint \Big(f(y) - f(x)\Big)^2 P^{\pi}(dy|x)\, \xi(dx) \\ & = & \langle f, f\rangle_\xi - 2\langle f, \hat{P}^{\pi}[f]\rangle_\xi + \int f^2(y) \underbrace{\int P^{\pi}(dy|x)\, \xi(dx)}_{\xi(dy)\text{ due to ergodicity}} = 2\langle f, (\hat{I} - \hat{P}^{\pi})[f]\rangle_\xi . \end{array}$$

■

*Theorem 7: On average over all reward functions $r^\pi : \mathcal{Z} \to \mathbb{R}$ drawn from a white noise functional $\rho$, the squared norm of a Krylov basis $\{\phi_i^K\}_{i=1}^p$ from an ergodic transition kernel $P^\pi$ encodes squared diffusion distances based on $\hat{P}^\pi$ up to horizon $p-1$, that is*

$$d_t^2(x,y) \;=\; \mathbb{E}\left[\left\|\phi^K(x)-\phi^K(y)\right\|_{\varrho}^2 \middle| r^\pi \sim \rho\right], \quad \forall x,y \in \mathcal{Z},\, \exists \varrho \in (\mathbb{R}^+)^p,\, \forall t \in \{1,\dots,p-1\}.$$

**Proof** Let $\xi$ denote the steady state distribution of transition kernel $P^\pi$. Given a reward function $r^\pi \in L^2(\mathcal{Z},\xi)$, a Krylov feature can be posed in terms of functions $\mu_x^t \in L^2(\mathcal{Z},\xi)$ (see Definition 1, Page 2083, and the Proof of Lemma 3), that is

$$\phi_i^K(x) \;:=\; (\hat{P}^\pi)^{i-1}[r^\pi](x) \;=\; \int (P^\pi)^{i-1}(dy|x)\, r^\pi(y) \;=\; \langle \mu_x^{i-1}, r^\pi\rangle_\xi, \qquad \forall x \in \mathcal{Z}.$$

One can use a property of white noise functionals (Footnote 25, p. 2085) to prove the theorem.

$$\begin{aligned}\mathbb{E}\left[\left\|\phi^K(x)-\phi^K(y)\right\|_{\varrho}^2\right] &= \sum_{i=1}^p \varrho_i\, \mathbb{E}\left[\left(\langle \mu_x^{i-1}-\mu_y^{i-1}, r^\pi\rangle_\xi\right)^2\right] \\ &= \sum_{i=1}^p \varrho_i \left\|\mu_x^{i-1}-\mu_y^{i-1}\right\|_\xi^2 \;=\; \sum_{i=1}^p \varrho_i\, d_{i-1}^2(x,y).\end{aligned}$$

$\varrho \in (\mathbb{R}^+)^p$ can be chosen freely; all diffusion distances with $t < p$ are therefore encoded. ■

*Lemma 8: Let $\{\phi_i\}_{i=1}^p$ denote any $p$ SFA features from a MDP with self-adjoint transition operator, then the LSTD fixed point $f^\pi = \hat{\Pi}_\xi^\phi\left[\hat{B}^\pi[f^\pi]\right]$ and the projection of true value function $v^\pi = \hat{B}^\pi[v^\pi]$ coincide, that is*

$$f^\pi(x) \;=\; \hat{\Pi}_\xi^\phi[v^\pi](x) \;=\; \sum_{i=1}^p \langle r^\pi, \phi_i\rangle_\xi\, \tau_i\, \phi_i(x), \quad \forall x \in \mathcal{Z}, \quad \tau_i := \left(1-\gamma+\tfrac{\gamma}{2}\mathcal{S}(\phi_i)\right)^{-1}.$$

**Proof** Let $\psi_i \in L^2(\mathcal{Z},\xi)$ denote the (due to the Hilbert-Schmidt theorem orthonormal) eigenfunctions of $\hat{P}^\pi$ and $\lambda_i$ the corresponding eigenvalues, that is, $\hat{P}^\pi[\psi_i] = \lambda_i \psi_i$. $\{\psi_i\}_{i=1}^\infty$ is a full basis of $L^2(\mathcal{Z},\xi)$ and thus $r^\pi = \sum_{i=1}^\infty \langle r^\pi, \psi_i\rangle_\xi \psi_i$ with $\phi_i = \psi_i, \forall i \le p$ and $\langle \phi_i, \psi_j\rangle_\xi = \delta_{ij}$. From this we can conclude $\hat{\Pi}_\xi^\phi[\psi_i] = \phi_i, \forall i \le p$, and $\hat{\Pi}_\xi^\phi[\psi_i] = 0, \forall i > p$. Due to the geometric series and Lemma 5 also holds $\sum_{t=0}^\infty \gamma^t \lambda_i^t = (1-\gamma\lambda_i)^{-1} = \tau_i$. Therefore,

$$\begin{aligned}\hat{\Pi}_\xi^\phi[v^\pi] &= \sum_{t=0}^\infty \gamma^t\, \hat{\Pi}_\xi^\phi\left[(\hat{P}^\pi)^t[r^\pi]\right] \;=\; \sum_{i=1}^\infty \langle r^\pi, \psi_i\rangle_\xi \sum_{t=0}^\infty \gamma^t\, \hat{\Pi}_\xi^\phi\left[(\hat{P}^\pi)^t[\psi_i]\right] \\ &= \sum_{i=1}^p \langle r^\pi, \phi_i\rangle_\xi\, \phi_i \sum_{t=0}^\infty \gamma^t \lambda_i^t \;=\; \sum_{i=1}^p \langle r^\pi, \phi_i\rangle_\xi\, \tau_i\, \phi_i, \\ f^\pi &= \hat{\Pi}_\xi^\phi\left[\hat{B}^\pi[f^\pi]\right] \;=\; \sum_{t=0}^\infty \gamma^t \left(\hat{\Pi}_\xi^\phi[\hat{P}^\pi]\right)^t \left[\hat{\Pi}_\xi^\phi[r^\pi]\right] \;=\; \sum_{i=1}^p \langle r^\pi, \phi_i\rangle_\xi\, \phi_i \sum_{t=0}^\infty \gamma^t \lambda_i^t.\end{aligned}$$

■

*Theorem 11: Let $\xi$ be the steady state distribution on $\mathcal{Z}$ of a MDP with policy $\pi$ and a self-adjoint transition operator in $L^2(\mathcal{Z},\xi)$. Let further $\Phi_p = \{\phi_i\}_{i=1}^p$ be any set of p SFA features and $v^\pi \in L^2(\mathcal{Z},\xi)$ the true value of the above MDP. The improvement of the LSTD solution $f^{(p)} := \hat{\Pi}_\xi^{\Phi_p}\left[\hat{B}^\pi[f^{(p)}]\right]$ by including the p'th feature is bounded from below by*

$$\left\|v^\pi - f^{(p-1)}\right\|_\xi - \left\|v^\pi - f^{(p)}\right\|_\xi \quad \geq \quad \tfrac{1-\gamma}{2}\,\frac{\langle r^\pi,\phi_p\rangle_\xi^2}{\|r^\pi\|_\xi}\,\tau_p^2\,, \qquad \tau_p := \left(1-\gamma+\tfrac{\gamma}{2}\mathcal{S}(\phi_p)\right)^{-1}.$$

**Proof** Let $\{\phi_i\}_{i=1}^\infty$ denote the extension of $\Phi_p$ to a full orthonormal basis of $L^2(\mathcal{Z},\xi)$, that is, $f(\cdot) = \sum_{i=1}^\infty \langle f,\phi_i\rangle_\xi\,\phi_i(\cdot)$, $\forall f \in L^2(\mathcal{Z},\xi)$. Lemma 8 shows that $f^{(p)} = \hat{\Pi}_\xi^{\Phi_p}[v^\pi], \forall p \in \mathbb{N}$,

$$\left\|v^\pi - \hat{\Pi}_\xi^{\Phi_{p-1}}[v^\pi]\right\|_\xi^2 - \left\|v^\pi - \hat{\Pi}_\xi^{\Phi_p}[v^\pi]\right\|_\xi^2 = \left\|\sum_{i=p}^\infty \langle v^\pi,\phi_i\rangle_\xi\phi_i\right\|_\xi^2 - \left\|\sum_{i=p+1}^\infty \langle v^\pi,\phi_i\rangle_\xi\phi_i\right\|_\xi^2 = \langle v^\pi,\phi_p\rangle_\xi^2.$$

Note further that $\langle v^\pi,\phi_p\rangle_\xi^2 = \langle r^\pi,\phi_p\rangle_\xi^2\,\tau_p^2$, and that one can bound the norm of $v^\pi$:

$$\left\|v^\pi\right\|_\xi \;=\; \left\|\sum_{t=0}^\infty \gamma^t\,(\hat{P}^\pi)^t[r^\pi]\right\|_\xi \;\leq\; \sum_{t=0}^\infty \gamma^t\,\left\|(\hat{P}^\pi)^t[r^\pi]\right\|_\xi \;\leq\; \sum_{t=0}^\infty \gamma^t\,\left\|r^\pi\right\|_\xi \;=\; \frac{1}{1-\gamma}\,\left\|r^\pi\right\|_\xi.$$

The first equality follows from the proof of Lemma 14, the first inequality from the property of norms, the last inequality from Lemma 15 and the last equality from the geometric series. Using the identity $a^2-b^2=(a-b)(a+b)$ and inequality $\|v^\pi - \hat{\Pi}_\xi^{\Phi_p}[v^\pi]\|_\xi \leq \|v^\pi\|_\xi$,

$$\left\|v^\pi - f\right\|_\xi - \left\|v^\pi - f'\right\|_\xi \;=\; \frac{\left\|v^\pi - f\right\|_\xi^2 - \left\|v^\pi - f'\right\|_\xi^2}{\left\|v^\pi - f\right\|_\xi + \left\|v^\pi - f'\right\|_\xi} \;\geq\; \frac{\langle v^\pi,\phi_p\rangle_\xi^2}{2\,\|v^\pi\|_\xi} \;\geq\; \tfrac{1-\gamma}{2}\,\frac{\langle r^\pi,\phi_p\rangle_\xi^2}{\|r^\pi\|_\xi}\,\tau_p^2\,,$$

where $f := f^{(p-1)}$ and $f' := f^{(p)}$. ■

*Theorem 13: For any infinite ergodic Markov chain with steady state distribution $\xi$ over state space $\mathcal{Z}$, SFA selects features from function set $\mathcal{F} \subset L^2(\mathcal{Z},\xi)$ that minimize an upper bound on the optimality criterion of Definition 12 for sampling policy $\pi$ and discount factor $\gamma > 0$, under the assumption that the mean-reward functions $r^\pi : \mathcal{Z} \to \mathbb{R}$ are drawn from a white noise functional in $L^2(\mathcal{Z},\xi)$.*

**Proof** Lemma 14 shows that for all $0 \leq \gamma < 1$ the operator $(\hat{I} - \gamma\hat{P}^\pi) : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$ is invertible. Let $\hat{\Theta}^\pi$ denote this inverse operator. For any mean reward function $r^\pi \in L^2(\mathcal{Z},\xi)$ the corresponding true value function $v_r^\pi \in L^2(\mathcal{Z},\xi)$ can be determined analytically: $v_r^\pi(x) = r^\pi(x) + \gamma\hat{P}^\pi[v_r^\pi](x) = \hat{\Theta}^\pi[r^\pi](x), \forall x \in \mathcal{Z}$. According to the assumptions, the mean reward functions $r^\pi \sim \rho$ are distributed as a white noise functional, which implies

$$\int \langle f, r^\pi\rangle_\xi^2\,\rho(dr^\pi) \;=\; \langle f,f\rangle_\xi\,, \quad \forall f \in L^2(\mathcal{Z},\xi)\,.$$

We will now show that the SFA objective minimizes an upper bound on Definition 12 and thus also minimizes the bound of Tsitsiklis and Van Roy (1997).

$$
\begin{aligned}
& \inf_{\phi\in(\mathcal{F})^p} \mathbb{E}\left[\left\|v_r^\pi - \hat{\Pi}_\xi^\phi[v_r^\pi]\right\|_\xi^2 \,\middle|\, r^\pi \sim \rho\right] \\
\equiv\ & \sup_{\phi\in(\mathcal{F})^p} \mathbb{E}\left[\langle v_r^\pi, \hat{\Pi}_\xi^\phi[v_r^\pi]\rangle_\xi \,\middle|\, r^\pi \sim \rho\right] \\
\equiv\ & \sup_{\phi\in(\mathcal{F})^p} \sum_{i=1}^p \mathbb{E}\left[\langle \phi_i, \hat{\Theta}^\pi[r^\pi]\rangle_\xi^2 \,\middle|\, r^\pi \sim \rho\right] && \text{s.t. } \langle \phi_i, \phi_j\rangle_\xi = \delta_{ij}\,,\ \forall i,j && (C_{ij}{:=}\delta_{ij}) \\
\equiv\ & \sup_{\phi\in(\mathcal{F})^p} \sum_{i=1}^p \left\|(\hat{\Theta}^\pi)^*[\phi_i]\right\|_\xi^2 && \text{s.t. } \langle \phi_i, \phi_j\rangle_\xi = \delta_{ij}\,,\ \forall i,j && \text{(assumption)} \\
\equiv\ & \inf_{\phi\in(\mathcal{F})^p} \sum_{i=1}^p \left\|\left(\hat{I} - \gamma(\hat{P}^\pi)^*\right)[\phi_i]\right\|_\xi^2 && \text{s.t. } \langle \phi_i, \phi_j\rangle_\xi = \delta_{ij}\,,\ \forall i,j && \text{(lemma 16)} \\
\leq\ & \inf_{\phi\in(\mathcal{F})^p} \sum_{i=1}^p \left((1+\gamma^2)\underbrace{\langle \phi_i, \phi_i\rangle_\xi}_{1} - 2\gamma\langle \phi_i, \hat{P}^\pi[\phi_i]\rangle_\xi\right) && \text{s.t. } \langle \phi_i, \phi_j\rangle_\xi = \delta_{ij}\,,\ \forall i,j && \text{(lemmas 15\&4)} \\
\equiv\ & \inf_{\phi\in(\mathcal{F})^p} -2\gamma \sum_{i=1}^p \langle \phi_i, \hat{P}^\pi[\phi_i]\rangle_\xi && \text{s.t. } \langle \phi_i, \phi_j\rangle_\xi = \delta_{ij}\,,\ \forall i,j \\
\overset{(*)}{\equiv}\ & \inf_{\phi\in(\mathcal{F})^p} 2\sum_{i=1}^p \langle \phi_i, (\hat{I} - \hat{P}^\pi)[\phi_i]\rangle_\xi && \text{s.t. } \begin{array}{rcll} \langle \phi_i, \phi_j\rangle_\xi &=& \delta_{ij}\,, & \forall i,j \\ \langle \phi_i, 1\rangle_\xi &=& 0\,, & \forall i \end{array} && (\gamma{>}0) \\
\equiv\ & \inf_{\phi\in(\mathcal{F})^p} \sum_{i=1}^p \mathcal{S}(\phi_i) && \text{s.t. } \begin{array}{rcll} \langle \phi_i, \phi_j\rangle_\xi &=& \delta_{ij}\,, & \forall i,j \\ \langle \phi_i, 1\rangle_\xi &=& 0\,, & \forall i \end{array} && \text{(Lemma 5).}
\end{aligned}
$$

The equivalency marked $(*)$ holds because the infimum is the same for all $\gamma > 0$. In the limit $\gamma \to 1$, however, $(\hat{I} - \gamma\hat{P}^\pi)$ is not invertible. The fist (constant) right eigenfunction of $\hat{P}^\pi$ must thus be excluded by the zero mean constraint. The last equation is the SFA optimization problem, which therefore minimizes an upper bound on the optimality criterion of Definition 12. ■

**Lemma 14** *For an ergodic transition operator $\hat{P}^\pi$ in $L^2(\mathcal{Z},\xi)$ with steady state distribution $\xi$ and $0 \leq \gamma < 1$, the operator $(\hat{I} - \gamma\hat{P}^\pi)$ is invertible. Let $\hat{\Theta}^\pi$ denote the inverse,*

$$\left\|(\hat{I} - \gamma\hat{P}^\pi)\left[\hat{\Theta}^\pi[f]\right] - f\right\|_\xi = 0\,, \quad \forall f \in L^2(\mathcal{Z},\xi)\,.$$

**Proof** Let $(\hat{P}^\pi)^t$ denote the composition of $t$ operators $\hat{P}^\pi$ with $(\hat{P}^\pi)^0 = \hat{I}$ and let $\hat{\Theta}^\pi := \lim_{n\to\infty} \sum_{t=0}^{n-1} \gamma^t (\hat{P}^\pi)^t$, then $\forall f \in L^2(\mathcal{Z},\xi)$:

$$
\begin{aligned}
\left\|\left(\hat{I} - \gamma\hat{P}^\pi\right)\left[\hat{\Theta}^\pi[f]\right] - f\right\|_\xi &= \lim_{n\to\infty}\left\|\left(\hat{I} - \gamma\hat{P}^\pi\right)\left[\sum_{t=0}^{n-1}\gamma^t(\hat{P}^\pi)^t[f]\right] - f\right\|_\xi \\
= \lim_{n\to\infty}\left\|\left(\hat{I} - \gamma^n(\hat{P}^\pi)^n\right)[f] - f\right\|_\xi = \lim_{n\to\infty}\gamma^n\left\|(\hat{P}^\pi)^n[f]\right\|_\xi &\leq \lim_{n\to\infty}\gamma^n\|f\|_\xi = 0\,.
\end{aligned}
$$

The inequality holds because Lemma 15 shows that $\hat{P}^\pi$ is a non-expansion and the last equality because all $f \in L^2(\mathcal{Z},\xi)$ are bounded from above, that is, $\|f\|_\xi < \infty$. ■

**Lemma 15** *An ergodic transition operator $\hat{P}^{\pi}$ in $L^2(\mathcal{Z},\xi)$ with steady state distribution $\xi$ is a non-expansion, defined as*

$$\left\|\hat{P}^{\pi}[f]\right\|_{\xi} \leq \left\|f\right\|_{\xi}, \qquad \forall f \in L^2(\mathcal{Z},\xi).$$

**Proof** Due to Jensens inequality[46] we have

$$\left\|\hat{P}^{\pi}[f]\right\|_{\xi}^2 = \langle \hat{P}^{\pi}[f], \hat{P}^{\pi}[f]\rangle_{\xi} = \int \xi(dx)\Big(\int P^{\pi}(dy|x)f(y)\Big)^2$$
$$\leq \int \underbrace{\int \xi(dx)P^{\pi}(dy|x)}_{\xi(dy) \text{ due to ergodicity}} f^2(y) = \langle f,f\rangle_{\xi} = \left\|f\right\|_{\xi}^2, \quad \forall f \in L^2(\mathcal{Z},\xi).$$

■

**Lemma 16** *For any invertible linear operator $\hat{A} : L^2(\mathcal{Z},\xi) \to L^2(\mathcal{Z},\xi)$ holds*

$$\sup_{\substack{f\in L^2(\mathcal{Z},\xi),\\ \langle f,f\rangle_{\xi}=1}} \left\|\hat{A}[f]\right\|_{\xi} \equiv \inf_{\substack{f\in L^2(\mathcal{Z},\xi),\\ \langle f,f\rangle_{\xi}=1}} \left\|\hat{A}^{-1}[f]\right\|_{\xi}.$$

**Proof** The *operator norm* of an operator $\hat{A}$ in $L^2(\mathcal{Z},\xi)$ is defined as

$$\|\hat{A}\|_{\xi} := \sup_{f\in L^2(\mathcal{Z},\xi)} \left\{ \frac{\|\hat{A}[f]\|_{\xi}}{\|f\|_{\xi}} \,\middle|\, f \neq 0 \right\} = \sup_{f\in L^2(\mathcal{Z},\xi)} \left\{ \|\hat{A}[f]\|_{\xi} \,\middle|\, \|f\|_{\xi} = 1 \right\}.$$

Using the one-to-one transformation $f \leftarrow \hat{A}^{-1}[f]$ in the equivalency marked $(*)$,

$$\sup_{\substack{f\in L^2(\mathcal{Z},\xi),\\ \langle f,f\rangle_{\xi}=1}} \left\|\hat{A}[f]\right\|_{\xi} \equiv \|\hat{A}\|_{\xi} \equiv \sup_{f\in L^2(\mathcal{Z},\xi)} \left\{ \tfrac{\|\hat{A}[f]\|_{\xi}}{\|f\|_{\xi}} \,\middle|\, f \neq 0 \right\} \equiv \inf_{f\in L^2(\mathcal{Z},\xi)} \left\{ \tfrac{\|f\|_{\xi}}{\|\hat{A}[f]\|_{\xi}} \,\middle|\, f \neq 0 \right\}$$
$$\overset{(*)}{\equiv} \inf_{f\in L^2(\mathcal{Z},\xi)} \left\{ \tfrac{\|\hat{A}^{-1}[f]\|_{\xi}}{\|f\|_{\xi}} \,\middle|\, f \neq 0 \right\} \equiv \inf_{\substack{f\in L^2(\mathcal{Z},\xi),\\ \langle f,f\rangle_{\xi}=1}} \left\|\hat{A}^{-1}[f]\right\|_{\xi}.$$

■

## References

P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in Neural Information Processing Systems*, pages 1–8, 2007.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

46. If $P^{\pi}$ is a transition kernel, Jensens inequality (e.g., Boyd and Vandenberghe, 2004) allows $\left(\int P^{\pi}(dy|x)f(y)\right)^2 \leq \int P^{\pi}(dy|x)f^2(y), \forall x \in \mathcal{Z}, \forall f \in L^2(\mathcal{Z},\xi)$.

P. Berkes and L. Wiskott. Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5:579–602, 2005.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, 3rd edition, 2007.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

W. Böhmer, S. Grünewälder, H. Nickisch, and K. Obermayer. Generating feature spaces for linear algorithms with regularized sparse kernel slow feature analysis. *Machine Learning*, 89(1-2): 67–86, 2012.

M. Bowling, A. Ghodsi, and D. Wilkinson. Action respecting embedding. In *International Conference on Machine learning*, 2005.

J. A. Boyan and A. W. Moore. Generalization in reinforcement learning: safely approximating the value function. In *Advances in Neural Information Processing Systems*, pages 369–376, 1995.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. J. Bradtke and A. G. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1/2/3):33–57, 1996.

R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Nadler, F. Warner, and S. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data. Part I: diffusion maps. *Proceedings of the National Academy of Science*, 102(21):7426 – 7431, May 2005.

L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

E. Brian Davies. *Linear Operators and their Spectra*. Cambridge University Press, 2007.

A. J. Davison. Real-time simultaneous localization and mapping with a single camera. In *IEEE International Conference on Computer Vision*, volume 2, page 1403, 2003.

D.P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

Y. Engel, S. Mannor, and R. Meir. Bayes meets Bellman: the Gaussian process approach to temporal difference learning. In *International Conference on Machine Learning*, pages 154–161, 2003.

K. Ferguson and S. Mahadevan. Proto-transfer learning in Markov decision processes using spectral methods. In *ICML Workshop on Transfer Learning*, 2006.

E. Ferrante, A. Lazaric, and M. Restelli. Transfer of task representation in reinforcement learning using policy-based proto-value functions. In *International Joint Conference on Autonomous Agents and Multiagent Systems*, 2008.

P. Földiák. Learning invariance from transformation sequences. *Neural Computation*, 3(2):194–200, 1991.

M. Franzius, H. Sprekeler, and L. Wiskott. Slowness and sparseness leads to place, head-direction, and spatial-view cells. *PLoS Computational Biology*, 3(8):e166, 2007.

S. Grünewälder and K. Obermayer. The optimal unbiased value estimator and its relation to LSTD, TD and MC. *Machine Learning*, 83:289–330, 2011.

C. Guestrin, D. Koller, and R. Parr. Max-norm projections for factored MDPs. In *International Joint Conference on Artificial Intelligence*, pages 673–682, 2001.

C. Guestrin, M. Hauskrecht, and B. Kveton. Solving factored MDPs with continuous and discrete variables. In *Uncertainty in Artificial Intelligence*, pages 235–242, 2004.

M. Hauskrecht and B. Kveton. Linear program approximations for factored continuous-state Markov decision processes. In *Advances in Neural Information Processing Systems*, pages 895–902, 2003.

S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2nd edition, 1998. ISBN 978-0132733502.

G. E. Hinton and S. Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

H. Holden, B. Øksendal, J. Ubøe, and T. Zhang. *Stochastic Partial Differential Equations*. Springer Science+Business Media, 2nd edition, 2010.

O. C. Jenkins and M. J. Mataric. A spatio-temporal extension to Isomap nonlinear dimension reduction. In *International Conference on Machine Learning*, 2004.

S. T. Jensen and A. Rahbek. On the law of large numbers for (geometrically) ergodic Markov chains. *Economic Theory*, 23:761–766, 2007.

S. Jodogne and J. H. Piater. Closed-loop learning of visual control policies. *Journal of Artificial Intelligence Research*, 28:349–391, 2007.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

J. Kober and J. Peters. Policy search for motor primitives in robotics. In *Advances in Neural Information Processing Systems*, 2009.

D. Koller and R. Parr. Computing factored value functions for policies in structured MDPs. In *International Joint Conference on Artificial Intelligence*, pages 1332–1339, 1999.

D. Koller and R. Parr. Policy iteration for factored MDPs. In *Uncertainty in Artificial Intelligence*, pages 326–334, 2000.

V. R. Kompella, M. D. Luciw, and J. Schmidhuber. Incremental slow feature analysis: adaptive low-complexity slow feature updating from high-dimensional input streams. *Neural Computation*, 24 (11):2994–3024, 2012.

G. D. Konidaris, S. Osentoski, and P.S. Thomas. Value function approximation in reinforcement learning using the Fourier basis. In *Proceedings of the Twenty-Fifth Conference on Artificial Intelligence*, 2011.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

S. Lange and M. Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *International Joint Conference on Neural Networks*, pages 1–8, 2010.

R. Legenstein, N. Wilbert, and L. Wiskott. Reinforcement learning on slow features of high-dimensional input streams. *PLoS Computational Biology*, 6(8):e1000894, 2010.

M. L. Littman, R. S. Sutton, and S. Singh. Predictive representations of state. In *In Advances In Neural Information Processing Systems 14*, 2001.

D. G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, 1999.

M. Luciw and J. Schmidhuber. Low complexity proto-value function learning from sensory observations with incremental slow feature analysis. In *International Conference on Artificial Neural Networks and Machine Learning*, volume III, pages 279–287. Springer-Verlag, 2012.

S. Mahadevan and B. Liu. Basis construction from power series expansions of value functions. In *Advances in Neutral Information Processing Systems*, pages 1540–1548, 2010.

S. Mahadevan and M. Maggioni. Proto-value functions: a Laplacian framework for learning representations and control in Markov decision processes. *Journal of Machine Learning Research*, 8: 2169–2231, 2007.

S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions On Signal Processing*, 41:3397–3415, 1993.

N. Mehta, S. Natarajan, P. Tadepalli, and A. Fern. Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning*, 73:289–312, 2008.

R. Parr, C. Painter-Wakefield, L. Li, and M. Littman. Analyzing feature generation for value-function approximation. In *International Conference on Machine Learning*, 2007.

R. Parr, L. Li, G. Taylor, C. Painter-Wakefiled, and M. L. Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *International Conference on Machine Learning*, 2008.

K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559 – 572, 1901.

M. Petrik. An analysis of Laplacian methods for value function approximation in MDPs. In *International Joint Conference on Artificial Intelligence*, pages 2574–2579, 2007.

M. Petrik and S. Zilberstein. Robust approximate bilinear programming for value function approximation. *Journal of Machine Learning Research*, 12:3027–3063, 2011.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

M. Reed and B. Simon. *Methods of Modern Mathematical Physics I: Functional Analysis*. Academic Press, 1980. ISBN 0-12-585050-6.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002. ISBN 978-0262194754.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

R. Smith, M. Slef, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous Robot Vehicles*. Springer-Verlag, 1990.

A.J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings to the 17th International Conference Machine Learning*, pages 911–918, 2000.

M. Snel and S. Whiteson. Multi-task reinforcement learning: Shaping and feature selection. In *European Workshop on Reinforcement Learning*, pages 237–248, 2011.

N. Sprague. Predictive projections. In *International Joint Conference on Artificial Intelligence*, pages 1223–1229, 2009.

H. Sprekeler. On the relationship of slow feature analysis and Laplacian eigenmaps. *Neural Computation*, 23(12):3287–3302, 2011.

Y. Sun, F. Gomez, M. Ring, and J. Schmidhuber. Incremental basis construction from temporal difference error. In *International Conference on Machine Learning*, pages 481–488, 2011.

R. S. Sutton. Generalization in reinforcement learning: successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems*, pages 1038–1044, 1996.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10:1633–1685, 2009.

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

S. Thrun and A. Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of the Fourth Connectionist Models Summer School*, 1993.

J.N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

G. Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990.

C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

D. Wingate. Predictively defined representations of state. In M. Wiering and M. van Otterlo, editors, *Reinforcement Learning: State-of-the-Art*, pages 415–439. Springer-Verlag Berlin Heidelberg, 2012.

D. Wingate and S. P. Singh. On discovery and learning of models with predictive representations of state for agents with continuous actions and observations. In *International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1128–1135, 2007.

L. Wiskott. Slow feature analysis: a theoretical analysis of optimal free responses. *Neural Computation*, 15(9):2147–2177, 2003.

L. Wiskott and T. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

X. Xu. A sparse kernel-based least-squares temporal difference algorithm for reinforcement learning. In *Advances in Natural Computation*, volume 4221 of *Lecture Notes in Computer Science*, pages 47–56. Springer Berlin / Heidelberg, 2006.

Journal of Machine Learning Research 14 (2013) 2119-2149 Submitted 7/12; Revised 3/13; Published 7/13

# Distribution-Dependent Sample Complexity of Large Margin Learning

**Sivan Sabato** SIVAN.SABATO@MICROSOFT.COM
*Microsoft Research New England*
*1 Memorial Drive*
*Cambridge, MA 02142, USA*

**Nathan Srebro** NATI@TTIC.EDU
*Toyota Technological Institute at Chicago*
*6045 S. Kenwood Ave.*
*Chicago, IL 60637, USA*

**Naftali Tishby** TISHBY@CS.HUJI.AC.IL
*The Rachel and Selim Benin School of Computer Science and Engineering*
*The Hebrew University*
*Jerusalem 91904, Israel*

**Editor:** John Shawe-Taylor

## Abstract

We obtain a tight distribution-specific characterization of the sample complexity of large-margin classification with $L_2$ regularization: We introduce the *margin-adapted dimension*, which is a simple function of the second order statistics of the data distribution, and show distribution-specific upper *and* lower bounds on the sample complexity, both governed by the margin-adapted dimension of the data distribution. The upper bounds are universal, and the lower bounds hold for the rich family of sub-Gaussian distributions with independent features. We conclude that this new quantity tightly characterizes the true sample complexity of large-margin classification. To prove the lower bound, we develop several new tools of independent interest. These include new connections between shattering and hardness of learning, new properties of shattering with linear classifiers, and a new lower bound on the smallest eigenvalue of a random Gram matrix generated by sub-Gaussian variables. Our results can be used to quantitatively compare large margin learning to other learning rules, and to improve the effectiveness of methods that use sample complexity bounds, such as active learning.

**Keywords:** supervised learning, sample complexity, linear classifiers, distribution-dependence

## 1. Introduction

In this paper we pursue a tight characterization of the sample complexity of learning a classifier, under a particular data distribution, and using a particular learning rule.

Most learning theory work focuses on providing sample-complexity upper bounds which hold for a large class of distributions. For instance, standard distribution-free VC-dimension analysis shows that if one uses the Empirical Risk Minimization (ERM) learning rule, then the sample complexity of learning a classifier from a hypothesis class with VC-dimension $d$ is at most $O\left(\frac{d}{\varepsilon^2}\right)$, where $\varepsilon$ is the maximal excess classification error (Vapnik and Chervonenkis, 1971; Anthony and Bartlett, 1999). Such upper bounds can be useful for understanding the positive aspects of a learning rule.

©2013 Sivan Sabato, Nati Srebro and Naftali Tishby.

However, it is difficult to understand the deficiencies of a learning rule, or to compare between different rules, based on upper bounds alone. This is because it is possible, and is often the case, that the actual number of samples required to get a low error, for a given data distribution using a given learning rule, is much lower than the sample-complexity upper bound. As a simple example, suppose that the support of a given distribution is restricted to a subset of the domain. If the VC-dimension of the hypothesis class, when restricted to this subset, is smaller than $d$, then learning with respect to this distribution will require less examples than the upper bound predicts.

Of course, some sample complexity upper bounds are known to be tight or to have an almost-matching lower bound. For instance, the VC-dimension upper bound is tight (Vapnik and Chervonenkis, 1974). This means that there exists *some* data distribution in the class covered by the upper bound, for which this bound cannot be improved. Such a tightness result shows that there cannot be a better upper bound that holds for this entire class of distributions. But it does not imply that the upper bound characterizes the true sample complexity for every *specific* distribution in the class.

The goal of this paper is to identify a simple quantity, which is a function of the distribution, that *does* precisely characterize the sample complexity of learning this distribution under a specific learning rule. We focus on the important hypothesis class of linear classifiers, and on the popular rule of *margin-error-minimization* (MEM). Under this learning rule, a learner must always select a linear classifier that minimizes the margin-error on the input sample.

The VC-dimension of the class of homogeneous linear classifiers in $\mathbb{R}^d$ is $d$ (Dudley, 1978). This implies a sample complexity upper bound of $O\left(\frac{d}{\varepsilon^2}\right)$ using any MEM algorithm, where $\varepsilon$ is the excess error relative to the optimal margin error.[1] We also have that the sample complexity of any MEM algorithm is at most $O\left(\frac{B^2}{\gamma^2\varepsilon^2}\right)$, where $B^2$ is the average squared norm of the data and $\gamma$ is the size of the margin (Bartlett and Mendelson, 2002). Both of these upper bounds are tight. For instance, there exists a distribution with an average squared norm of $B^2$ that requires as many as $C \cdot \frac{B^2}{\gamma^2\varepsilon^2}$ examples to learn, for some universal constant $C$ (see, e.g., Anthony and Bartlett, 1999). However, the VC-dimension upper bound indicates, for instance, that if a distribution induces a large average norm but is supported by a low-dimensional sub-space, then the true number of examples required to reach a low error is much smaller. Thus, neither of these upper bounds fully describes the sample complexity of MEM for a *specific* distribution.

We obtain a tight distribution-specific characterization of the sample complexity of large-margin learning for a rich class of distributions. We present a new quantity, termed the *margin-adapted dimension*, and use it to provide a tighter distribution-dependent upper bound, and a matching distribution-dependent lower bound for MEM. The upper bound is universal, and the lower bound holds for a rich class of distributions with independent features.

The margin-adapted dimension refines both the dimension and the average norm of the data distribution, and can be easily calculated from the covariance matrix and the mean of the distribution. We denote this quantity, for a margin of $\gamma$, by $k_\gamma$. Our sample-complexity upper bound shows that $\tilde{O}(\frac{k_\gamma}{\varepsilon^2})$ examples suffice in order to learn any distribution with a margin-adapted dimension of $k_\gamma$ using a MEM algorithm with margin $\gamma$. We further show that for every distribution in a rich family of 'light tailed' distributions—specifically, product distributions of sub-Gaussian random variables—the number of samples required for learning by minimizing the margin error is at least $\Omega(k_\gamma)$.

1. This upper bound can be derived analogously to the result for ERM algorithms with $\varepsilon$ being the excess classification error. It can also be concluded from our analysis in Theorem 11 below.

Denote by $m(\varepsilon,\gamma,D)$ the number of examples required to achieve an excess error of no more than $\varepsilon$ relative to the best possible $\gamma$-margin error for a specific distribution $D$, using a MEM algorithm. Our main result shows the following matching distribution-specific upper and lower bounds on the sample complexity of MEM:

$$\Omega(k_\gamma(D)) \leq m(\varepsilon,\gamma,D) \leq \tilde{O}\left(\frac{k_\gamma(D)}{\varepsilon^2}\right). \tag{1}$$

Our tight characterization, and in particular the distribution-specific lower bound on the sample complexity that we establish, can be used to compare large-margin ($L_2$ regularized) learning to other learning rules. We provide two such examples: we use our lower bound to rigorously establish a sample complexity gap between $L_1$ and $L_2$ regularization previously studied in Ng (2004), and to show a large gap between discriminative and generative learning on a Gaussian-mixture distribution. The tight bounds can also be used for active learning algorithms in which sample-complexity bounds are used to decide on the next label to query.

In this paper we focus only on large margin classification. But in order to obtain the distribution-specific lower bound, we develop new tools that we believe can be useful for obtaining lower bounds also for other learning rules. We provide several new results which we use to derive our main results. These include:

- Linking the fat-shattering of a sample with non-negligible probability to a difficulty of learning using MEM.
- Showing that for a convex hypothesis class, fat-shattering is equivalent to shattering with exact margins.
- Linking the fat-shattering of a set of vectors with the eigenvalues of the Gram matrix of the vectors.
- Providing a new lower bound for the smallest eigenvalue of a random Gram matrix generated by sub-Gaussian variables. This bound extends previous results in analysis of random matrices.

### 1.1 Paper Structure

We discuss related work on sample-complexity upper bounds in Section 2. We present the problem setting and notation in Section 3, and provide some necessary preliminaries in Section 4. We then introduce the margin-adapted dimension in Section 5. The sample-complexity upper bound is proved in Section 6. We prove the lower bound in Section 7. In Section 8 we show that any non-trivial sample-complexity lower bound for more general distributions must employ properties other than the covariance matrix of the distribution. We summarize and discuss implication in Section 9. Proofs omitted from the text are provided in Appendix A

## 2. Related Work

As mentioned above, most work on "sample complexity lower bounds" is directed at proving that under some set of assumptions, there exists a data distribution for which one needs at least a certain number of examples to learn with required error and confidence (for instance Antos and Lugosi,

1998; Ehrenfeucht et al., 1988; Gentile and Helmbold, 1998). This type of a lower bound does not, however, indicate much on the sample complexity of other distributions under the same set of assumptions.

For distribution-specific lower bounds, the classical analysis of Vapnik (1995, Theorem 16.6) provides not only sufficient but also necessary conditions for the learnability of a hypothesis class with respect to a specific distribution. The essential condition is that the metric entropy of the hypothesis class with respect to the distribution be sub-linear in the limit of an infinite sample size. In some sense, this criterion can be seen as providing a "lower bound" on learnability for a specific distribution. However, we are interested in finite-sample convergence rates, and would like those to depend on simple properties of the distribution. The asymptotic arguments involved in Vapnik's general learnability claim do not lend themselves easily to such analysis.

Benedek and Itai (1991) show that if the distribution is known to the learner, a specific hypothesis class is learnable if and only if there is a finite $\varepsilon$-cover of this hypothesis class with respect to the distribution. Ben-David et al. (2008) consider a similar setting, and prove sample complexity lower bounds for learning with any data distribution, for some binary hypothesis classes on the real line. Vayatis and Azencott (1999) provide distribution-specific sample complexity upper bounds for hypothesis classes with a limited VC-dimension, as a function of how balanced the hypotheses are with respect to the considered distributions. These bounds are not tight for all distributions, thus they also do not fully characterize the distribution-specific sample complexity.

As can be seen in Equation (1), we do not tightly characterize the dependence of the sample complexity on the desired error (as done, for example, in Steinwart and Scovel, 2007), thus our bounds are not tight for asymptotically small error levels. Our results are most significant if the desired error level is a constant well below chance but bounded away from zero. This is in contrast to classical statistical asymptotics that are also typically tight, but are valid only for very small $\varepsilon$. As was recently shown by Liang and Srebro (2010), the sample complexity for very small $\varepsilon$ (in the classical statistical asymptotic regime) depends on quantities that can be very different from those that control the sample complexity for moderate error rates, which are more relevant for machine learning.

## 3. Problem Setting and Definitions

Consider a domain $\mathcal{X}$, and let $D$ be a distribution over $\mathcal{X} \times \{\pm 1\}$. We denote by $D_X$ the marginal distribution of $D$ on $\mathcal{X}$. The misclassification error of a classifier $h : \mathcal{X} \to \mathbb{R}$ on a distribution $D$ is

$$\ell_0(h, D) \triangleq \mathbb{P}_{(X,Y)\sim D}[Y \cdot h(X) \leq 0].$$

The margin error of a classifier $w$ with respect to a margin $\gamma > 0$ on $D$ is

$$\ell_\gamma(h, D) \triangleq \mathbb{P}_{(X,Y)\sim D}[Y \cdot h(X) \leq \gamma].$$

For a given hypothesis class $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$, the best achievable margin error on $D$ is

$$\ell^*_\gamma(\mathcal{H}, D) \triangleq \inf_{h\in\mathcal{H}} \ell_\gamma(h, D).$$

We usually write simply $\ell^*_\gamma(D)$ since $\mathcal{H}$ is clear from context.

A labeled sample is a (multi-)set $S = \{(x_i, y_i)\}_{i=1}^m \subseteq \mathcal{X} \times \{\pm 1\}$. Given $S$, we denote the set of its examples without their labels by $S_X \triangleq \{x_1, \ldots, x_m\}$. We use $S$ also to refer to the uniform distribution over the elements in $S$. Thus the misclassification error of $h : \mathcal{X} \to \{\pm 1\}$ on $S$ is

$$\ell(h, S) \triangleq \frac{1}{m} |\{i \mid y_i \cdot h(x_i) \leq 0\}|,$$

and the $\gamma$-margin error on $S$ is

$$\ell_\gamma(h, S) \triangleq \frac{1}{m} |\{i \mid y_i \cdot h(x_i) \leq \gamma\}|.$$

A learning algorithm is a function $\mathcal{A} : \cup_{m=1}^\infty (\mathcal{X} \times \{\pm 1\})^m \to \mathbb{R}^\mathcal{X}$, that receives a training set as input, and returns a function for classifying objects in $\mathcal{X}$ into real values. The high-probability loss of an algorithm $\mathcal{A}$ with respect to samples of size $m$, a distribution $D$ and a confidence parameter $\delta \in (0,1)$ is

$$\ell(\mathcal{A}, D, m, \delta) = \inf\{\varepsilon \geq 0 \mid \mathbb{P}_{S \sim D^m}[\ell(\mathcal{A}(S), D) \geq \varepsilon] \leq \delta\}.$$

In this work we investigate the sample complexity of learning using margin-error minimization (MEM). The relevant class of algorithms is defined as follows.

**Definition 1** *An* margin-error minimization (MEM) algorithm $\mathcal{A}$ *maps a margin parameter* $\gamma > 0$ *to a learning algorithm* $\mathcal{A}_\gamma$*, such that*

$$\forall S \subseteq \mathcal{X} \times \{\pm 1\}, \quad \mathcal{A}_\gamma(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} \ell_\gamma(h, S).$$

The distribution-specific sample complexity for MEM algorithms is the sample size required to guarantee low excess error for the given distribution. Formally, we have the following definition.

**Definition 2 (Distribution-specific sample complexity)** *Fix a hypothesis class* $\mathcal{H} \subseteq \{\pm 1\}^\mathcal{X}$*. For* $\gamma > 0$*,* $\varepsilon, \delta \in [0,1]$*, and a distribution* $D$*, the* distribution-specific sample complexity*, denoted by* $m(\varepsilon, \gamma, D, \delta)$*, is the minimal sample size such that for any MEM algorithm* $\mathcal{A}$*, and for any* $m \geq m(\varepsilon, \gamma, D, \delta)$*,*

$$\ell_0(\mathcal{A}_\gamma, D, m, \delta) - \ell_\gamma^*(D) \leq \varepsilon.$$

Note that we require that *all* possible MEM algorithms do well on the given distribution. This is because we are interested in the MEM strategy in general, and thus we study the guarantees that can be provided regardless of any specific MEM implementation. We sometimes omit $\delta$ and write simply $m(\varepsilon, \gamma, D)$, to indicate that $\delta$ is assumed to be some fixed small constant.

In this work we focus on linear classifiers. For simplicity of notation, we assume a Euclidean space $\mathbb{R}^d$ for some integer $d$, although the results can be easily extended to any separable Hilbert space. For a real vector $x$, $\|x\|$ stands for the Euclidean norm. For a real matrix $\mathbb{X}$, $\|\mathbb{X}\|$ stands for the Euclidean operator norm.

Denote the unit ball in $\mathbb{R}^d$ by $\mathbb{B}_1^d \triangleq \{w \in \mathbb{R}^d \mid \|w\| \leq 1\}$. We consider the hypothesis class of homogeneous linear separators, $\mathcal{W} = \{x \mapsto \langle x, w \rangle \mid w \in \mathbb{B}_1^d\}$. We often slightly abuse notation by using $w$ to denote the mapping $x \mapsto \langle x, w \rangle$.

We often represent sets of vectors in $\mathbb{R}^d$ using matrices. We say that $\mathbb{X} \in \mathbb{R}^{m \times d}$ is the matrix of a set $\{x_1, \ldots, x_m\} \subseteq \mathbb{R}^d$ if the rows in the matrix are exactly the vectors in the set. For uniqueness, one may assume that the rows of $\mathbb{X}$ are sorted according to an arbitrary fixed full order on vectors in

$\mathbb{R}^d$. For a PSD matrix $\mathbb{X}$ denote the largest eigenvalue of $\mathbb{X}$ by $\lambda_{\max}(\mathbb{X})$ and the smallest eigenvalue by $\lambda_{\min}(\mathbb{X})$.

We use the $O$-notation as follows: $O(f(z))$ stands for $C_1 + C_2 f(z)$ for some constants $C_1, C_2 \geq 0$. $\Omega(f(z))$ stands for $C_2 f(z) - C_1$ for some constants $C_1, C_2 \geq 0$. $\widetilde{O}(f(z))$ stands for $f(z)p(\ln(z)) + C$ for some polynomial $p(\cdot)$ and some constant $C > 0$.

## 4. Preliminaries

As mentioned above, for the hypothesis class of linear classifiers $\mathcal{W}$, one can derive a sample-complexity upper bound of the form $O(B^2/\gamma^2\varepsilon^2)$, where $B^2 = \mathbb{E}_{X\sim D}[\|X\|^2]$ and $\varepsilon$ is the excess error relative to the $\gamma$-margin loss. This can be achieved as follows (Bartlett and Mendelson, 2002). Let $\mathcal{Z}$ be some domain. The empirical Rademacher complexity of a class of functions $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{Z}}$ with respect to a set $S = \{z_i\}_{i\in[m]} \subseteq \mathcal{Z}$ is

$$\mathcal{R}(\mathcal{F}, S) = \frac{1}{m}\mathbb{E}_\sigma[|\sup_{f\in\mathcal{F}} \sum_{i\in[m]} \sigma_i f(z_i)|],$$

where $\sigma = (\sigma_1, \ldots, \sigma_m)$ are $m$ independent uniform $\{\pm 1\}$-valued variables. The average Rademacher complexity of $\mathcal{F}$ with respect to a distribution $D$ over $\mathcal{Z}$ and a sample size $m$ is

$$\mathcal{R}_m(\mathcal{F}, D) = \mathbb{E}_{S\sim D^m}[\mathcal{R}(\mathcal{F}, S)].$$

Assume a hypothesis class $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$ and a loss function $\ell : \{\pm 1\} \times \mathbb{R} \to \mathbb{R}$. For a hypothesis $h \in \mathcal{H}$, we introduce the function $h_\ell : \mathcal{X} \times \{\pm 1\} \to \mathbb{R}$, defined by $h_\ell(x, y) = \ell(y, h(x))$. We further define the function class $\mathcal{H}_\ell = \{h_\ell \mid h \in \mathcal{H}\} \subseteq \mathbb{R}^{\mathcal{X}\times\{\pm 1\}}$.

Assume that the range of $\mathcal{H}_\ell$ is in $[0, 1]$. For any $\delta \in (0, 1)$, with probability of $1 - \delta$ over the draw of samples $S \subseteq \mathcal{X} \times \{\pm 1\}$ of size $m$ according to $D$, every $h \in \mathcal{H}$ satisfies (Bartlett and Mendelson, 2002)

$$\ell(h, D) \leq \ell(h, S) + 2\mathcal{R}_m(\mathcal{H}_\ell, D) + \sqrt{\frac{8\ln(2/\delta)}{m}}. \tag{2}$$

To get the desired upper bound for linear classifiers we use the *ramp loss*, which is defined as follows. For a number $r$, denote $[\![r]\!] \triangleq \min(\max(r, 0), 1)$. The $\gamma$-ramp-loss of a labeled example $(x, y) \in \mathbb{R}^d \times \{\pm 1\}$ with respect to a linear classifier $w \in \mathbb{B}_1^d$ is $\text{ramp}_\gamma(w, x, y) = [\![1 - y\langle w, x\rangle/\gamma]\!]$. Let $\text{ramp}_\gamma(w, D) = \mathbb{E}_{(X,Y)\sim D}[\text{ramp}_\gamma(w, X, Y)]$, and denote the class of ramp-loss functions by

$$\text{RAMP}_\gamma = \{(x, y) \mapsto \text{ramp}_\gamma(w, x, y) \mid w \in \mathbb{B}_1^d\}.$$

The ramp-loss is upper-bounded by the margin loss and lower-bounded by the misclassification error. Therefore, the following result can be shown.

**Proposition 3** *For any MEM algorithm $\mathcal{A}$, we have*

$$\ell_0(\mathcal{A}_\gamma, D, m, \delta) \leq \ell_\gamma^*(\mathcal{H}, D) + 2\mathcal{R}_m(\text{RAMP}_\gamma, D) + \sqrt{\frac{14\ln(2/\delta)}{m}}. \tag{3}$$

We give the proof in Appendix A.1 for completeness. Since the $\gamma$-ramp loss is $1/\gamma$ Lipschitz, it follows from Bartlett and Mendelson (2002) that

$$\mathcal{R}_m(\text{RAMP}_\gamma, D) \leq \sqrt{\frac{B^2}{\gamma^2 m}}.$$

Combining this with Proposition 3 we can conclude a sample complexity upper bound of $O(B^2/\gamma^2\varepsilon^2)$.

In addition to the Rademacher complexity, we will also use the classic notions of *fat-shattering* (Kearns and Schapire, 1994) and *pseudo-shattering* (Pollard, 1984), defined as follows.

**Definition 4** *Let $\mathcal{F}$ be a set of functions $f : \mathcal{X} \to \mathbb{R}$, and let $\gamma > 0$. The set $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ is* $\gamma$-shattered *by $\mathcal{F}$ with the witness $r \in \mathbb{R}^m$ if for all $y \in \{\pm 1\}^m$ there is an $f \in \mathcal{F}$ such that $\forall i \in [m], y[i](f(x_i) - r[i]) \geq \gamma$.*

The $\gamma$-shattering dimension of a hypothesis class is the size of the largest set that is $\gamma$-shattered by this class. We say that a set is $\gamma$-*shattered at the origin* if it is $\gamma$-shattered with the zero vector as a witness.

**Definition 5** *Let $\mathcal{F}$ be a set of functions $f : \mathcal{X} \to \mathbb{R}$, and let $\gamma > 0$. The set $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ is* pseudo-shattered *by $\mathcal{F}$ with the witness $r \in \mathbb{R}^m$ if for all $y \in \{\pm 1\}^m$ there is an $f \in \mathcal{F}$ such that $\forall i \in [m], y[i](f(x_i) - r[i]) > 0$.*

The pseudo-dimension of a hypothesis class is the size of the largest set that is pseudo-shattered by this class.

## 5. The Margin-Adapted Dimension

When considering learning of linear classifiers using MEM, the dimension-based upper bound and the norm-based upper bound are both tight in the worst-case sense, that is, they are the best bounds that rely only on the dimensionality or only on the norm respectively. Nonetheless, neither is tight in a distribution-specific sense: If the average norm is unbounded while the dimension is small, then there can be an arbitrarily large gap between the true distribution-dependent sample complexity and the bound that depends on the average norm. If the converse holds, that is, the dimension is arbitrarily large while the average-norm is bounded, then the dimensionality bound is loose.

Seeking a tight distribution-specific analysis, one simple approach to tighten these bounds is to consider their minimum, which is proportional to $\min(d, B^2/\gamma^2)$. Trivially, this is an upper bound on the sample complexity as well. However, this simple combination is also not tight: Consider a distribution in which there are a few directions with very high variance, but the combined variance in all other directions is small (see Figure 1). We will show that in such situations the sample complexity is characterized not by the minimum of dimension and norm, but by the sum of the number of high-variance dimensions and the average squared norm in the other directions. This behavior is captured by the *margin-adapted dimension* which we presently define, using the following auxiliary definition.

**Definition 6** *Let $b > 0$ and let $k$ be a positive integer. A distribution $D_X$ over $\mathbb{R}^d$ is* $(b,k)$-limited *if there exists a sub-space $V \subseteq \mathbb{R}^d$ of dimension $d - k$ such that $\mathbb{E}_{X \sim D_X}[\|\mathbb{O}_V \cdot X\|^2] \leq b$, where $\mathbb{O}_V$ is an orthogonal projection onto $V$.*

**Definition 7 (margin-adapted dimension)** *The* margin-adapted dimension *of a distribution $D_X$, denoted by $k_\gamma(D_X)$, is the minimum $k$ such that the distribution is $(\gamma^2 k, k)$-limited.*

We sometimes drop the argument of $k_\gamma$ when it is clear from context. It is easy to see that for any distribution $D_X$ over $\mathbb{R}^d$, $k_\gamma(D_X) \leq \min(d, \mathbb{E}[\|X\|^2]/\gamma^2)$. Moreover, $k_\gamma$ can be much smaller than this minimum. For example, consider a random vector $X \in \mathbb{R}^{1001}$ with mean zero and statistically independent coordinates, such that the variance of the first coordinate is 1000, and the variance in each remaining coordinate is 0.001. We have $k_1 = 1$ but $d = \mathbb{E}[\|X\|^2] = 1001$.

$k_\gamma(D_X)$ can be calculated from the uncentered covariance matrix $\mathbb{E}_{X\sim D_X}[XX^T]$ as follows: Let $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_d \geq 0$ be the eigenvalues of this matrix. Then

$$k_\gamma = \min\{k \mid \sum_{i=k+1}^{d} \lambda_i \leq \gamma^2 k\}. \tag{4}$$

A quantity similar to this definition of $k_\gamma$ was studied previously in Bousquet (2002). The eigenvalues of the *empirical* covariance matrix were used to provide sample complexity bounds, for instance in Schölkopf et al. (1999). However, $k_\gamma$ generates a different type of bound, since it is defined based on the eigenvalues of the distribution and not of the sample. We will see that for small finite samples, the latter can be quite different from the former.

Finally, note that while we define the margin-adapted dimension for a finite-dimensional space for ease of notation, the same definition carries over to an infinite-dimensional Hilbert space. Moreover, $k_\gamma$ can be finite even if some of the eigenvalues $\lambda_i$ are infinite, implying a distribution with unbounded covariance.

## 6. A Distribution-Dependent Upper Bound

In this section we prove an upper bound on the sample complexity of learning with MEM, using the margin-adapted dimension. We do this by providing a tighter upper bound for the Rademacher complexity of $\text{RAMP}_\gamma$. We bound $\mathcal{R}_m(\text{RAMP}_\gamma, D)$ for any $(B^2, k)$-limited distribution $D_X$, using $L_2$ covering numbers, defined as follows.

Let $(\mathcal{X}, \|\cdot\|_\circ)$ be a normed space. An $\eta$-covering of a set $\mathcal{F} \subseteq \mathcal{X}$ with respect to the norm $\|\cdot\|_\circ$ is a set $\mathcal{C} \subseteq \mathcal{X}$ such that for any $f \in \mathcal{F}$ there exists a $g \in \mathcal{C}$ such that $\|f - g\|_\circ \leq \eta$. The covering-number for given $\eta > 0$, $\mathcal{F}$ and $\circ$ is the size of the smallest such $\eta$-covering, and is denoted by $\mathcal{N}(\eta, \mathcal{F}, \circ)$. Let $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^d$. For a function $f : \mathbb{R}^d \to \mathbb{R}$, the $L_2(S)$ norm of $f$ is $\|f\|_{L_2(S)} = \sqrt{\mathbb{E}_{X\sim S}[f(X)^2]}$. Thus, we consider covering-numbers of the form $\mathcal{N}(\eta, \text{RAMP}_\gamma, L_2(S))$.

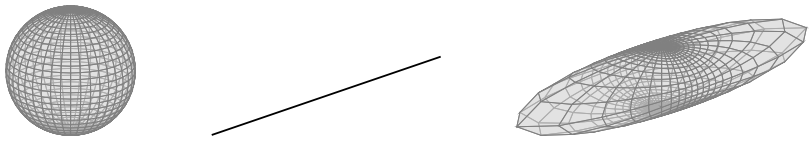


Figure 1: Illustrating covariance matrix ellipsoids. left: norm bound is tight; middle: dimension bound is tight; right: neither bound is tight.

The empirical Rademacher complexity of a function class can be bounded by the $L_2$ covering numbers of the same function class as follows (Mendelson, 2002, Lemma 3.7): Let $\varepsilon_i = 2^{-i}$. Then

$$\sqrt{m}\mathcal{R}(\text{RAMP}_\gamma, S) \leq C \sum_{i \in [N]} \varepsilon_{i-1} \sqrt{\ln \mathcal{N}(\varepsilon_i, \text{RAMP}_\gamma, L_2(S))} + 2\varepsilon_N \sqrt{m}. \tag{5}$$

To bound the covering number of $\text{RAMP}_\gamma$, we will restate the functions in $\text{RAMP}_\gamma$ as sums of two functions, each selected from a function class with bounded complexity. The first function class will be bounded because of the norm bound on the subspace $V$ used in Definition 6, and the second function class will have a bounded pseudo-dimension. However, the second function class will depend on the choice of the first function in the sum. Therefore, we require the following lemma, which provides an upper bound on such sums of functions. We use the notion of a *Hausdorff distance* between two sets $\mathcal{G}_1, \mathcal{G}_2 \subseteq \mathcal{X}$, defined as $\Delta_H(\mathcal{G}_1, \mathcal{G}_2) = \sup_{g_1 \in \mathcal{G}_1} \inf_{g_2 \in \mathcal{G}_2} \|g_1 - g_2\|_\circ$.

**Lemma 8** *Let $(\mathcal{X}, \|\cdot\|_\circ)$ be a normed space. Let $\mathcal{F} \subseteq \mathcal{X}$ be a set, and let $\mathcal{G} : \mathcal{X} \to 2^{\mathcal{X}}$ be a mapping from objects in $\mathcal{X}$ to sets of objects in $\mathcal{X}$. Assume that $\mathcal{G}$ is c-Lipschitz with respect to the Hausdorff distance on sets, that is assume that*

$$\forall f_1, f_2 \in \mathcal{X}, \Delta_H(\mathcal{G}(f_1), \mathcal{G}(f_2)) \leq c\|f_1 - f_2\|_\circ.$$

*Let $\mathcal{F}_\mathcal{G} = \{f + g \mid f \in \mathcal{F}, g \in \mathcal{G}(f)\}$. Then*

$$\mathcal{N}(\eta, \mathcal{F}_\mathcal{G}, \circ) \leq \mathcal{N}(\eta/(2+c), \mathcal{F}, \circ) \cdot \sup_{f \in \mathcal{F}} \mathcal{N}(\eta/(2+c), \mathcal{G}(f), \circ).$$

**Proof** For any set $A \subseteq \mathcal{X}$, denote by $\mathcal{C}_A$ a minimal $\eta$-covering for $A$ with respect to $\|\cdot\|_\circ$, so that $|\mathcal{C}_A| = \mathcal{N}(\eta, A, \circ)$. Let $f + g \in \mathcal{F}_\mathcal{G}$ such that $f \in \mathcal{F}, g \in \mathcal{G}(f)$. There is a $\hat{f} \in \mathcal{C}_\mathcal{F}$ such that $\|f - \hat{f}\|_\circ \leq \eta$. In addition, by the Lipschitz assumption there is a $\tilde{g} \in \mathcal{G}(\hat{f})$ such that $\|g - \tilde{g}\|_\circ \leq c\|f - \hat{f}\|_\circ \leq c\eta$. Lastly, there is a $\hat{g} \in \mathcal{C}_{\mathcal{G}(\hat{f})}$ such that $\|\tilde{g} - \hat{g}\|_\circ \leq \eta$. Therefore

$$\|f + g - (\hat{f} + \hat{g})\|_\circ \leq \|f - \hat{f}\|_\circ + \|g - \tilde{g}\|_\circ + \|\tilde{g} - \hat{g}\|_\circ \leq (2+c)\eta.$$

Thus the set $\{f + g \mid f \in \mathcal{C}_\mathcal{F}, g \in \mathcal{C}_{\mathcal{G}(f)}\}$ is a $(2+c)\eta$ cover of $\mathcal{F}_\mathcal{G}$. The size of this cover is at most $|\mathcal{C}_\mathcal{F}| \cdot \sup_{f \in \mathcal{F}} |\mathcal{C}_{\mathcal{G}(f)}| \leq \mathcal{N}(\eta, \mathcal{F}, \circ) \cdot \sup_{f \in \mathcal{F}} \mathcal{N}(\eta, \mathcal{G}(f), \circ)$. ■

The following lemma provides us with a useful class of mappings which are 1-Lipschitz with respect to the Hausdorff distance, as required in Lemma 8. The proof is provided in Appendix A.2.

**Lemma 9** *Let $f : \mathcal{X} \to \mathbb{R}$ be a function and let $Z \subseteq \mathbb{R}^\mathcal{X}$ be a function class over some domain $\mathcal{X}$. Let $\mathcal{G} : \mathbb{R}^\mathcal{X} \to 2^{\mathbb{R}^\mathcal{X}}$ be the mapping defined by*

$$\mathcal{G}(f) \triangleq \{x \mapsto [\![f(x) + z(x)]\!] - f(x) \mid z \in Z\}. \tag{6}$$

*Then $\mathcal{G}$ is 1-Lipschitz with respect to the Hausdorff distance.*

The function class induced by the mapping above preserves the pseudo-dimension of the original function class, as the following lemma shows. The proof is provided in Appendix A.3.

**Lemma 10** *Let $f : \mathcal{X} \to \mathbb{R}$ be a function and let $Z \subseteq \mathbb{R}^{\mathcal{X}}$ be a function class over some domain $\mathcal{X}$. Let $\mathcal{G}(f)$ be defined as in Equation (6). Then the pseudo-dimension of $\mathcal{G}(f)$ is at most the pseudo-dimension of $Z$.*

Equipped with these lemmas, we can now provide the new bound on the Rademacher complexity of $\text{RAMP}_\gamma$ in the following theorem. The subsequent corollary states the resulting sample-complexity upper bound for MEM, which depends on $k_\gamma$.

**Theorem 11** *Let $D$ be a distribution over $\mathbb{R}^d \times \{\pm 1\}$, and assume $D_X$ is $(B^2, k)$-limited. Then*

$$\mathcal{R}(\text{RAMP}_\gamma, D) \leq \sqrt{\frac{O(k + B^2/\gamma^2)\ln(m)}{m}}.$$

**Proof** In this proof all absolute constants are assumed to be positive and are denoted by $C$ or $C_i$ for some integer $i$. Their values may change from line to line or even within the same line.

Consider the distribution $\tilde{D}$ which results from drawing $(X,Y) \sim D$ and emitting $(Y \cdot X, 1)$. It too is $(B^2,k)$-limited, and $\mathcal{R}(\text{RAMP}_\gamma, D) = \mathcal{R}(\text{RAMP}_\gamma, \tilde{D})$. Therefore, we assume without loss of generality that for all $(X,Y)$ drawn from $D$, $Y = 1$. Accordingly, we henceforth omit the $y$ argument from $\text{ramp}_\gamma(w,x,y)$ and write simply $\text{ramp}_\gamma(w,x) \triangleq \text{ramp}_\gamma(w,x,1)$.

Following Definition 6, Let $\mathbb{O}_V$ be an orthogonal projection onto a sub-space $V$ of dimension $d-k$ such that $\mathbb{E}_{X \sim D_X}[\|\mathbb{O}_V \cdot X\|^2] \leq B^2$. Let $\bar{V}$ be the complementary sub-space to $V$. For a set $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^d$, denote $B(S) = \sqrt{\frac{1}{m}\sum_{i \in [m]} \|\mathbb{O}_V \cdot X\|^2}$.

We would like to use Equation (5) to bound the Rademacher complexity of $\text{RAMP}_\gamma$. Therefore, we will bound $\mathcal{N}(\eta, \text{RAMP}_\gamma, L_2(S))$ for $\eta > 0$. Note that

$$\text{ramp}_\gamma(w,x) = [\![1 - \langle w,x\rangle/\gamma]\!] = 1 - [\![\langle w,x\rangle/\gamma]\!].$$

Shifting by a constant and negating do not change the covering number of a function class. Therefore, $\mathcal{N}(\eta, \text{RAMP}_\gamma, L_2(S))$ is equal to the covering number of $\{x \mapsto [\![\langle w,x\rangle/\gamma]\!] \mid w \in \mathbb{B}_1^d\}$. Moreover, let

$$\text{RAMP}'_\gamma = \{x \mapsto [\![\langle w_a + w_b, x\rangle/\gamma]\!] \mid w_a \in \mathbb{B}_1^d \cap V,\, w_b \in \bar{V}\}.$$

Then $\{x \mapsto [\![\langle w,x\rangle/\gamma]\!] \mid w \in \mathbb{B}_1^d\} \subseteq \text{RAMP}'_\gamma$, thus it suffices to bound $\mathcal{N}(\eta, \text{RAMP}'_\gamma, L_2(S))$. To do that, we show that $\text{RAMP}'_\gamma$ satisfies the assumptions of Lemma 8 for the normed space $(\mathbb{R}^{\mathbb{R}^d}, \|\cdot\|_{L_2(S)})$. Define

$$\mathcal{F} = \{x \mapsto \langle w_a, x\rangle/\gamma \mid w_a \in \mathbb{B}_1^d \cap V\}.$$

Let $\mathcal{G} : \mathbb{R}^{\mathbb{R}^d} \to 2^{\mathbb{R}^{\mathbb{R}^d}}$ be the mapping defined by

$$\mathcal{G}(f) \triangleq \{x \mapsto [\![f(x) + \langle w_b, x\rangle/\gamma]\!] - f(x) \mid w_b \in \bar{V}\}.$$

Clearly, $\mathcal{F}_\mathcal{G} = \{f + g \mid f \in \mathcal{F}, g \in \mathcal{G}(f)\} = \text{RAMP}'_\gamma$. Furthermore, by Lemma 9, $\mathcal{G}$ is 1-Lipschitz with respect to the Hausdorff distance. Thus, by Lemma 8

$$\mathcal{N}(\eta, \text{RAMP}'_\gamma, L_2(S)) \leq \mathcal{N}(\eta/3, \mathcal{F}, L_2(S)) \cdot \sup_{f \in \mathcal{F}} \mathcal{N}(\eta/3, \mathcal{G}(f), L_2(S)). \tag{7}$$

We now proceed to bound the two covering numbers on the right hand side. First, consider $\mathcal{N}(\eta/3, \mathcal{G}(f), L_2(S))$. By Lemma 10, the pseudo-dimension of $\mathcal{G}(f)$ is the same as the pseudo-dimension of $\{x \mapsto \langle w,x\rangle/\gamma \mid w \in \bar{V}\}$, which is exactly $k$, the dimension of $\bar{V}$. The $L_2$ covering

number of $\mathcal{G}(f)$ can be bounded by the pseudo-dimension of $\mathcal{G}(f)$ as follows (see, e.g., Bartlett, 2006, Theorem 3.1):

$$\mathcal{N}(\eta/3, \mathcal{G}(f), L_2(S)) \leq C_1 \left(\frac{C_2}{\eta^2}\right)^k. \tag{8}$$

Second, consider $\mathcal{N}(\eta/3, \mathcal{F}, L_2(S))$. Sudakov's minoration theorem (Sudakov 1971, and see also Ledoux and Talagrand, 1991, Theorem 3.18) states that for any $\eta > 0$

$$\ln \mathcal{N}(\eta, \mathcal{F}, L_2(S)) \leq \frac{C}{m\eta^2} \mathbb{E}_s^2[\sup_{f\in\mathcal{F}} \sum_{i\in[m]} s_i f(x_i)],$$

where $s = (s_1, \ldots, s_m)$ are independent standard normal variables. The right-hand side can be bounded as follows:

$$\begin{aligned}
&\gamma \mathbb{E}_s[\sup_{f\in\mathcal{F}} |\sum_{i=1}^m s_i f(x_i)|] = \mathbb{E}_s[\sup_{w\in\mathbb{B}_1^d\cap V} |\langle w, \sum_{i=1}^m s_i x_i\rangle|] \\
&\leq \mathbb{E}_s[\|\sum_{i=1}^m s_i \mathbb{O}_V x_i\|] \leq \sqrt{\mathbb{E}_s[\|\sum_{i=1}^m s_i \mathbb{O}_V x_i\|^2]} = \sqrt{\sum_{i\in[m]} \|\mathbb{O}_V x_i\|^2} = \sqrt{m} B(S).
\end{aligned}$$

Therefore $\ln \mathcal{N}(\eta, \mathcal{F}, L_2(S)) \leq C\frac{B^2(S)}{\gamma^2\eta^2}$. Substituting this and Equation (8) for the right-hand side in Equation (7), and adjusting constants, we get

$$\ln \mathcal{N}(\eta, \text{RAMP}_\gamma, L_2(S)) \leq \ln \mathcal{N}(\eta, \text{RAMP}'_\gamma, L_2(S)) \leq C_1(1 + k\ln(\frac{C_2}{\eta}) + \frac{B^2(S)}{\gamma^2\eta^2}),$$

To finalize the proof, we plug this inequality into Equation (5) to get

$$\begin{aligned}
&\sqrt{m}\mathcal{R}(\text{RAMP}_\gamma, S) \leq C_1 \sum_{i\in[N]} \varepsilon_{i-1} \sqrt{1 + k\ln(C_2/\varepsilon_i) + \frac{B^2(S)}{\gamma^2\varepsilon_i^2}} + 2\varepsilon_N\sqrt{m} \\
&\leq C_1 \left(\sum_{i\in[N]} \varepsilon_{i-1} \left(1 + \sqrt{k\ln(C_2/\varepsilon_i)} + \sqrt{\frac{B^2(S)}{\gamma^2\varepsilon_i^2}}\right)\right) + 2\varepsilon_N\sqrt{m} \\
&= C_1 \left(\sum_{i\in[N]} 2^{-i+1} + \sqrt{k} \sum_{i\in[N]} 2^{-i+1}\ln(C_2/2^{-i}) + \sum_{i\in[N]} \frac{B(S)}{\gamma}\right) + 2^{-N+1}\sqrt{m} \\
&\leq C\left(1 + \sqrt{k} + \frac{B(S)\cdot N}{\gamma}\right) + 2^{-N+1}\sqrt{m}.
\end{aligned}$$

In the last inequality we used the fact that $\sum_i i2^{-i+1} \leq 4$. Setting $N = \ln(2m)$ we get

$$\mathcal{R}(\text{RAMP}_\gamma, S) \leq \frac{C}{\sqrt{m}}\left(1 + \sqrt{k} + \frac{B(S)\ln(2m)}{\gamma}\right).$$

Taking expectation over both sides, and noting that $\mathbb{E}[B(S)] \leq \sqrt{\mathbb{E}[B^2(S)]} \leq B$, we get

$$\mathcal{R}(\text{RAMP}_\gamma, S) \leq \frac{C}{\sqrt{m}}(1 + \sqrt{k} + \frac{B\ln(2m)}{\gamma}) \leq \sqrt{\frac{O(k + B^2\ln^2(2m)/\gamma^2)}{m}}.$$

■

**Corollary 12 (Sample complexity upper bound)** *Let $D$ be a distribution over $\mathbb{R}^d \times \{\pm 1\}$. Then*

$$m(\varepsilon, \gamma, D) \leq \tilde{O}\left(\frac{k_\gamma(D_X)}{\varepsilon^2}\right).$$

**Proof** By Proposition 3, we have

$$\ell_0(\mathcal{A}_\gamma, D, m, \delta) \leq \ell_\gamma^*(\mathcal{W}, D) + 2\mathcal{R}_m(\text{RAMP}_\gamma, D) + \sqrt{\frac{14\ln(2/\delta)}{m}}.$$

By definition of $k_\gamma(D_X)$, $D_X$ is $(\gamma^2 k_\gamma, k_\gamma)$-limited. Therefore, by Theorem 11,

$$\mathcal{R}_m(\text{RAMP}_\gamma, D) \leq \sqrt{\frac{O(k_\gamma(D_X))\ln(m)}{m}}.$$

We conclude that

$$\ell_0(\mathcal{A}_\gamma, D, m, \delta) \leq \ell_\gamma^*(\mathcal{W}, D) + \sqrt{\frac{O(k_\gamma(D_X)\ln(m) + \ln(1/\delta))}{m}}.$$

Bounding the second right-hand term by $\varepsilon$, we conclude that $m(\varepsilon, \gamma, D) \leq \tilde{O}(k_\gamma/\varepsilon^2)$. ■

One should note that a similar upper bound can be obtained much more easily under a uniform upper bound on the eigenvalues of the uncentered covariance matrix.[2] However, such an upper bound would not capture the fact that a finite dimension implies a finite sample complexity, regardless of the size of the covariance. If one wants to estimate the sample complexity, then large covariance matrix eigenvalues imply that more examples are required to estimate the covariance matrix from a sample. However, these examples need not be labeled. Moreover, estimating the covariance matrix is not necessary to achieve the sample complexity, since the upper bound holds for any margin-error minimization algorithm.

## 7. A Distribution-Dependent Lower Bound

The new upper bound presented in Corollary 12 can be tighter than both the norm-only and the dimension-only upper bounds. But does the margin-adapted dimension characterize the true sample complexity of the distribution, or is it just another upper bound? To answer this question, we first need tools for deriving sample complexity lower bounds. Section 7.1 relates fat-shattering with a lower bound on sample complexity. In Section 7.2 we use this result to relate the smallest eigenvalue of a Gram-matrix to a lower bound on sample complexity. In Section 7.3 the family of sub-Gaussian product distributions is presented. We prove a sample-complexity lower bound for this family in Section 7.4.

2. This has been pointed out to us by an anonymous reviewer of this manuscript. An upper bound under sub-Gaussianity assumptions can be found in Sabato et al. (2010).

### 7.1 A Sample Complexity Lower Bound Based on Fat-Shattering

The ability to learn is closely related to the probability of a sample to be shattered, as evident in Vapnik's formulations of learnability as a function of the $\varepsilon$-entropy (Vapnik, 1995). It is well known that the maximal size of a shattered set dictates a sample-complexity upper bound. In the theorem below, we show that for some hypothesis classes it also implies a lower bound. The theorem states that if a sample drawn from a data distribution is fat-shattered with a non-negligible probability, then MEM can fail to learn a good classifier for this distribution.[3] This holds not only for linear classifiers, but more generally for all *symmetric* hypothesis classes. Given a domain $\mathcal{X}$, we say that a hypothesis class $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$ is symmetric if for all $h \in \mathcal{H}$, we have $-h \in \mathcal{H}$ as well. This clearly holds for the class of linear classifiers $\mathcal{W}$.

**Theorem 13** *Let $\mathcal{X}$ be some domain, and assume that $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$ is a symmetric hypothesis class. Let $D$ be a distribution over $\mathcal{X} \times \{\pm 1\}$. If the probability of a sample of size $m$ drawn from $D_X^m$ to be $\gamma$-shattered at the origin by $\mathcal{W}$ is at least $\eta$, then $m(\varepsilon, \gamma, D, \eta/2) \geq \lfloor m/2 \rfloor$ for all $\varepsilon < 1/2 - \ell_\gamma^*(D)$.*

**Proof** Let $\varepsilon \leq \frac{1}{2} - \ell_\gamma^*(D)$. We show a MEM algorithm $\mathcal{A}$ such that

$$\ell_0(\mathcal{A}_\gamma, D, \lfloor m/2 \rfloor, \eta/2) \geq \frac{1}{2} > \ell_\gamma^*(D) + \varepsilon,$$

thus proving the desired lower bound on $m(\varepsilon, \gamma, D, \eta/2)$.

Assume for simplicity that $m$ is even (otherwise replace $m$ with $m-1$). Consider two sets $S, \tilde{S} \subseteq \mathcal{X} \times \{\pm 1\}$, each of size $m/2$, such that $S_X \cup \tilde{S}_X$ is $\gamma$-shattered at the origin by $\mathcal{W}$. Then there exists a hypothesis $h_1 \in \mathcal{H}$ such that the following holds:

- For all $x \in S_X \cup \tilde{S}_X$, $|h_1(x)| \geq \gamma$.
- For all $(x, y) \in S$, $\text{sign}(h_1(x)) = y$.
- For all $(x, y) \in \tilde{S}$, $\text{sign}(h_1(x)) = -y$.

It follows that $\ell_\gamma(h_1, S) = 0$. In addition, let $h_2 = -h_1$. Then $\ell_\gamma(h_2, \tilde{S}) = 0$. Moreover, we have $h_2 \in \mathcal{H}$ due to the symmetry of $\mathcal{H}$. On each point in $\mathcal{X}$, at least one of $h_1$ and $h_2$ predict the wrong sign. Thus $\ell_0(h_1, D) + \ell_0(h_2, D) \geq 1$. It follows that for at least one of $i \in \{1, 2\}$, we have $\ell_0(h_i, D) \geq \frac{1}{2}$. Denote the set of hypotheses with a high misclassification error by

$$\mathcal{H}_\otimes = \{h \in \mathcal{H} \mid \ell_0(h, D) \geq \frac{1}{2}\}.$$

We have just shown that if $S_X \cup \tilde{S}_X$ is $\gamma$-shattered by $\mathcal{W}$ then at least one of the following holds: (1) $h_1 \in \mathcal{H}_\otimes \cap \text{argmin}_{h \in \mathcal{H}} \ell_\gamma(h, S)$ or (2) $h_2 \in \mathcal{H}_\otimes \cap \text{argmin}_{h \in \mathcal{H}} \ell_\gamma(h, \tilde{S})$.

Now, consider a MEM algorithm $\mathcal{A}$ such that whenever possible, it returns a hypothesis from $\mathcal{H}_\otimes$. Formally, given the input sample $S$, if $\mathcal{H}_\otimes \cap \text{argmin}_{h \in \mathcal{H}} \ell_\gamma(h, S) \neq \emptyset$, then

3. In contrast, the average Rademacher complexity cannot be used to derive general lower bounds for MEM algorithms, since it is related to the rate of uniform convergence of the entire hypothesis class, while MEM algorithms choose low-error hypotheses (see, e.g., Bartlett et al., 2005).

$\mathcal{A}(S) \in \mathcal{H}_\otimes \cap \operatorname{argmin}_{h\in\mathcal{H}} \ell_\gamma(h,S)$. It follows that

$$\begin{aligned}
&\mathbb{P}_{S\sim D^{m/2}}[\ell_0(\mathcal{A}(S),D) \geq \tfrac{1}{2}] \geq \mathbb{P}_{S\sim D^{m/2}}[\mathcal{H}_\otimes \cap \operatorname*{argmin}_{h\in\mathcal{H}} \ell_\gamma(h,S) \neq \emptyset] \\
&= \frac{1}{2}(\mathbb{P}_{S\sim D^{m/2}}[\mathcal{H}_\otimes \cap \operatorname*{argmin}_{h\in\mathcal{H}} \ell_\gamma(h,S) \neq \emptyset] + \mathbb{P}_{\tilde{S}\sim D^{m/2}}[\mathcal{H}_\otimes \cap \operatorname*{argmin}_{h\in\mathcal{H}} \ell_\gamma(h,\tilde{S}) \neq \emptyset]) \\
&\geq \frac{1}{2}(\mathbb{P}_{S,\tilde{S}\sim D^{m/2}}[\mathcal{H}_\otimes \cap \operatorname*{argmin}_{h\in\mathcal{H}} \ell_\gamma(h,S) \neq \emptyset \text{ OR } \mathcal{H}_\otimes \cap \operatorname*{argmin}_{h\in\mathcal{H}} \ell_\gamma(h,\tilde{S}) \neq \emptyset]) \\
&\geq \frac{1}{2}\mathbb{P}_{S,\tilde{S}\sim D^{m/2}}[S_X \cup \tilde{S}_X \text{ is } \gamma\text{-shattered at the origin }].
\end{aligned}$$

The last inequality follows from the argument above regarding $h_1$ and $h_2$. The last expression is simply half the probability that a sample of size $m$ from $D_X$ is shattered. By assumption, this probability is at least $\eta$. Thus we conclude that $\mathbb{P}_{S\sim D^{m/2}}[\ell_0(\mathcal{A}(S),D) \geq \frac{1}{2}] \geq \eta/2$. It follows that $\ell_0(\mathcal{A}_\gamma, D, m/2, \eta/2) \geq \frac{1}{2}$. ■

As a side note, it is interesting to observe that Theorem 13 does not hold in general for non-symmetric hypothesis classes. For example, assume that the domain is $\mathcal{X} = [0,1]$, and the hypothesis class is the set of all functions that label a finite number of points in $[0,1]$ by $+1$ and the rest by $-1$. Consider learning using MEM, when the distribution is uniform over $[0,1]$, and all the labels are $-1$. For any $m > 0$ and $\gamma \in (0,1)$, a sample of size $m$ is $\gamma$-shattered at the origin with probability 1. However, any learning algorithm that returns a hypothesis from the hypothesis class will incur zero error on this distribution. Thus, shattering alone does not suffice to ensure that learning is hard.

### 7.2 A Sample Complexity Lower Bound with Gram-Matrix Eigenvalues

We now return to the case of homogeneous linear classifiers, and link high-probability fat-shattering to properties of the distribution. First, we present an equivalent and simpler characterization of fat-shattering for linear classifiers. We then use it to provide a sufficient condition for the fat-shattering of a sample, based on the smallest eigenvalue of its Gram matrix.

**Theorem 14** *Let $\mathbb{X} \in \mathbb{R}^{m\times d}$ be the matrix of a set of size $m$ in $\mathbb{R}^d$. The set is $\gamma$-shattered at the origin by $\mathcal{W}$ if and only if $\mathbb{X}\mathbb{X}^T$ is invertible and for all $y \in \{\pm 1\}^m$, $y^T(\mathbb{X}\mathbb{X}^T)^{-1}y \leq \gamma^{-2}$.*

To prove Theorem 14 we require two auxiliary lemmas. The first lemma, stated below, shows that for convex function classes, $\gamma$-shattering can be substituted with shattering with exact $\gamma$-margins.

**Lemma 15** *Let $\mathcal{F} \subseteq \mathbb{R}^\mathcal{X}$ be a class of functions, and assume that $\mathcal{F}$ is convex, that is*

$$\forall f_1, f_2 \in \mathcal{F}, \forall \lambda \in [0,1], \quad \lambda f_1 + (1-\lambda) f_2 \in \mathcal{F}.$$

*If $S = \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ is $\gamma$-shattered by $\mathcal{F}$ with witness $r \in \mathbb{R}^m$, then for every $y \in \{\pm 1\}^m$ there is an $f \in \mathcal{F}$ such that for all $i \in [m]$, $y[i](f(x_i) - r[i]) = \gamma$.*

The proof of this lemma is provided in Appendix A.4. The second lemma that we use allows converting the representation of the Gram-matrix to a different feature space, while keeping the separation properties intact. For a matrix $\mathbb{M}$, denote its pseudo-inverse by $\mathbb{M}^+$.

**Lemma 16** *Let $\mathbb{X} \in \mathbb{R}^{m\times d}$ be a matrix such that $\mathbb{X}\mathbb{X}^T$ is invertible, and let $\mathbb{Y} \in \mathbb{R}^{m\times k}$ such that $\mathbb{X}\mathbb{X}^T = \mathbb{Y}\mathbb{Y}^T$. Let $r \in \mathbb{R}^m$ be some real vector. If there exists a vector $\widetilde{w} \in \mathbb{R}^k$ such that $\mathbb{Y}\widetilde{w} = r$, then there exists a vector $w \in \mathbb{R}^d$ such that $\mathbb{X}w = r$ and $\|w\| = \|\mathbb{Y}^T(\mathbb{Y}^T)^+\widetilde{w}\| \leq \|\widetilde{w}\|$.*

**Proof** Denote $\mathbb{K} = \mathbb{X}\mathbb{X}^T = \mathbb{Y}\mathbb{Y}^T$. Let $\mathbb{S} = \mathbb{Y}^T\mathbb{K}^{-1}\mathbb{X}$ and let $w = \mathbb{S}^T\widetilde{w}$. We have $\mathbb{X}w = \mathbb{X}\mathbb{S}^T\widetilde{w} = \mathbb{X}\mathbb{X}^T\mathbb{K}^{-1}\mathbb{Y}\widetilde{w} = \mathbb{Y}\widetilde{w} = r$. In addition, $\|w\|^2 = w^Tw = \widetilde{w}^T\mathbb{S}\mathbb{S}^T\widetilde{w}$. By definition of $\mathbb{S}$,

$$\mathbb{S}\mathbb{S}^T = \mathbb{Y}^T\mathbb{K}^{-1}\mathbb{X}\mathbb{X}^T\mathbb{K}^{-1}\mathbb{Y} = \mathbb{Y}^T\mathbb{K}^{-1}\mathbb{Y} = \mathbb{Y}^T(\mathbb{Y}\mathbb{Y}^T)^{-1}\mathbb{Y} = \mathbb{Y}^T(\mathbb{Y}^T)^+.$$

Denote $\mathbb{O} = \mathbb{Y}^T(\mathbb{Y}^T)^+$. $\mathbb{O}$ is an orthogonal projection matrix: by the properties of the pseudo-inverse, $\mathbb{O} = \mathbb{O}^T$ and $\mathbb{O}^2 = \mathbb{O}$. Therefore $\|w\|^2 = \widetilde{w}^T\mathbb{S}\mathbb{S}^T\widetilde{w} = \widetilde{w}^T\mathbb{O}\widetilde{w} = \widetilde{w}^T\mathbb{O}\mathbb{O}^T\widetilde{w} = \|\mathbb{O}\widetilde{w}\|^2 \leq \|\widetilde{w}\|^2$. ■

**Proof** [of Theorem 14] We prove the theorem for 1-shattering. The case of $\gamma$-shattering follows by rescaling $X$ appropriately. Let $\mathbb{X}\mathbb{X}^T = \mathbb{U}\Lambda\mathbb{U}^T$ be the SVD of $\mathbb{X}\mathbb{X}^T$, where $\mathbb{U}$ is an orthogonal matrix and $\Lambda$ is a diagonal matrix. Let $\mathbb{Y} = \mathbb{U}\Lambda^{\frac{1}{2}}$. We have $\mathbb{X}\mathbb{X}^T = \mathbb{Y}\mathbb{Y}^T$. We show that the specified conditions are sufficient and necessary for the shattering of the set:

1. Sufficient: If $\mathbb{X}\mathbb{X}^T$ is invertible, then $\Lambda$ is invertible, thus so is $\mathbb{Y}$. For any $y \in \{\pm 1\}^m$, Let $w_y = \mathbb{Y}^{-1}y$. Then $\mathbb{Y}w_y = y$. By Lemma 16, there exists a separator $w$ such that $\mathbb{X}w = y$ and $\|w\| \leq \|w_y\| = \sqrt{y^T(\mathbb{Y}\mathbb{Y}^T)^{-1}y} = \sqrt{y^T(\mathbb{X}\mathbb{X}^T)^{-1}y} \leq 1$.
2. Necessary: If $\mathbb{X}\mathbb{X}^T$ is not invertible then the vectors in $S$ are linearly dependent, thus $S$ cannot be shattered using linear separators (see, e.g., Vapnik, 1995). The first condition is therefore necessary. Assume $S$ is 1-shattered at the origin and show that the second condition necessarily holds. By Lemma 15, for all $y \in \{\pm 1\}^m$ there exists a $w_y \in \mathbb{B}_1^d$ such that $\mathbb{X}w_y = y$. Thus by Lemma 16 there exists a $\widetilde{w}_y$ such that $\mathbb{Y}\widetilde{w}_y = y$ and $\|\widetilde{w}_y\| \leq \|w_y\| \leq 1$. $\mathbb{X}\mathbb{X}^T$ is invertible, thus so is $\mathbb{Y}$. Therefore $\widetilde{w}_y = \mathbb{Y}^{-1}y$. Thus $y^T(\mathbb{X}\mathbb{X}^T)^{-1}y = y^T(\mathbb{Y}\mathbb{Y}^T)^{-1}y = \|\widetilde{w}_y\| \leq 1$.

■

We are now ready to provide a sufficient condition for fat-shattering based on the smallest eigenvalue of the Gram matrix.

**Corollary 17** *Let $\mathbb{X} \in \mathbb{R}^{m\times d}$ be the matrix of a set of size $m$ in $\mathbb{R}^d$. If $\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \geq m\gamma^2$ then the set is $\gamma$-shattered at the origin by $\mathcal{W}$.*

**Proof** If $\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \geq m\gamma^2$ then $\mathbb{X}\mathbb{X}^T$ is invertible and $\lambda_{\max}((\mathbb{X}\mathbb{X}^T)^{-1}) \leq (m\gamma^2)^{-1}$. For any $y \in \{\pm 1\}^m$ we have $\|y\| = \sqrt{m}$ and

$$y^T(\mathbb{X}\mathbb{X}^T)^{-1}y \leq \|y\|^2\lambda_{\max}((\mathbb{X}\mathbb{X}^T)^{-1}) \leq m(m\gamma^2)^{-1} = \gamma^{-2}.$$

By Theorem 14 the sample is $\gamma$-shattered at the origin. ■

Corollary 17 generalizes the requirement of linear independence for shattering with no margin: A set of vectors is shattered with no margin if the vectors are linearly independent, that is if $\lambda_{\min} > 0$. The corollary shows that for $\gamma$-fat-shattering, we can require instead $\lambda_{\min} \geq m\gamma^2$. We can now

conclude that if it is highly probable that the smallest eigenvalue of the sample Gram matrix is large, then MEM might fail to learn a good classifier for the given distribution. This is formulated in the following theorem.

**Theorem 18** *Let $D$ be a distribution over $\mathbb{R}^d \times \{\pm 1\}$. Let $m > 0$ and let $\mathbb{X}$ be the matrix of a sample drawn from $D_X^m$. Let $\eta = \mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \geq m\gamma^2]$. Then for all $\varepsilon < 1/2 - \ell_\gamma^*(D)$, $m(\varepsilon, \gamma, D, \eta/2) \geq \lfloor m/2 \rfloor$.*

The proof of the theorem is immediate by combining Theorem 13 and Corollary 17.

Theorem 18 generalizes the case of learning a linear separator without a margin: If a sample of size $m$ is linearly independent with high probability, then there is no hope of using $m/2$ points to predict the label of the other points. The theorem extends this observation to the case of learning with a margin, by requiring a stronger condition than just linear independence of the points in the sample.

Recall that our upper-bound on the sample complexity from Section 6 is $\tilde{O}(k_\gamma)$. We now define the family of sub-Gaussian product distributions, and show that for this family, the lower bound that can be deduced from Theorem 18 is also linear in $k_\gamma$.

### 7.3 Sub-Gaussian Distributions

In order to derive a lower bound on distribution-specific sample complexity in terms of the covariance of $X \sim D_X$, we must assume that $X$ is not too heavy-tailed. This is because for any data distribution there exists another distribution which is almost identical and has the same sample complexity, but has arbitrarily large covariance values. This can be achieved by mixing the original distribution with a tiny probability for drawing a vector with a huge norm. We thus restrict the discussion to multidimensional sub-Gaussian distributions. This ensures light tails of the distribution in all directions, while still allowing a rich family of distributions, as we presently see. Sub-Gaussianity is defined for scalar random variables as follows (see, e.g., Buldygin and Kozachenko, 1998).

**Definition 19 (Sub-Gaussian random variables)** *A random variable $X \in \mathbb{R}$ is* sub-Gaussian with moment *$B$, for $B \geq 0$, if*

$$\forall t \in \mathbb{R}, \quad \mathbb{E}[\exp(tX)] \leq \exp(t^2 B^2/2).$$

*In this work we further say that $X$ is sub-Gaussian with* relative moment *$\rho > 0$ if $X$ is sub-Gaussian with moment $\rho\sqrt{\mathbb{E}[X^2]}$, that is,*

$$\forall t \in \mathbb{R}, \quad \mathbb{E}[\exp(tX)] \leq \exp(t^2 \rho^2 \mathbb{E}[X^2]/2).$$

Note that a sub-Gaussian variable with moment $B$ and relative moment $\rho$ is also sub-Gaussian with moment $B'$ and relative moment $\rho'$ for any $B' \geq B$ and $\rho' \geq \rho$.

The family of sub-Gaussian distributions is quite extensive: For instance, it includes any bounded, Gaussian, or Gaussian-mixture random variable with mean zero. Specifically, if $X$ is a mean-zero Gaussian random variable, $X \sim N(0, \sigma^2)$, then $X$ is sub-Gaussian with relative moment 1 and the inequalities in the definition above hold with equality. As another example, if $X$ is a uniform random variable over $\{\pm b\}$ for some $b \geq 0$, then $X$ is sub-Gaussian with relative moment 1, since

$$\mathbb{E}[\exp(tX)] = \frac{1}{2}(\exp(tb) + \exp(-tb)) \leq \exp(t^2 b^2/2) = \exp(t^2 \mathbb{E}[X^2]/2). \tag{9}$$

Let $\mathbb{B} \in \mathbb{R}^{d\times d}$ be a symmetric PSD matrix. A random vector $X \in \mathbb{R}^d$ is a *sub-Gaussian random vector* with moment matrix $\mathbb{B}$ if for all $u \in \mathbb{R}^d$, $\mathbb{E}[\exp(\langle u, X\rangle)] \leq \exp(\langle \mathbb{B}u, u\rangle/2)$. The following lemma provides a useful connection between the trace of the sub-Gaussian moment matrix and the moment-generating function of the squared norm of the random vector. The proof is given in Appendix A.5.

**Lemma 20** *Let $X \in \mathbb{R}^d$ be a sub-Gaussian random vector with moment matrix $\mathbb{B}$. Then for all $t \in (0, \frac{1}{4\lambda_{\max}(\mathbb{B})}]$, $\mathbb{E}[\exp(t\|X\|^2)] \leq \exp(2t \cdot \mathrm{trace}(\mathbb{B}))$.*

Our lower bound holds for the family of sub-Gaussian product distributions, defined as follows.

**Definition 21 (Sub-Gaussian product distributions)** *A distribution $D_X$ over $\mathbb{R}^d$ is a* sub-Gaussian product distribution *with moment $B$ and relative moment $\rho$ if there exists some orthonormal basis $a_1, \ldots, a_d \in \mathbb{R}^d$, such that for $X \sim D_X$, $\langle a_i, X\rangle$ are independent sub-Gaussian random variables, each with moment $B$ and relative moment $\rho$.*

Note that a sub-Gaussian product distribution has mean zero, thus its covariance matrix is equal to its uncentered covariance matrix. For any fixed $\rho \geq 0$, we denote by $\mathcal{D}^{sg}_\rho$ the family of all sub-Gaussian product distributions with relative moment $\rho$, in arbitrary dimension. For instance, all multivariate Gaussian distributions and all uniform distributions on the corners of a centered hyper-rectangle are in $\mathcal{D}^{sg}_1$. All uniform distributions over a full centered hyper-rectangle are in $\mathcal{D}^{sg}_{3/2}$. Note that if $\rho_1 \leq \rho_2$, $\mathcal{D}^{sg}_{\rho_1} \subseteq \mathcal{D}^{sg}_{\rho_2}$.

We will provide a lower bound for all distributions in $\mathcal{D}^{sg}_\rho$. This lower bound is linear in the margin-adapted dimension of the distribution, thus it matches the upper bound provided in Corollary 12. The constants in the lower bound depend only on the value of $\rho$, which we regard as a constant.

### 7.4 A Sample-Complexity Lower Bound for Sub-Gaussian Product Distributions

As shown in Section 7.2, to obtain a sample complexity lower bound it suffices to have a lower bound on the value of the smallest eigenvalue of a random Gram matrix. The distribution of the smallest eigenvalue of a random Gram matrix has been investigated under various assumptions. The cleanest results are in the asymptotic case where the sample size and the dimension approach infinity, the ratio between them approaches a constant, and the coordinates of each example are identically distributed.

**Theorem 22 (Bai and Silverstein 2010, Theorem 5.11)** *Let $\{\mathbb{X}_i\}_{i=1}^\infty$ be a series of matrices of sizes $m_i \times d_i$, whose entries are i.i.d. random variables with mean zero, variance $\sigma^2$ and finite fourth moments. If $\lim_{i\to\infty} \frac{m_i}{d_i} = \beta < 1$, then $\lim_{i\to\infty} \lambda_{\min}(\frac{1}{d_i}\mathbb{X}_i\mathbb{X}_i^T) = \sigma^2(1-\sqrt{\beta})^2$.*

This asymptotic limit can be used to approximate an asymptotic lower bound on $m(\varepsilon, \gamma, D)$, if $D_X$ is a product distribution of i.i.d. random variables with mean zero, variance $\sigma^2$, and finite fourth moment. Let $\mathbb{X} \in \mathbb{R}^{m\times d}$ be the matrix of a sample of size $m$ drawn from $D_X$. We can find $m = m_\circ$ such that $\lambda_{m_\circ}(\mathbb{X}\mathbb{X}^T) \approx \gamma^2 m_\circ$, and use Theorem 18 to conclude that $m(\varepsilon, \gamma, D) \geq m_\circ/2$. If $d$ and $m$ are large enough, we have by Theorem 22 that for $\mathbb{X}$ drawn from $D_X^m$:

$$\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \approx d\sigma^2(1-\sqrt{m/d})^2 = \sigma^2(\sqrt{d}-\sqrt{m})^2.$$

Solving the equality $\sigma^2(\sqrt{d}-\sqrt{m_\circ})^2 = m_\circ\gamma^2$ we get $m_\circ = d/(1+\gamma/\sigma)^2$. The margin-adapted dimension for $D_X$ is $k_\gamma \approx d/(1+\gamma^2/\sigma^2)$, thus $\frac{1}{2}k_\gamma \le m_\circ \le k_\gamma$. In this case, then, the sample complexity lower bound is indeed the same order as $k_\gamma$, which controls also the upper bound in Corollary 12. However, this is an asymptotic analysis, which holds for a highly limited set of distributions. Moreover, since Theorem 22 holds asymptotically for each distribution separately, we cannot use it to deduce a uniform finite-sample lower bound for families of distributions.

For our analysis we require *finite-sample* bounds for the smallest eigenvalue of a random Gram-matrix. Rudelson and Vershynin (2009, 2008) provide such finite-sample lower bounds for distributions which are products of identically distributed sub-Gaussians. In Theorem 23 below we provide a new and more general result, which holds for any sub-Gaussian product distribution. The proof of Theorem 23 is provided in Appendix A.6. Combining Theorem 23 with Theorem 18 above we prove the lower bound, stated in Theorem 24 below.

**Theorem 23** *For any $\rho > 0$ and $\delta \in (0,1)$ there are $\beta > 0$ and $C > 0$ such that the following holds. For any $D_X \in \mathcal{D}_\rho^{sg}$ with covariance matrix $\Sigma \le I$, and for any $m \le \beta \cdot \mathrm{trace}(\Sigma) - C$, if $\mathbb{X}$ is the $m \times d$ matrix of a sample drawn from $D_X^m$, then*

$$\mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \ge m] \ge \delta.$$

**Theorem 24 (Sample complexity lower bound for distributions in $\mathcal{D}_\rho^{\mathbf{sg}}$)** *For any $\rho > 0$ there are constants $\beta > 0, C \ge 0$ such that for any $D$ with $D_X \in \mathcal{D}_\rho^{sg}$, for any $\gamma > 0$ and for any $\varepsilon < \frac{1}{2} - \ell_\gamma^*(D)$,*

$$m(\varepsilon, \gamma, D, 1/4) \ge \beta k_\gamma(D_X) - C.$$

**Proof** Assume w.l.o.g. that the orthonormal basis $a_1, \ldots, a_d$ of independent sub-Gaussian directions of $D_X$, defined in Definition 21, is the natural basis $e_1, \ldots, e_d$. Define $\lambda_i = \mathbb{E}_{X \sim D_X}[X[i]^2]$, and assume w.l.o.g. $\lambda_1 \ge \ldots \ge \lambda_d > 0$. Let $\mathbb{X}$ be the $m \times d$ matrix of a sample drawn from $D_X^m$. Fix $\delta \in (0,1)$, and let $\beta$ and $C$ be the constants for $\rho$ and $\delta$ in Theorem 23. Throughout this proof we abbreviate $k_\gamma \triangleq k_\gamma(D_X)$. Let $m \le \beta(k_\gamma - 1) - C$. We would like to use Theorem 23 to bound $\lambda_{\min}(\mathbb{X}\mathbb{X}^T)$ with high probability, so that Theorem 18 can be applied to get the desired lower bound. However, Theorem 23 holds only if $\Sigma \le I$. Thus we split to two cases—one in which the dimensionality controls the lower bound, and one in which the norm controls it. The split is based on the value of $\lambda_{k_\gamma}$.

- Case I: Assume $\lambda_{k_\gamma} \ge \gamma^2$. Then $\forall i \in [k_\gamma], \lambda_i \ge \gamma^2$. By our assumptions on $D_X$, for all $i \in [d]$ the random variable $X[i]$ is sub-Gaussian with relative moment $\rho$. Consider the random variables $Z[i] = X[i]/\sqrt{\lambda_i}$ for $i \in [k_\gamma]$. $Z[i]$ is also sub-Gaussian with relative moment $\rho$, and $\mathbb{E}[Z[i]^2] = 1$. Consider the product distribution of $Z[1], \ldots, Z[k_\gamma]$, and let $\Sigma'$ be its covariance matrix. We have $\Sigma' = I_{k_\gamma}$, and $\mathrm{trace}(\Sigma') = k_\gamma$. Let $\mathbb{Z}$ be the matrix of a sample of size $m$ drawn from this distribution. By Theorem 23, $\mathbb{P}[\lambda_{\min}(\mathbb{Z}\mathbb{Z}^T) \ge m] \ge \delta$, which is equivalent to

  $$\mathbb{P}[\lambda_{\min}(\mathbb{X} \cdot \mathrm{diag}(1/\lambda_1, \ldots, 1/\lambda_{k_\gamma}, 0, \ldots, 0) \cdot \mathbb{X}^T) \ge m] \ge \delta.$$

  Since $\forall i \in [k_\gamma], \lambda_i \ge \gamma^2$, we have $\mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \ge m\gamma^2] \ge \delta$.

- Case II: Assume $\lambda_{k_\gamma} < \gamma^2$. Then $\lambda_i < \gamma^2$ for all $i \in \{k_\gamma, \ldots, d\}$. Consider the random variables $Z[i] = X[i]/\gamma$ for $i \in \{k_\gamma, \ldots, d\}$. $Z[i]$ is sub-Gaussian with relative moment $\rho$ and $\mathbb{E}[Z[i]^2] \le 1$.

Consider the product distribution of $Z[k_\gamma],\ldots,Z[d]$, and let $\Sigma'$ be its covariance matrix. We have $\Sigma' < I_{d-k_\gamma+1}$. By the minimality in Equation (4) we also have $\mathrm{trace}(\Sigma') = \frac{1}{\gamma^2}\sum_{i=k_\gamma}^{d}\lambda_i \geq k_\gamma - 1$. Let $\mathbb{Z}$ be the matrix of a sample of size $m$ drawn from this product distribution. By Theorem 23, $\mathbb{P}[\lambda_{\min}(\mathbb{Z}\mathbb{Z}^T) \geq m] \geq \delta$. Equivalently,

$$\mathbb{P}[\lambda_{\min}(\mathbb{X}\cdot \mathrm{diag}(0,\ldots,0,1/\gamma^2,\ldots,1/\gamma^2)\cdot \mathbb{X}^T) \geq m] \geq \delta,$$

therefore $\mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \geq m\gamma^2] \geq \delta$.

In both cases $\mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \geq m\gamma^2] \geq \delta$. This holds for any $m \leq \beta(k_\gamma - 1) - C$, thus by Theorem 18 $m(\varepsilon,\gamma,D,\delta/2) \geq \lfloor(\beta(k_\gamma-1)-C)/2\rfloor$ for $\varepsilon < 1/2 - \ell^*_\gamma(D)$. We finalize the proof by setting $\delta = \frac{1}{2}$ and adjusting $\beta$ and $C$. ■

## 8. On the Limitations of the Covariance Matrix

We have shown matching upper and lower bounds for the sample complexity of learning with MEM, for any sub-Gaussian product distribution with a bounded relative moment. This shows that the margin-adapted dimension fully characterizes the sample complexity of learning with MEM for such distributions. What properties of a distribution play a role in determining the sample complexity for general distributions? In the following theorem we show that these properties must include more than the covariance matrix of the distribution, even when assuming sub-Gaussian tails and bounded relative moments.

**Theorem 25** *For any integer $d > 1$, there exist two distributions $D$ and $P$ over $\mathbb{R}^d \times \{\pm 1\}$ with identical covariance matrices, such that for any $\varepsilon,\delta \in (0,\frac{1}{4})$, $m(\varepsilon,1,P,\delta) \geq \Omega(d)$ while $m(\varepsilon,1,D,\delta) \leq \lceil \log_2(1/\delta)\rceil$. Both $D_X$ and $P_X$ are sub-Gaussian random vectors, with a relative moment of $\sqrt{2}$ in all directions.*

**Proof** Let $D_a$ and $D_b$ be distributions over $\mathbb{R}^d$ such that $D_a$ is uniform over $\{\pm 1\}^d$ and $D_b$ is uniform over $\{\pm 1\}\times\{0\}^{d-1}$. Let $D_X$ be a balanced mixture of $D_a$ and $D_b$. Let $P_X$ be uniform over $\{\pm 1\}\times\{\frac{1}{\sqrt{2}}\}^{d-1}$. For both $D$ and $P$, let $\mathbb{P}[Y = \langle e_1, X\rangle] = 1$. The covariance matrix of $D_X$ and $P_X$ is $\mathrm{diag}(1,\frac{1}{2},\ldots,\frac{1}{2})$, thus $k_1(D_X) = k_1(P_X) \geq \Omega(d)$.

By Equation (9), $P_X, D_a$ and $D_b$ are all sub-Gaussian product distribution with relative moment 1, thus also with moment $\sqrt{2} > 1$. The projection of $D_X$ along any direction $u \in \mathbb{R}^d$ is sub-Gaussian with relative moment $\sqrt{2}$ as well, since

$$\begin{aligned}
&\mathbb{E}_{X\sim D_X}[\exp(\langle u,X\rangle)] = \frac{1}{2}(\mathbb{E}_{X\sim D^a}[\exp(\langle u,X\rangle)] + \mathbb{E}_{X\sim D^b}[\exp(\langle u,X\rangle)]) \\
&= \frac{1}{2}(\prod_{i\in[d]}(\exp(u_i)+\exp(-u_i))/2 + (\exp(u_1)+\exp(-u_1))/2) \\
&\leq \frac{1}{2}(\prod_{i\in[d]}\exp(u_i^2/2) + \exp(u_1^2/2)) \leq \exp(\|u\|^2/2) \leq \exp((\|u\|^2+u_1^2)/2) \\
&= \exp(\mathbb{E}_{X\sim D_X}[\langle u,X\rangle^2]).
\end{aligned}$$

For $P$ we have by Theorem 24 that for any $\varepsilon \leq \frac{1}{4}$, $m(\varepsilon, 1, P, \frac{1}{4}) \geq \Omega(k_1(P_X)) \geq \Omega(d)$. In contrast, any MEM algorithm $\mathcal{A}_1$ will output the correct separator for $D$ whenever the sample has at least one point drawn from $D_b$. This is because the separator $e_1$ is the only $w \in \mathbb{B}_1^d$ that classifies this point with zero 1-margin errors. Such a point exists in a sample of size $m$ with probability $1 - 2^{-m}$. Therefore $\ell_0(\mathcal{A}_1, D, m, 1/2^m) = 0$. It follows that for all $\varepsilon > 0$, $m(\varepsilon, 1, D, \delta) \leq \lceil \log_2(1/\delta) \rceil$. ■

## 9. Conclusions

Corollary 12 and Theorem 24 together provide a tight characterization of the sample complexity of any sub-Gaussian product distribution with a bounded relative moment. Formally, fix $\rho > 0$. For any $D$ such that $D_X \in \mathcal{D}_\rho^{\text{sg}}$, and for any $\gamma > 0$ and $\varepsilon \in (0, \frac{1}{2} - \ell_\gamma^*(D))$

$$\Omega(k_\gamma(D_X)) \leq m(\varepsilon, \gamma, D) \leq \tilde{O}\left(\frac{k_\gamma(D_X)}{\varepsilon^2}\right). \tag{10}$$

The upper bound holds uniformly for all distributions, and the constants in the lower bound depend only on $\rho$. This result shows that the true sample complexity of learning each of these distributions with MEM is characterized by the margin-adapted dimension. An interesting conclusion can be drawn as to the influence of the conditional distribution of labels $D_{Y|X}$: Since Equation (10) holds for *any* $D_{Y|X}$, the effect of the direction of the best separator on the sample complexity is bounded, even for highly non-spherical distributions.

We note that the upper bound that we have proved involves logarithmic factors which might not be necessary. There are upper bounds that depend on the margin alone and on the dimension alone without logarithmic factors. On the other hand, in our bound, which combines the two quantities, there is a logarithmic dependence which stems from the margin component of the bound. It might be possible to tighten the bound and remove the logarithmic dependence.

Equation (10) can be used to easily characterize the sample complexity behavior for interesting distributions, to compare $L_2$ margin minimization to other learning methods, and to improve certain active learning strategies. We elaborate on each of these applications in the following examples.

**Example 1 (Gaps between $L_1$ and $L_2$ regularization in the presence of irrelevant features)** *Ng (2004) considers learning a single relevant feature in the presence of many irrelevant features, and compares using $L_1$ regularization and $L_2$ regularization. When $\|X\|_\infty \leq 1$, upper bounds on learning with $L_1$ regularization guarantee a sample complexity of $O(\ln(d))$ for an $L_1$-based learning rule (Zhang, 2002). In order to compare this with the sample complexity of $L_2$ regularized learning and establish a gap, one must use a* lower bound *on the $L_2$ sample complexity. The argument provided by Ng actually assumes scale-invariance of the learning rule, and is therefore valid only for* unregularized *linear learning. In contrast, using our results we can easily establish a lower bound of $\Omega(d)$ for many specific distributions with a bounded $\|X\|_\infty$ and $Y = sign(X[i])$ for some $i$. For instance, if each coordinate is a bounded independent sub-Gaussian random variable with a bounded relative moment, we have $k_1 = \lceil d/2 \rceil$ and Theorem 24 implies a lower bound of $\Omega(d)$ on the $L_2$ sample complexity.*

**Example 2 (Gaps between generative and discriminative learning for a Gaussian mixture)** *Let there be two classes, each drawn from a unit-variance spherical Gaussian in $\mathbb{R}^d$ with a large distance $2v >> 1$ between the class means, such that $d >> v^4$. Then $\mathbb{P}_D[X|Y = y] = \mathcal{N}(yv \cdot e_1, I_d)$,*

*where $e_1$ is a unit vector in $\mathbb{R}^d$. For any $v$ and $d$, we have $D_X \in \mathcal{D}_1^{sg}$. For large values of $v$, we have extremely low margin error at $\gamma = v/2$, and so we can hope to learn the classes by looking for a large-margin separator. Indeed, we can calculate $k_\gamma = \lceil d/(1+\frac{v^2}{4}) \rceil$, and conclude that the required sample complexity is $\tilde{\Theta}(d/v^2)$. Now consider a generative approach: fitting a spherical Gaussian model for each class. This amounts to estimating each class center as the empirical average of the points in the class, and classifying based on the nearest estimated class center. It is possible to show that for any constant $\varepsilon > 0$, and for large enough $v$ and $d$, $O(d/v^4)$ samples are enough in order to ensure an error of $\varepsilon$. This establishes a rather large gap of $\Omega(v^2)$ between the sample complexity of the discriminative approach and that of the generative one.*

**Example 3 (Active learning)** *In active learning, there is an abundance of unlabeled examples, but labels are costly, and the active learning algorithm needs to decide which labels to query based on the labels seen so far. A popular approach to active learning involves estimating the current set of possible classifiers using sample complexity upper bounds (see, e.g., Balcan et al., 2009; Beygelzimer et al., 2010). Without any distribution-specific information, only general distribution-free upper bounds can be used. However, since there is an abundance of unlabeled examples, the active learner can use these to estimate tighter distribution-specific upper bounds. In the case of linear classifiers, the margin-adapted dimension can be calculated from the uncentered covariance matrix of the distribution, which can be easily estimated from unlabeled data. Thus, our sample complexity upper bounds can be used to improve the active learner's label complexity. Moreover, the lower bound suggests that any further improvement of such active learning strategies would require more information other than the distribution's covariance matrix.*

To summarize, we have shown that the true sample complexity of large-margin learning of each of a rich family of distributions is characterized by the margin-adapted dimension. Characterizing the true sample complexity allows a better comparison between this learning approach and other algorithms, and has many potential applications. The challenge of characterizing the true sample complexity extends to any distribution and any learning approach. Theorem 25 shows that other properties but the covariance matrix must be taken into account for general distributions. We believe that obtaining answers to these questions is of great importance, both to learning theory and to learning applications.

## Acknowledgments

The authors thank Boaz Nadler for many insightful discussions. During part of this research, Sivan Sabato was supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities. This work is partly supported by the Gatsby Charitable Foundation, The DARPA MSEE project, the Intel ICRI-CI center, and the Israel Science Foundation center of excellence grant.

## Appendix A. Proofs Omitted from the Text

In this appendix we give detailed proofs which were omitted from the text.

### A.1 Proof of Proposition 3

**Proof** Let $w^* \in \operatorname{argmin}_{w \in \mathbb{B}_1^d} \ell_\gamma(w, D)$. By Equation (2), with probability $1 - \delta/2$

$$\operatorname{ramp}_\gamma(\mathcal{A}_\gamma(S), D) \leq \operatorname{ramp}_\gamma(\mathcal{A}_\gamma(S), S) + 2\mathcal{R}_m(\text{RAMP}_\gamma, D) + \sqrt{\frac{8\ln(2/\delta)}{m}}.$$

Set $h^* \in \mathcal{H}$ such that $\ell_\gamma(h^*, D) = \ell_\gamma^*(\mathcal{H}, D)$. We have

$$\operatorname{ramp}_\gamma(\mathcal{A}_\gamma(S), S) \leq \ell_\gamma(\mathcal{A}_\gamma(S), S) \leq \ell_\gamma(h^*, S).$$

The first inequality follows since the ramp loss is upper bounded by the margin loss. The second inequality follows since $\mathcal{A}$ is a MEM algorithm. Now, by Hoeffding's inequality, since the range of $\operatorname{ramp}_\gamma$ is in $[0,1]$, with probability at least $1 - \delta/2$

$$\ell_\gamma(h^*, S) \leq \ell_\gamma(h^*, D) + \sqrt{\frac{\ln(2/\delta)}{2m}}.$$

It follows that with probability $1 - \delta$

$$\operatorname{ramp}_\gamma(\mathcal{A}_\gamma(S), D) \leq \ell_\gamma^*(\mathcal{H}, D) + 2\mathcal{R}_m(\text{RAMP}_\gamma, D) + \sqrt{\frac{14\ln(2/\delta)}{m}}. \tag{11}$$

We have $\ell_0 \leq \operatorname{ramp}_\gamma$. Combining this with Equation (11) we conclude Equation (3). ■

### A.2 Proof of Lemma 9

**Proof** [of Lemma 9] For a function $f : \mathcal{X} \to \mathbb{R}$ and a $z \in Z$, define the function $G[f,z]$ by

$$\forall x \in \mathcal{X}, \quad G[f,z](x) = [\![ f(x) + z(x) ]\!] - f(x).$$

Let $f_1, f_2 \in \mathbb{R}^{\mathcal{X}}$ be two functions, and let $g_1 = G[f_1, z] \in \mathcal{G}(f_1)$ for some $w_b \in \bar{V}$. Then, since $G[f_2, z] \in \mathcal{G}(f_2)$, we have

$$\inf_{g_2 \in \mathcal{G}(f_2)} \|g_1 - g_2\|_{L_2(S)} \leq \|G[f_1, z] - G[f_2, z]\|.$$

Now, for all $x \in \mathbb{R}$,

$$\begin{aligned} |G[f_1,z](x) - G[f_2,z](x)| &= |[\![ f_1(x) + z(x) ]\!] - f_1(x) - [\![ f_2(x) + z(x) ]\!] + f_2(x)| \\ &\leq |f_1(x) - f_2(x)|. \end{aligned}$$

Thus, for any $S \subseteq \mathcal{X}$,

$$\begin{aligned} \|G[f_1,z] - G[f_2,z]\|_{L_2(S)}^2 &= \mathbb{E}_{X \sim S}(G[f_1,z](X) - G[f_2,z](X))^2 \\ &\leq \mathbb{E}_{X \sim S}(f_1(X) - f_2(X))^2 = \|f_1 - f_2\|_{L_2(S)}^2. \end{aligned}$$

It follows that $\inf_{g_2 \in \mathcal{G}(f_2)} \|g_1 - g_2\|_{L_2(S)} \leq \|f_1 - f_2\|_{L_2(S)}$. This holds for any $g_1 \in \mathcal{G}(f_1)$, thus $\Delta_H(\mathcal{G}(f_1), \mathcal{G}(f_2)) \leq \|f_1 - f_2\|_{L_2(S)}$. ■

### A.3 Proof of Lemma 10

**Proof** [of Lemma 10] Let $k$ be the pseudo-dimension of $\mathcal{G}(f)$, and let $\{x_1,\ldots,x_k\} \subseteq \mathcal{X}$ be a set which is pseudo-shattered by $\mathcal{G}(f)$. We show that the same set is pseudo-shattered by $Z$ as well, thus proving the lemma. Since $\mathcal{G}(f)$ is pseudo-shattered, there exists a vector $r \in \mathbb{R}^k$ such that for all $y \in \{\pm 1\}^k$ there exists a $g_y \in \mathcal{G}(f)$ such that $\forall i \in [m], \text{sign}(g_y(x_i) - r[i]) = y[i]$. Therefore for all $y \in \{\pm 1\}^k$ there exists a $z_y \in Z$ such that

$$\forall i \in [k], \text{sign}(\llbracket f(x_i) + z_y(x_i) \rrbracket - f(x_i) - r[i]) = y[i].$$

By considering the case $y[i] = 1$, we have

$$0 < \llbracket f(x_i) + z_y(x_i) \rrbracket - f(x_i) - r[i] \leq 1 - f(x_i) - r[i].$$

By considering the case $y[i] = -1$, we have

$$0 > \llbracket f(x_i) + z_y(x_i) \rrbracket - f(x_i) - r[i] \geq -f(x_i) - r[i].$$

Therefore $0 < f(x_i) + r[i] < 1$. Now, let $y \in \{\pm 1\}^k$ and consider any $i \in [k]$. If $y[i] = 1$ then

$$\llbracket f(x_i) + z_y(x_i) \rrbracket - f(x_i) - r[i] > 0$$

It follows that

$$\llbracket f(x_i) + z_y(x_i) \rrbracket > f(x_i) + r[i] > 0,$$

thus

$$f(x_i) + z_y(x_i) > f(x_i) + r[i].$$

In other words, $\text{sign}(z_y(x_i) - r[i]) = 1 = y[i]$. If $y[i] = -1$ then

$$\llbracket f(x_i) + z_y(x_i) \rrbracket - f(x_i) - r[i] < 0.$$

It follows that

$$\llbracket f(x_i) + z_y(x_i) \rrbracket < f(x_i) + r[i] < 1,$$

thus

$$f(x_i) + z_y(x_i) < f(x_i) + r[i].$$

in other words, $\text{sign}(z_y(x_i) - r[i]) = -1 = y[i]$. We conclude that $Z$ shatters $\{x_1,\ldots,x_k\}$ as well, using the same vector $r \in \mathbb{R}^k$. Thus the pseudo-dimension of $Z$ is at least $k$. ■

### A.4 Proof of Lemma 15

To prove Lemma 15, we first prove the following lemma. Denote by $\text{conv}(A)$ the convex hull of a set $A$.

**Lemma 26** *Let $\gamma > 0$. For each $y \in \{\pm 1\}^m$, select $r_y \in \mathbb{R}^m$ such that for all $i \in [m]$, $r_y[i]y[i] \geq \gamma$. Let $R = \{r_y \in \mathbb{R}^m \mid y \in \{\pm 1\}^m\}$. Then $\{\pm\gamma\}^m \subseteq \text{conv}(R)$.*

**Proof** We will prove the claim by induction on the dimension $m$. For the base case, if $m = 1$, we have $R = \{a, b\} \subseteq \mathbb{R}$ where $a \leq -\gamma$ and $b \geq \gamma$. Clearly, $\text{conv}(R) = [a, b]$, and $\pm\gamma \in [a, b]$.

For the inductive step, assume the lemma holds for $m-1$. For a vector $t \in \mathbb{R}^m$, denote by $\bar{t}$ its projection $(t[1], \ldots, t[m-1])$ on $\mathbb{R}^{m-1}$. Similarly, for a set of vectors $S \subseteq \mathbb{R}^m$, let $\bar{S} = \{\bar{s} \mid s \in S\} \subseteq \mathbb{R}^{m-1}$. Define $Y_+ = \{\pm 1\}^{m-1} \times \{+1\}$ and $Y_- = \{\pm 1\}^{m-1} \times \{-1\}$. Let $R_+ = \{r_y \mid y \in Y_+\}$, and similarly for $R_-$. Then the induction hypothesis holds for $\bar{R}_+$ and $\bar{R}_-$ with dimension $m-1$. Let $z \in \{\pm\gamma\}^m$. We wish to prove $z \in \text{conv}(R)$. From the induction hypothesis we have $\bar{z} \in \text{conv}(\bar{R}_+)$ and $\bar{z} \in \text{conv}(\bar{R}_-)$. Thus, for all $y \in \{\pm 1\}$ there exist $\alpha_y, \beta_y \geq 0$ such that $\sum_{y \in Y_+} \alpha_y = \sum_{y \in Y_-} \beta_y = 1$, and

$$\bar{z} = \sum_{y \in Y_+} \alpha_y \bar{r}_y = \sum_{y \in Y_-} \beta_y \bar{r}_y.$$

Let $z_a = \sum_{y \in Y_+} \alpha_y r_y$ and $z_b = \sum_{y \in Y_-} \beta_y r_y$ We have that $\forall y \in Y_+, r_y[m] \geq \gamma$, and $\forall y \in Y_-, r_y[m] \leq -\gamma$. Therefore, $z_b[m] \leq -\gamma \leq z[m] \leq \gamma \leq z_a[m]$. In addition, $\bar{z}_a = \bar{z}_b = \bar{z}$. Select $\lambda \in [0, 1]$ such that $z[m] = \lambda z_a[m] + (1-\lambda) z_b[m]$, then $z = \lambda z_a + (1-\lambda) z_b$. Since $z_a, z_b \in \text{conv}(R)$, we have $z \in \text{conv}(R)$. ■

**Proof** [of Lemma 15] Denote by $f(S)$ the vector $(f(x_1), \ldots, f(x_m))$. Recall that $r \in \mathbb{R}^m$ is the witness for the shattering of $S$, and let

$$L = \{f(S) - r \mid f \in \mathcal{F}\} \subseteq \mathbb{R}^m.$$

Since $S$ is shattered, for any $y \in \{\pm 1\}^m$ there is an $r_y \in L$ such that $\forall i \in [m], r_y[i] y[i] \geq \gamma$. By Lemma 26, $\{\pm\gamma\}^m \subseteq \text{conv}(L)$. Since $\mathcal{F}$ is convex, $L$ is also convex. Therefore $\{\pm\gamma\}^m \subseteq L$. ■

### A.5 Proof of Lemma 20

**Proof** [of Lemma 20] It suffices to consider diagonal moment matrices: If $\mathbb{B}$ is not diagonal, let $\mathbb{V} \in \mathbb{R}^{d \times d}$ be an orthogonal matrix such that $\mathbb{V}\mathbb{B}\mathbb{V}^T$ is diagonal, and let $Y = \mathbb{V}X$. We have $\mathbb{E}[\exp(t\|Y\|^2)] = \mathbb{E}[\exp(t\|X\|^2)]$ and $\text{trace}(\mathbb{V}\mathbb{B}\mathbb{V}^T) = \text{trace}(\mathbb{B})$. In addition, for all $u \in \mathbb{R}^d$,

$$\mathbb{E}[\exp(\langle u, Y\rangle)] = \mathbb{E}[\exp(\langle \mathbb{V}^T u, X\rangle)] \leq \exp(\frac{1}{2}\langle \mathbb{B}\mathbb{V}^T u, \mathbb{V}^T u\rangle) = \exp(\frac{1}{2}\langle \mathbb{V}\mathbb{B}\mathbb{V}^T u, u\rangle).$$

Therefore $Y$ is sub-Gaussian with the diagonal moment matrix $\mathbb{V}\mathbb{B}\mathbb{V}^T$. Thus assume w.l.o.g. that $\mathbb{B} = \text{diag}(\lambda_1, \ldots, \lambda_d)$ where $\lambda_1 \geq \ldots \geq \lambda_d \geq 0$.

We have $\exp(t\|X\|^2) = \prod_{i\in[d]} \exp(tX[i]^2)$. In addition, for any $t > 0$ and $x \in \mathbb{R}$, $2\sqrt{\Pi t}\cdot\exp(tx^2) = \int_{-\infty}^{\infty}\exp(sx - \frac{s^2}{4t})ds$. Therefore, for any $u \in \mathbb{R}^d$,

$$\begin{aligned}
(2\sqrt{\Pi t})^d \cdot \mathbb{E}[\exp(t\|X\|^2)] &= \mathbb{E}\left[\prod_{i\in[d]}\int_{-\infty}^{\infty}\exp(u[i]X[i] - \frac{u[i]^2}{4t})du[i]\right] \\
&= \mathbb{E}\left[\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\prod_{i\in[d]}\exp(u[i]X[i] - \frac{u[i]^2}{4t})du[i]\right] \\
&= \mathbb{E}\left[\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\exp(\langle u, X\rangle - \frac{\|u\|^2}{4t})\prod_{i\in[d]}du[i]\right] \\
&= \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\mathbb{E}[\exp(\langle u, X\rangle)]\exp(-\frac{\|u\|^2}{4t})\prod_{i\in[d]}du[i]
\end{aligned}$$

By the sub-Gaussianity of $X$, the last expression is bounded by

$$\begin{aligned}
&\leq \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\exp(\frac{1}{2}\langle\mathbb{B}u, u\rangle - \frac{\|u\|^2}{4t})\prod_{i\in[d]}du[i] \\
&= \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\prod_{i\in[d]}\exp(\frac{\lambda_i u[i]^2}{2} - \frac{u[i]^2}{4t})du[i] \\
&= \prod_{i\in[d]}\int_{-\infty}^{\infty}\exp(u[i]^2(\frac{\lambda_i}{2} - \frac{1}{4t}))du[i] = \Pi^{d/2}\big(\prod_{i\in[d]}(\frac{1}{4t} - \frac{\lambda_i}{2})\big)^{-\frac{1}{2}}.
\end{aligned}$$

The last equality follows from the fact that for any $a > 0$, $\int_{-\infty}^{\infty}\exp(-a\cdot s^2)ds = \sqrt{\Pi/a}$, and from the assumption $t \leq \frac{1}{4\lambda_1}$. We conclude that

$$\mathbb{E}[\exp(t\|X\|^2)] \leq \big(\prod_{i\in[d]}(1 - 2\lambda_i t)\big)^{-\frac{1}{2}} \leq \exp(2t\cdot\sum_{i=1}^{d}\lambda_i) = \exp(2t\cdot\mathrm{trace}(\mathbb{B})),$$

where the second inequality holds since $\forall x \in [0,1]$, $(1 - x/2)^{-1} \leq \exp(x)$. ■

### A.6 Proof of Theorem 23

In the proof of Theorem 23 we use the fact $\lambda_{\min}(\mathbb{X}\mathbb{X}^T) = \inf_{\|x\|_2=1}\|\mathbb{X}^T x\|^2$ and bound the right-hand side via an $\varepsilon$-net of the unit sphere in $\mathbb{R}^m$, denoted by $S^{m-1} \triangleq \{x \in \mathbb{R}^m \mid \|x\|_2 = 1\}$. An $\varepsilon$-net of the unit sphere is a set $C \subseteq S^{m-1}$ such that $\forall x \in S^{m-1}, \exists x' \in C, \|x - x'\| \leq \varepsilon$. Denote the minimal size of an $\varepsilon$-net for $S^{m-1}$ by $\mathcal{N}_m(\varepsilon)$, and by $\mathcal{C}_m(\varepsilon)$ a minimal $\varepsilon$-net of $S^{m-1}$, so that $\mathcal{C}_m(\varepsilon) \subseteq S^{m-1}$ and $|\mathcal{C}_m(\varepsilon)| = \mathcal{N}_m(\varepsilon)$. The proof of Theorem 23 requires several lemmas. First we prove a concentration result for the norm of a matrix defined by sub-Gaussian variables. Then we bound the probability that the squared norm of a vector is small.

**Lemma 27** *Let $\mathbb{Y}$ be a $d \times m$ matrix with $m \leq d$, such that $\mathbb{Y}_{ij}$ are independent sub-Gaussian variables with moment $B$. Let $\Sigma$ be a diagonal $d \times d$ PSD matrix such that $\Sigma \leq I$. Then for all $t \geq 0$*

*and* $\varepsilon \in (0,1)$,

$$\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}\| \geq t] \leq \mathcal{N}_m(\varepsilon)\exp(\frac{\text{trace}(\Sigma)}{2} - \frac{t^2(1-\varepsilon)^2}{4B^2}).$$

**Proof** We have $\|\sqrt{\Sigma}\mathbb{Y}\| \leq \max_{x\in\mathcal{C}_m(\varepsilon)} \|\sqrt{\Sigma}\mathbb{Y}x\|/(1-\varepsilon)$, see for instance in Bennett et al. (1975). Therefore,

$$\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}\| \geq t] \leq \sum_{x\in\mathcal{C}_m(\varepsilon)} \mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\| \geq (1-\varepsilon)t]. \tag{12}$$

Fix $x \in \mathcal{C}_m(\varepsilon)$. Let $V = \sqrt{\Sigma}\mathbb{Y}x$, and assume $\Sigma = \text{diag}(\lambda_1,\ldots,\lambda_d)$. For $u \in \mathbb{R}^d$,

$$\begin{aligned}
\mathbb{E}[\exp(\langle u,V\rangle)] &= \mathbb{E}[\exp(\sum_{i\in[d]} u_i\sqrt{\lambda_i}\sum_{j\in[m]}\mathbb{Y}_{ij}x_j)] = \prod_{j,i}\mathbb{E}[\exp(u_i\sqrt{\lambda_i}\mathbb{Y}_{ij}x_j)] \\
&\leq \prod_{j,i}\exp(u_i^2\lambda_i B^2 x_j^2/2) = \exp(\frac{B^2}{2}\sum_{i\in[d]}u_i^2\lambda_i\sum_{j\in[m]}x_j^2) \\
&= \exp(\frac{B^2}{2}\sum_{i\in[d]}u_i^2\lambda_i) = \exp(\langle B^2\Sigma u,u\rangle/2).
\end{aligned}$$

Thus $V$ is a sub-Gaussian vector with moment matrix $B^2\Sigma$. Let $s = 1/(4B^2)$. Since $\Sigma \leq I$, we have $s \leq 1/(4B^2\max_{i\in[d]}\lambda_i)$. Therefore, by Lemma 20,

$$\mathbb{E}[\exp(s\|V\|^2)] \leq \exp(2sB^2\,\text{trace}(\Sigma)).$$

By Chernoff's method, $\mathbb{P}[\|V\|^2 \geq z^2] \leq \mathbb{E}[\exp(s\|V\|^2)]/\exp(sz^2)$. Thus

$$\mathbb{P}[\|V\|^2 \geq z^2] \leq \exp(2sB^2\,\text{trace}(\Sigma) - sz^2) = \exp(\frac{\text{trace}(\Sigma)}{2} - \frac{z^2}{4B^2}).$$

Set $z = t(1-\varepsilon)$. Then for all $x \in S^{m-1}$

$$\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\| \geq t(1-\varepsilon)] = \mathbb{P}[\|V\| \geq t(1-\varepsilon)] \leq \exp(\frac{\text{trace}(\Sigma)}{2} - \frac{t^2(1-\varepsilon)^2}{4B^2}).$$

Therefore, by Equation (12),

$$\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}\| \geq t] \leq \mathcal{N}_m(\varepsilon)\exp(\frac{\text{trace}(\Sigma)}{2} - \frac{t^2(1-\varepsilon)^2}{4B^2}).$$

■

**Lemma 28** *Let $\mathbb{Y}$ be a $d\times m$ matrix with $m \leq d$, such that $\mathbb{Y}_{ij}$ are independent centered random variables with variance 1 and fourth moments at most $B$. Let $\Sigma$ be a diagonal $d\times d$ PSD matrix such that $\Sigma \leq I$. There exist $\alpha > 0$ and $\eta \in (0,1)$ that depend only on $B$ such that for any $x \in S^{m-1}$*

$$\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\|^2 \leq \alpha\cdot(\text{trace}(\Sigma)-1)] \leq \eta^{\text{trace}(\Sigma)}.$$

To prove Lemma 28 we require Lemma 29 (Rudelson and Vershynin, 2008, Lemma 2.2) and Lemma 30, which extends Lemma 2.6 in the same work.

**Lemma 29** *Let $T_1,\ldots,T_n$ be independent non-negative random variables. Assume that there are $\theta>0$ and $\mu\in(0,1)$ such that for any $i$, $\mathbb{P}[T_i\leq\theta]\leq\mu$. There are $\alpha>0$ and $\eta\in(0,1)$ that depend only on $\theta$ and $\mu$ such that*

$$\mathbb{P}[\sum_{i=1}^{n}T_i<\alpha n]\leq\eta^n.$$

**Lemma 30** *Let $\mathbb{Y}$ be a $d\times m$ matrix with $m\leq d$, such that the columns of $\mathbb{Y}$ are i.i.d. random vectors. Assume further that $\mathbb{Y}_{ij}$ are centered, and have a variance of 1 and a fourth moment at most $B$. Let $\Sigma$ be a diagonal $d\times d$ PSD matrix. Then for all $x\in S^{m-1}$,*

$$\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\|\leq\sqrt{\mathrm{trace}(\Sigma)/2}]\leq 1-1/(196B).$$

**Proof** Let $x\in S^{m-1}$, and $T_i=(\sum_{j=1}^{m}\mathbb{Y}_{ij}x_j)^2$. Let $\lambda_1,\ldots,\lambda_d$ be the values on the diagonal of $\Sigma$, and let $T_\Sigma=\|\sqrt{\Sigma}\mathbb{Y}x\|^2=\sum_{i=1}^{d}\lambda_iT_i$. First, since $\mathbb{E}[\mathbb{Y}_{ij}]=0$ and $\mathbb{E}[\mathbb{Y}_{ij}]=1$ for all $i,j$, we have

$$\mathbb{E}[T_i]=\sum_{i\in[m]}x_j^2\mathbb{E}[\mathbb{Y}_{ij}^2]=\|x\|^2=1.$$

Therefore $\mathbb{E}[T_\Sigma]=\mathrm{trace}(\Sigma)$. Second, since $\mathbb{Y}_{i1},\ldots,\mathbb{Y}_{im}$ are independent and centered, we have (Ledoux and Talagrand, 1991, Lemma 6.3)

$$\mathbb{E}[T_i^2]=\mathbb{E}[(\sum_{j\in[m]}\mathbb{Y}_{ij}x_j)^4]\leq 16\mathbb{E}_\sigma[(\sum_{j\in[m]}\sigma_j\mathbb{Y}_{ij}x_j)^4],$$

where $\sigma_1,\ldots,\sigma_m$ are independent uniform $\{\pm1\}$ variables. Now, by Khinchine's inequality (Nazarov and Podkorytov, 2000),

$$\mathbb{E}_\sigma[(\sum_{j\in[m]}\sigma_j\mathbb{Y}_{ij}x_j)^4]\leq 3\mathbb{E}[(\sum_{j\in[m]}\mathbb{Y}_{ij}^2x_j^2)^2]=3\sum_{j,k\in[m]}x_j^2x_k^2\mathbb{E}[\mathbb{Y}_{ij}^2]\mathbb{E}[\mathbb{Y}_{ik}^2].$$

Now $\mathbb{E}[\mathbb{Y}_{ij}^2]\mathbb{E}[\mathbb{Y}_{ik}^2]\leq\sqrt{\mathbb{E}[\mathbb{Y}_{ij}^4]\mathbb{E}[\mathbb{Y}_{ik}^4]}\leq B$. Thus $\mathbb{E}[T_i^2]\leq 48B\sum_{j,k\in[m]}x_j^2x_k^2=48B\|x\|^4=48B$. Thus,

$$\begin{aligned}\mathbb{E}[T_\Sigma^2]&=\mathbb{E}[(\sum_{i=1}^{d}\lambda_iT_i)^2]=\sum_{i,j=1}^{d}\lambda_i\lambda_j\mathbb{E}[T_iT_j]\\&\leq\sum_{i,j=1}^{d}\lambda_i\lambda_j\sqrt{\mathbb{E}[T_i^2]\mathbb{E}[T_j^2]}\leq 48B(\sum_{i=1}^{d}\lambda_i)^2=48B\cdot\mathrm{trace}(\Sigma)^2.\end{aligned}$$

By the Paley-Zigmund inequality (Paley and Zygmund, 1932), for $\theta\in[0,1]$

$$\mathbb{P}[T_\Sigma\geq\theta\mathbb{E}[T_\Sigma]]\geq(1-\theta)^2\frac{\mathbb{E}[T_\Sigma]^2}{\mathbb{E}[T_\Sigma^2]}\geq\frac{(1-\theta)^2}{48B}.$$

Therefore, setting $\theta=1/2$, we get $\mathbb{P}[T_\Sigma\leq\mathrm{trace}(\Sigma)/2]\leq 1-1/(196B)$. ■

**Proof** [of Lemma 28] Let $\lambda_1,\ldots,\lambda_d\in[0,1]$ be the values on the diagonal of $\Sigma$. Consider a partition $Z_1,\ldots,Z_k$ of $[d]$, and denote $L_j=\sum_{i\in Z_j}\lambda_i$. There exists such a partition such that for all $j\in[k]$,

$L_j \leq 1$, and for all $j \in [k-1]$, $L_j > \frac{1}{2}$. Let $\Sigma[j]$ be the sub-matrix of $\Sigma$ that includes the rows and columns whose indexes are in $Z_j$. Let $\mathbb{Y}[j]$ be the sub-matrix of $\mathbb{Y}$ that includes the rows in $Z_j$. Denote $T_j = \|\sqrt{\Sigma[j]}\mathbb{Y}[j]x\|^2$. Then

$$\|\sqrt{\Sigma}\mathbb{Y}x\|^2 = \sum_{j\in[k]}\sum_{i\in Z_j}\lambda_i(\sum_{j=1}^{m}\mathbb{Y}_{ij}x_j)^2 = \sum_{j\in[k]}T_j.$$

We have $\text{trace}(\Sigma) = \sum_{i=1}^{d}\lambda_i \geq \sum_{j\in[k-1]}L_j \geq \frac{1}{2}(k-1)$. In addition, $L_j \leq 1$ for all $j \in [k]$. Thus $\text{trace}(\Sigma) \leq k \leq 2\,\text{trace}(\Sigma)+1$. For all $j \in [k-1]$, $L_j \geq \frac{1}{2}$, thus by Lemma 30, $\mathbb{P}[T_j \leq 1/4] \leq 1 - 1/(196B)$. Therefore, by Lemma 29 there are $\alpha > 0$ and $\eta \in (0,1)$ that depend only on $B$ such that

$$\begin{aligned}\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\|^2 < \alpha\cdot(\text{trace}(\Sigma)-1)] &\leq \mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\|^2 < \alpha(k-1)]\\ &= \mathbb{P}[\sum_{j\in[k]}T_j < \alpha(k-1)] \leq \mathbb{P}[\sum_{j\in[k-1]}T_j < \alpha(k-1)] \leq \eta^{k-1} \leq \eta^{2\,\text{trace}(\Sigma)}.\end{aligned}$$

The lemma follows by substituting $\eta$ for $\eta^2$. ■

**Proof** [of Theorem 23] We have

$$\sqrt{\lambda_{\min}(\mathbb{X}\mathbb{X}^T)} = \inf_{x\in S^{m-1}}\|\mathbb{X}^T x\| \geq \min_{x\in\mathcal{C}_m(\varepsilon)}\|\mathbb{X}^T x\| - \varepsilon\|\mathbb{X}^T\|. \tag{13}$$

For brevity, denote $L = \text{trace}(\Sigma)$. Assume $L \geq 2$. Let $m \leq L\cdot\min(1,(c-K\varepsilon)^2)$ where $c, K, \varepsilon$ are constants that will be set later such that $c - K\varepsilon > 0$. By Equation (13)

$$\begin{aligned}\mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \leq m] &\leq \mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \leq (c-K\varepsilon)^2 L]\\ &\leq \mathbb{P}[\min_{x\in\mathcal{C}_m(\varepsilon)}\|\mathbb{X}^T x\| - \varepsilon\|\mathbb{X}^T\| \leq (c-K\varepsilon)\sqrt{L}] \qquad (14)\\ &\leq \mathbb{P}[\|\mathbb{X}^T\| \geq K\sqrt{L}] + \mathbb{P}[\min_{x\in\mathcal{C}_m(\varepsilon)}\|\mathbb{X}^T x\| \leq c\sqrt{L}]. \qquad (15)\end{aligned}$$

The last inequality holds since the inequality in line (14) implies at least one of the inequalities in line (15). We will now upper-bound each of the terms in line (15). We assume w.l.o.g. that $\Sigma$ is not singular (since zero rows and columns can be removed from $\mathbb{X}$ without changing $\lambda_{\min}(\mathbb{X}\mathbb{X}^T)$). Define $\mathbb{Y} \triangleq \sqrt{\Sigma^{-1}}\mathbb{X}^T$. Note that $\mathbb{Y}_{ij}$ are independent sub-Gaussian variables with (absolute) moment $\rho$. To bound the first term in line (15), note that by Lemma 27, for any $K > 0$,

$$\mathbb{P}[\|\mathbb{X}^T\| \geq K\sqrt{L}] = \mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}\| \geq K\sqrt{L}] \leq \mathcal{N}_m(\frac{1}{2})\exp(L(\frac{1}{2} - \frac{K^2}{16\rho^2})).$$

By Rudelson and Vershynin (2009), Proposition 2.1, for all $\varepsilon \in [0,1]$, $\mathcal{N}_m(\varepsilon) \leq 2m(1+\frac{2}{\varepsilon})^{m-1}$. Therefore

$$\mathbb{P}[\|\mathbb{X}^T\| \geq K\sqrt{L}] \leq 2m5^{m-1}\exp(L(\frac{1}{2} - \frac{K^2}{16\rho^2})).$$

Let $K^2 = 16\rho^2(\frac{3}{2} + \ln(5) + \ln(2/\delta))$. Recall that by assumption $m \leq L$, and $L \geq 2$. Therefore

$$\begin{aligned}\mathbb{P}[\|\mathbb{X}^T\| \geq K\sqrt{L}] &\leq 2m5^{m-1}\exp(-L(1+\ln(5)+\ln(2/\delta)))\\ &\leq 2L5^{L-1}\exp(-L(1+\ln(5)+\ln(2/\delta))).\end{aligned}$$

Since $L \geq 2$, we have $2L\exp(-L) \leq 1$. Therefore

$$\mathbb{P}[\|\mathbb{X}^T\| \geq K\sqrt{L}] \leq 2L\exp(-L - \ln(2/\delta)) \leq \exp(-\ln(2/\delta)) = \frac{\delta}{2}. \tag{16}$$

To bound the second term in line (15), since $\mathbb{Y}_{ij}$ are sub-Gaussian with moment $\rho$, $\mathbb{E}[\mathbb{Y}_{ij}^4] \leq 5\rho^4$ (Buldygin and Kozachenko, 1998, Lemma 1.4). Thus, by Lemma 28, there are $\alpha > 0$ and $\eta \in (0,1)$ that depend only on $\rho$ such that for all $x \in S^{m-1}$, $\mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\|^2 \leq \alpha(L-1)] \leq \eta^L$. Set $c = \sqrt{\alpha/2}$. Since $L \geq 2$, we have $c\sqrt{L} \leq \sqrt{\alpha(L-1)}$. Thus

$$\begin{aligned}
\mathbb{P}[\min_{x \in C_m(\varepsilon)} \|\mathbb{X}^T x\| \leq c\sqrt{L}] &\leq \sum_{x \in C_m(\varepsilon)} \mathbb{P}[\|\mathbb{X}^T x\| \leq c\sqrt{L}] \\
&\leq \sum_{x \in C_m(\varepsilon)} \mathbb{P}[\|\sqrt{\Sigma}\mathbb{Y}x\| \leq \sqrt{\alpha(L-1)}] \leq \mathcal{N}_m(\varepsilon)\eta^L.
\end{aligned}$$

Let $\varepsilon = c/(2K)$, so that $c - K\varepsilon > 0$. Let $\theta = \min(\frac{1}{2}, \frac{\ln(1/\eta)}{2\ln(1+2/\varepsilon)})$. Set $L_\circ$ such that $\forall L \geq L_\circ$, $L \geq \frac{2\ln(2/\delta)+2\ln(L)}{\ln(1/\eta)}$. For $L \geq L_\circ$ and $m \leq \theta L \leq L/2$,

$$\begin{aligned}
\mathcal{N}_m(\varepsilon)\eta^L &\leq 2m(1+2/\varepsilon)^{m-1}\eta^L \\
&\leq L\exp(L(\theta\ln(1+2/\varepsilon) - \ln(1/\eta))) \\
&= \exp(\ln(L) + L(\theta\ln(1+2/\varepsilon) - \ln(1/\eta)/2) - L\ln(1/\eta)/2) \\
&\leq \exp(L(\theta\ln(1+2/\varepsilon) - \ln(1/\eta)/2) + \ln(\delta/2)) \qquad (17) \\
&\leq \exp(\ln(\delta/2)) = \frac{\delta}{2}. \qquad (18)
\end{aligned}$$

Line (17) follows from $L \geq L_\circ$, and line (18) follows from $\theta\ln(1+2/\varepsilon) - \ln(1/\eta)/2 \leq 0$. Set $\beta = \min\{(c-K\varepsilon)^2, 1, \theta\}$. Combining Equation (15), Equation (16) and Equation (18) we have that if $L \geq \bar{L} \triangleq \max(L_\circ, 2)$, then $\mathbb{P}[\lambda_{\min}(\mathbb{X}\mathbb{X}^T) \leq m] \leq \delta$ for all $m \leq \beta L$. Specifically, this holds for all $L \geq 0$ and for all $m \leq \beta(L - \bar{L})$. Letting $C = \beta\bar{L}$ and substituting $\delta$ for $1-\delta$ we get the statement of the theorem. ■

## References


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. *Mach. Learn.*, 30(1):31–56, 1998. ISSN 0885-6125.

Z. Bai and J. W. Silverstein. *Spectral Analysis of Large Dimensional Random Matrices*. Springer, second edition edition, 2010.

M. F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. *Journal of Computer and System Sciences*, 75(1):78–89, 2009.

P. Bartlett. Lecture notes. http://www.cs.berkeley.edu/~bartlett/courses/281b-sp06/lecture25.ps, 2006. unpublished.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

S. Ben-David, T. Lu, and D. Pál. Does unlabeled data provably help? In *Proceedings of the Twenty-First Annual Conference on Computational Learning Theory*, pages 33–44, 2008.

G. M. Benedek and A. Itai. Learnability with respect to fixed distributions. *Theoretical Computer Science*, 86(2):377–389, Sep 1991.

G. Bennett, V. Goodman, and C. M. Newman. Norms of random matrices. *Pacific J. Math.*, 59(2): 359–365, 1975.

A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. Agnostic active learning without constraints. In *Advances in Neural Information Processing Systems 23*, pages 199–207. 2010.

O. Bousquet. *Concentration Inequalities and Empirical Processes Theory Applied to the Analysis of Learning Algorithms*. PhD thesis, Ecole Polytechnique, 2002.

V. V. Buldygin and Y. V. Kozachenko. *Metric Characterization of Random Variables and Random Processes*. American Mathematical Society, 1998.

R. M. Dudley. Central limit theorems for empirical measures. *The Annals of Probability*, 6(6): 899–929, 1978.

A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant. A general lower bound on the number of examples needed for learning. In *Proceedings of the First Anuual Workshop on Computational Learning Theory*, pages 139–154, Aug 1988.

C. Gentile and D. P. Helmbold. Improved lower bounds for learning from noisy examples: an information-theoretic approach. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory (COLT)*, pages 104–115, 1998.

M. J. Kearns and R. E. Schapire. Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and System Sciences*, 48(3):464–497, 1994.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces*. Springer, 1991.

P. Liang and N. Srebro. On the interaction between norm and dimensionality: Multiple regimes in learning. In *Proceedings of the Twenty-Seventh International Conference on Machine learning (ICML)*, 2010.

S. Mendelson. Rademacher averages and phase transitions in glivenko-cantelli classes. *IEEE Transactions on Information Theory*, 48(1):251–263, 2002.

F. L. Nazarov and A. Podkorytov. Ball, haagerup, and distribution functions. *Operator Theory: Advances and Applications*, 113 (Complex Analysis, Operators, and Related Topics):247–267, 2000.

A. Ng. Feature selection, $l_1$ vs. $l_2$ regularization, and rotational invariance. In *Proceedings of the Twenty-First International Conference on Machine learning (ICML)*, 2004.

R. Paley and A. Zygmund. A note on analytic functions in the unit circle. *Proceedings of the Cambridge Philosophical Society*, 28:266272, 1932.

D. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, 1984.

M. Rudelson and R. Vershynin. The Littlewood-Offord problem and invertibility of random matrices. *Advances in Mathematics*, 218(2):600–633, 2008.

M. Rudelson and R. Vershynin. The smallest singular value of a random rectangular matrix. *Communications on Pure and Applied Mathematics*, 62:1707–1739, 2009.

S. Sabato, N. Srebro, and N. Tishby. Tight sample complexity of large-margin learning. In *Advances in Neural Information Processing Systems 23 (NIPS)*, pages 2038–2046, 2010.

B. Schölkopf, J. Shawe-Taylor, A. J. Smola, and R. Williamson. Generalization bounds via eigenvalues of the gram matrix. Technical Report NC2-TR-1999-035, NeuroCOLT2, 1999.

I. Steinwart and C. Scovel. Fast rates for support vector machines using Gaussian kernels. *Annals of Statistics*, 35(2):575–607, 2007.

V. N. Sudakov. Gaussian processes and measures of solid angles in hilbert space. *Sov. Math.Dokl.*, 12:412–415, 1971.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its applications*, XVI(2):264–280, 1971.

V. N. Vapnik and A. Y. Chervonenkis. *Theory of pattern recognition*. Nauka, Moscow, 1974. (In Russian).

N. Vayatis and R. Azencott. Distribution-dependent vapnik-chervonenkis bounds. In *EuroCOLT '99*, pages 230–240, London, UK, 1999. Springer-Verlag. ISBN 3-540-65701-0.

T. Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2:527–550, 2002.

Journal of Machine Learning Research 14 (2013) 2151-2188 Submitted 6/12; Revised 4/13; Published 7/13

# Convex and Scalable Weakly Labeled SVMs

**Yu-Feng Li** LIYF@LAMDA.NJU.EDU.CN
*National Key Laboratory for Novel Software Technology*
*Nanjing University*
*Nanjing 210023, China*

**Ivor W. Tsang** IVORTSANG@NTU.EDU.SG
*School of Computer Engineering*
*Nanyang Technological University*
*Singapore 639798*

**James T. Kwok** JAMESK@CSE.UST.HK
*Department of Computer Science and Engineering*
*Hong Kong University of Science & Technology*
*Hong Kong*

**Zhi-Hua Zhou** ZHOUZH@LAMDA.NJU.EDU.CN
*National Key Laboratory for Novel Software Technology*
*Nanjing University*
*Nanjing 210023, China*

**Editor:** Sathiya Keerthi

## Abstract

In this paper, we study the problem of learning from *weakly labeled data*, where labels of the training examples are incomplete. This includes, for example, (i) semi-supervised learning where labels are partially known; (ii) multi-instance learning where labels are implicitly known; and (iii) clustering where labels are completely unknown. Unlike supervised learning, learning with weak labels involves a difficult Mixed-Integer Programming (MIP) problem. Therefore, it can suffer from poor scalability and may also get stuck in local minimum. In this paper, we focus on SVMs and propose the WELLSVM via a novel *label generation* strategy. This leads to a convex relaxation of the original MIP, which is at least as tight as existing convex Semi-Definite Programming (SDP) relaxations. Moreover, the WELLSVM can be solved via a sequence of SVM subproblems that are much more scalable than previous convex SDP relaxations. Experiments on three weakly labeled learning tasks, namely, (i) semi-supervised learning; (ii) multi-instance learning for locating regions of interest in content-based information retrieval; and (iii) clustering, clearly demonstrate improved performance, and WELLSVM is also readily applicable on large data sets.

**Keywords:** weakly labeled data, semi-supervised learning, multi-instance learning, clustering, cutting plane, convex relaxation

## 1. Introduction

Obtaining labeled data is expensive and difficult. For example, in scientific applications, obtaining the labels involves repeated experiments that may be hazardous; in drug prediction, deriving active molecules of a new drug involves expensive expertise that may not even be available. On the other hand, *weakly labeled data*, where the labels are incomplete, are often ubiquitous in many

©2013 Yu-Feng Li, Ivor Tsang, James Kwok and Zhi-Hua Zhou.

applications. Therefore, exploiting weakly labeled training data may help improve performance and discover the underlying structure of the data. Indeed, this has been regarded as one of the most challenging tasks in machine learning research (Mitchell, 2006).

Many weak-label learning problems have been proposed. In the following, we summarize several major learning paradigms with weakly labeled data:

- *Labels are partially known.* A representative example is semi-supervised learning (SSL) (Chapelle et al., 2006b; Zhu, 2006; Zhou and Li, 2010), where most of the training examples are unlabeled and only a few are labeled. SSL improves generalization performance by using the unlabeled examples that are often abundant. In the past decade, SSL has attracted much attention and achieved successful results in diverse applications such as text categorization, image retrieval, and medical diagnosis.

- *Labels are implicitly known.* Multi-instance learning (MIL) (Dietterich et al., 1997) is the most prominent example in this category. In MIL, training examples are called *bags*, each of which contains multiple instances. Many real-world objects can be naturally described by multiple instances. For example, an image (bag) usually contains multiple semantic regions, and each region is an instance. Instead of describing an object as a single instance, the multi-instance representation can help separate different semantics within the object. MIL has been successfully applied to diverse domains such as image classification, text categorization, and web mining. The relationship between multi-instance learning and semi-supervised learning has also been discussed in Zhou and Xu (2007).

  In traditional MIL, a bag is labeled positive when it contains at least one positive instance, and is labeled negative otherwise. Although the bag labels are often available, the instance labels are only implicitly known. It is worth noting that identification of the key (or positive) instances from the positive bags can be very useful in many real-world applications. For example, in content-based information retrieval (CBIR), the explicit identification of regions of interest (ROI) can help the user to recognize images that he/she wants quickly (especially when the system returns a large number of images). Similarly, to detect suspect areas in some medical and military applications, a quick scanning of a huge number of images is required. Again, it is very desirable if ROIs can be identified. Besides providing an accurate and efficient prediction, the identification of key instances is also useful in understanding ambiguous objects (Li et al., 2012).

- *Labels are totally unknown.* This becomes unsupervised learning (Jain and Dubes, 1988), which aims at discovering the underlying structure (or concepts/labels) of the data and grouping similar examples together. Clustering is valuable in data analysis, and is widely used in various domains including information retrieval, computer version, and bioinformatics.

- There are other kinds of weak-label learning problems. For instances, Angluin and Laird (1988) and references therein studied noisy-tolerant problems where the label information is noisy; Sheng et al. (2008) and references therein considered learning from multiple annotation results by different experts in which all the experts are imperfect; Sun et al. (2010) and Bucak et al. (2011) considered weakly labeled data in the context of multi-label learning, whereas Yang et al. (2013) considered weakly labeled data in the context of multi-instance multi-label learning (Zhou et al., 2012).

Unlike supervised learning where the training labels are complete, weak-label learning needs to infer the integer-valued labels of the training examples, resulting in a difficult mixed-integer programming (MIP). To solve this problem, many algorithms have been proposed, including global optimization (Chapelle et al., 2008; Sindhwani et al., 2006) and convex SDP relaxations (Xu et al., 2005; Xu and Schuurmans, 2005; De Bie and Cristianini, 2006; Guo, 2009). Empirical studies have demonstrated their promising performance on small data sets. Although SDP convex relaxations can reduce the training time complexity of global optimization methods from exponential to polynomial, they still cannot handle medium-sized data sets having thousands of examples. Recently, several algorithms resort to using non-convex optimization techniques (such as alternating optimization methods (Andrews et al., 2003; Zhang et al., 2007; Li et al., 2009b) and constrained convex-concave procedure (Collobert et al., 2006; Cheung and Kwok, 2006; Zhao et al., 2008). Although these approaches are often efficient, they can only obtain locally optimal solutions and can easily get stuck in local minima. Therefore, it is desirable to develop a scalable yet convex optimization method for learning with large-scale weakly labeled data. Moreover, unlike several scalable graph-based methods proposed for the transductive setup (Subramanya and Bilmes, 2009; Zhang et al., 2009a; Vapnik, 1998), here we are more interested in inductive learning methods.

In this paper, we will focus on the binary support vector machines (SVM). Extending our preliminary works in Li et al. (2009a,c), we propose a convex weakly labeled SVM (denoted WELLSVM (WEakly LabeLed SVM)) via a novel "label generation" strategy. Instead of obtaining a label relation matrix via SDP, WELLSVM maximizes the margin by generating the most violated label vectors iteratively, and then combines them via efficient multiple kernel learning techniques. The whole procedure can be formulated as a convex relaxation of the original MIP problem. Furthermore, it can be shown that the learned linear combination of label vector outer-products is in the convex hull of the label space. Since the convex hull is the smallest convex set containing the target non-convex set (Boyd and Vandenberghe, 2004), our formulation is at least as tight as the convex SDP relaxations proposed in Xu et al. (2005), De Bie and Cristianini (2006) and Xu and Schuurmans (2005). Moreover, WELLSVM involves a series of SVM subproblems, which can be readily solved in a scalable and efficient manner via state-of-the-art SVM software such as LIBSVM (Fan et al., 2005), SVM-*perf* (Joachims, 2006), LIBLINEAR (Hsieh et al., 2008) and CVM (Tsang et al., 2006). Therefore, WELLSVM scales much better than existing SDP approaches or even some non-convex approaches. Experiments on three common weak-label learning tasks (semi-supervised learning, multi-instance learning, and clustering) validate the effectiveness and scalability of the proposed WELLSVM.

The rest of this paper is organized as follows. Section 2 briefly introduces large margin weak-label learning. Section 3 presents the proposed WELLSVM and analyzes its time complexity. Section 4 presents detailed formulations on three weak-label learning problems. Section 5 shows some comprehensive experimental results and the last section concludes.

In the following, $\mathbf{M} \succ 0$ (resp. $\mathbf{M} \succeq 0$) denotes that the matrix $\mathbf{M}$ is symmetric and positive definite (pd) (resp. positive semidefinite (psd)). The transpose of vector / matrix (in both the input and feature spaces) is denoted by the superscript $'$, and $\mathbf{0}, \mathbf{1} \in \mathbb{R}^n$ denote the zero vector and the vector of all ones, respectively. The inequality $\mathbf{v} = [v_1, \ldots, v_k]' \geq \mathbf{0}$ means that $v_i \geq 0$ for $i = 1, \ldots, k$. Similarly, $\mathbf{M} \geq \mathbf{0}$ means that all elements in the matrix $\mathbf{M}$ are nonnegative.

## 2. Large-Margin Weak-Label Learning

We commence with the simpler supervised learning scenario. Given a set of labeled examples $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$ where $\mathbf{x}_i \in \mathcal{X}$ is the input and $y_i \in \{\pm 1\}$ is the output, we aim to find a decision function $f : \mathcal{X} \to \{\pm 1\}$ such that the following structural risk functional is minimized:

$$\min_f \ \Omega(f) + C\, \ell_f(\mathcal{D}). \tag{1}$$

Here, $\Omega$ is a regularizer related to large margin on $f$, $\ell_f(\mathcal{D})$ is the empirical loss on $\mathcal{D}$, and $C$ is a regularization parameter that trades off the empirical risk and model complexity. Both $\Omega$ and $\ell_f(\cdot)$ are problem-dependent. In particular, when $\ell_f(\cdot)$ is the hinge loss (or its variants), the obtained $f$ is a large margin classifier. It is notable that both $\Omega$ and $L_f(\cdot)$ are usually convex. Thus, Equation (1) is a convex problem whose globally optimal solution can be efficiently obtained via various convex optimization techniques.

In weak-label learning, labels are not available on all $N$ training examples, and so also need to be learned. Let $\hat{\mathbf{y}} = [\hat{y}_1, \cdots, \hat{y}_N]' \in \{\pm 1\}^N$ be the vector of (known and unknown) labels on all the training examples. The basic idea of large-margin weak-label learning is that the structural risk functional in Equation (1) is minimized w.r.t. both the labeling[1] $\hat{\mathbf{y}}$ and decision function $f$. Hence, Equation (1) is extended to

$$\min_{\hat{\mathbf{y}} \in \mathcal{B}} \min_f \quad \Omega(f) + C\, \ell_f(\{\mathbf{x}_i, \hat{y}_i\}_{i=1}^N), \tag{2}$$

where $\mathcal{B}$ is a set of candidate label assignments obtained from some domain knowledge. For example, when the positive and negative examples are known to be approximately balanced, we can set $\mathcal{B} = \{\hat{\mathbf{y}} \ : \ -\beta \le \sum_{i=1}^N \hat{y}_i \le \beta\}$ where $\beta$ is a small constant controlling the class imbalance.

### 2.1 State-of-The-Art Approaches

As Equation (2) involves optimizing the integer variables $\hat{\mathbf{y}}$, it is no longer a convex optimization problem but a mixed-integer program. This can easily suffer from the local minimum problem. Recently, a lot of efforts have been devoted to solve this problem. They can be grouped into three categories. The first strategy optimizes Equation (2) via variants of non-convex optimization. Examples include alternating optimization (Zhang et al., 2009b; Li et al., 2009b; Andrews et al., 2003), in which we alternatively optimize variable $\hat{\mathbf{y}}$ (or $f$) by keeping the other variable $f$ (or $\hat{\mathbf{y}}$) constant; constrained convex-concave procedure (CCCP) (also known as DC programming) (Horst and Thoai, 1999; Zhao et al., 2008; Collobert et al., 2006; Cheung and Kwok, 2006), in which the non-convex objective function or constraint is decomposed as a difference of two convex functions; local combinatorial search (Joachims, 1999), in which the labels of two examples in opposite classes are sequentially switched. These approaches are often computationally efficient. However, since they are based on non-convex optimization, they may inevitably get stuck in local minima.

The second strategy obtains the globally optimal solution of Equation (2) via global optimization. Examples include branch-and-bound (Chapelle et al., 2008) and deterministic annealing (Sindhwani et al., 2006). Since they aim at obtaining the globally optimal (instead of the locally optimal) solution, excellent performance can be expected (Chapelle et al., 2008). However, their worst-case

1. To simplify notations, we write $\min_{\hat{\mathbf{y}} \in \mathcal{B}}$, though indeed one only needs to minimize w.r.t. the unknown labels in $\hat{\mathbf{y}}$.

computational costs can scale exponentially as the data set size. Hence, these approaches can only be applied to small data sets with just hundreds of training examples.

The third strategy is based on convex relaxations. The original non-convex problem is first relaxed to a convex problem, whose globally optimal solution can be efficiently obtained. This is then rounded to recover an approximate solution of the original problem. If the relaxation is tight, the approximate solution obtained is close to the global optimum of the original problem and good performance can be expected. Moreover, the involved convex programming solver has a time complexity substantially lower than that for global optimization. A prominent example of convex relaxation is the use of semidefinite programming (SDP) techniques (Xu et al., 2005; Xu and Schuurmans, 2005; De Bie and Cristianini, 2006; Guo, 2009), in which a positive semidefinite matrix is used to approximate the matrix of label outer-products. The time complexity of this SDP-based strategy is $O(N^{6.5})$ (Lobo et al., 1998; Nesterov and Nemirovskii, 1987), where $N$ is the data set size, and can be further reduced to $O(N^{4.5})$ (Zhang et al., 2009b; Valizadegan and Jin, 2007). However, this is still expensive for medium-sized data sets with several thousands of examples.

To summarize, existing weak-label learning approaches are not scalable or can be sensitive to initialization. In this paper, we propose the WELLSVM algorithm to address these two issues.

## 3. WELLSVM

In this section, we first introduce the SVM dual which will be used as a basic reformulation of our proposal, and then we present the general formulation of WELLSVM. Detailed formulations on three common weak-label learning tasks will be presented in Section 4.

### 3.1 Duals in Large Margin Classifiers

In large margin classifiers, the inner minimization problem of Equation (2) is often cast in the dual form. For example, for the standard SVM without offset, we have $\Omega = \frac{1}{2}\|\mathbf{w}\|^2$ and $\ell_f(D)$ is the summed hinge loss. The inner minimization problem is then

$$\begin{aligned} \min_{\mathbf{w},\xi} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & \hat{y}_i\mathbf{w}'\phi(\mathbf{x}_i) \geq 1-\xi_i,\ \xi_i \geq 0,\ i = 1\ldots,N, \end{aligned}$$

where $\phi(\mathbf{x}_i)$ is the feature map induced by kernel $\kappa$, and its dual is

$$\begin{aligned} \max_{\alpha} \quad & \alpha'\mathbf{1} - \frac{1}{2}\alpha'\left(\mathbf{K}\odot\hat{\mathbf{y}}\hat{\mathbf{y}}'\right)\alpha \\ \text{s.t.} \quad & C\mathbf{1} \geq \alpha \geq \mathbf{0}, \end{aligned}$$

where $\alpha \in \mathbb{R}^N$ is the dual variable, $\mathbf{K} \in \mathbb{R}^{N\times N}$ is the kernel matrix defined on the $N$ samples, and $\odot$ is the element-wise product. For more details on the duals of large margin classifiers, interested readers are referred to Schölkopf and Smola (2002) and Cristianini et al. (2002).

In this paper, we make the following assumption on this dual.

**Assumption 1** *The dual of the inner minimization of Equation (2) can be written as:* $\max_{\alpha\in\mathcal{A}} G(\alpha,\hat{\mathbf{y}})$, *where* $\alpha = [\alpha_1,\ldots,\alpha_N]'$ *contains the dual variables and*

- $\mathcal{A}$ *is a convex set;*

- $G(\alpha, \hat{\mathbf{y}})$ *is a concave function in* $\alpha$ *for any fixed* $\hat{\mathbf{y}}$*;*
- $g_{\mathbf{y}}(\alpha) = -G(\alpha, \mathbf{y})$ *is* $\lambda$*-strongly convex and M-Lipschitz. In other words,* $\nabla^2 g_{\mathbf{y}}(\alpha) - \lambda \mathbf{I} \succeq 0$*, where* $\mathbf{I}$ *is the identity matrix, and* $\|g_{\mathbf{y}}(\alpha) - g_{\mathbf{y}}(\bar{\alpha})\| \leq M\|\alpha - \bar{\alpha}\|$*,* $\forall \mathbf{y} \in \mathcal{B}$*,* $\alpha, \bar{\alpha} \in \mathcal{A}$*;*
- $\forall \hat{\mathbf{y}} \in \mathcal{B}$*,* $lb \leq \max_{\alpha \in \mathcal{A}} G(\alpha, \hat{\mathbf{y}}) \leq ub$*, where lb and ub are polynomial in N;*
- $G(\alpha, \hat{\mathbf{y}})$ *can be rewritten as* $\bar{G}(\alpha, \mathbf{M})$*, where* $\mathbf{M}$ *is a psd matrix, and* $\bar{G}$ *is concave in* $\alpha$ *and linear in* $\mathbf{M}$*.*

With this assumption, Equation (2) can be written as

$$\min_{\hat{\mathbf{y}} \in \mathcal{B}} \max_{\alpha \in \mathcal{A}} \quad G(\alpha, \hat{\mathbf{y}}), \tag{3}$$

Assume that the kernel matrix $\mathbf{K}$ is pd (i.e., the smallest eigenvalue $\lambda_{\min} > 0$) and all its entries are bounded ($|K_{ij}| \leq \upsilon$ for some $\upsilon$). It is easy to see that the following SVM variants satisfy Assumption 1.

- Standard SVM without offset: We have

$$\begin{aligned}
\mathcal{A} &= \{\alpha \mid C\mathbf{1} \geq \alpha \geq \mathbf{0}\}, \\
G(\alpha, \hat{\mathbf{y}}) &= \alpha'\mathbf{1} - \frac{1}{2}\alpha'\left(\mathbf{K} \odot \hat{\mathbf{y}}\hat{\mathbf{y}}'\right)\alpha, \\
\nabla^2 g_{\mathbf{y}}(\alpha) &= \mathbf{K} \odot \mathbf{y}\mathbf{y}' \succeq \lambda_{\min}(\mathbf{I} \odot \mathbf{y}\mathbf{y}') = \lambda_{\min}\mathbf{I}, \\
\|g_{\mathbf{y}}(\alpha) - g_{\mathbf{y}}(\bar{\alpha})\| &\leq (1 + C\upsilon N)\sqrt{N}\|\alpha - \bar{\alpha}\|, \\
0 &\leq \max_{\alpha \in \mathcal{A}} G(\alpha, \hat{\mathbf{y}}) \leq CN, \\
\bar{G}(\alpha, \mathbf{M}_{\hat{\mathbf{y}}}) &= \alpha'\mathbf{1} - \frac{1}{2}\alpha'\left(\mathbf{K} \odot \mathbf{M}_{\hat{\mathbf{y}}}\right)\alpha, \quad \text{where } \mathbf{M}_{\hat{\mathbf{y}}} = \hat{\mathbf{y}}\hat{\mathbf{y}}'.
\end{aligned}$$

- $\nu$-SVM (Schölkopf and Smola, 2002): We have

$$\begin{aligned}
\mathcal{A} &= \{\alpha \mid \alpha \geq \mathbf{0}, \alpha'\mathbf{1} = 1\}, \\
G(\alpha, \hat{\mathbf{y}}) &= -\frac{1}{2}\alpha'\left((\mathbf{K} + \frac{1}{C}\mathbf{I}) \odot \hat{\mathbf{y}}\hat{\mathbf{y}}'\right)\alpha, \\
\nabla^2 g_{\mathbf{y}}(\alpha) &= \left(\mathbf{K} + \frac{1}{C}\mathbf{I}\right) \odot \mathbf{y}\mathbf{y}' \succeq \left(\lambda_{\min} + \frac{1}{C}\right)\ (\mathbf{I} \odot \mathbf{y}\mathbf{y}') = \left(\lambda_{\min} + \frac{1}{C}\right)\mathbf{I}, \\
\|g_{\mathbf{y}}(\alpha) - g_{\mathbf{y}}(\bar{\alpha})\| &\leq \left(\upsilon + \frac{1}{C}\right)N\sqrt{N}\|\alpha - \bar{\alpha}\|, \\
-\frac{1}{2}\left(\upsilon + \frac{1}{C}\right) &\leq \max_{\alpha \in \mathcal{A}} G(\alpha, \hat{\mathbf{y}}) \leq 0, \\
\bar{G}(\alpha, \mathbf{M}_{\hat{\mathbf{y}}}) &= -\frac{1}{2}\alpha'\left((\mathbf{K} + \frac{1}{C}\mathbf{I}) \odot \mathbf{M}_{\hat{\mathbf{y}}}\right)\alpha.
\end{aligned}$$

### 3.2 WELLSVM

Interchanging the order of $\max_{\alpha\in\mathcal{A}}$ and $\min_{\hat{\mathbf{y}}\in\mathcal{B}}$ in Equation (3), we obtain the proposed WELLSVM:

$$(\text{WELLSVM}) \quad \max_{\alpha\in\mathcal{A}} \min_{\hat{\mathbf{y}}\in\mathcal{B}} \; G(\alpha,\hat{\mathbf{y}}). \tag{4}$$

Using the minimax theorem (Kim and Boyd, 2008), the optimal objective of Equation (3) upper-bounds that of Equation (4). Moreover, Equation (4) can be transformed as

$$\max_{\alpha\in\mathcal{A}} \; \Big\{ \max_{\theta} \; \theta \quad \text{s.t.} \quad G(\alpha,\hat{\mathbf{y}}_t) \geq \theta, \; \forall \hat{\mathbf{y}}_t \in \mathcal{B} \Big\}, \tag{5}$$

from which we obtain the following Proposition.

**Proposition 1** *The objective of* WELLSVM *can be rewritten as the following optimization problem:*

$$\min_{\mu\in\mathcal{M}} \max_{\alpha\in\mathcal{A}} \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t G(\alpha,\hat{\mathbf{y}}_t), \tag{6}$$

*where $\mu$ is the vector of $\mu_t$'s, $\mathcal{M}$ is the simplex $\{\mu \mid \sum_t \mu_t = 1, \mu_t \geq 0\}$, and $\hat{\mathbf{y}}_t \in \mathcal{B}$.*

**Proof** For the inner optimization in Equation (5), let $\mu_t \geq 0$ be the dual variable for each constraint. Its Lagrangian can be obtained as

$$\theta + \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t \Big( G(\alpha,\hat{\mathbf{y}}_t) - \theta \Big).$$

Setting the derivative w.r.t. $\theta$ to zero, we have $\sum_t \mu_t = 1$. We can then replace the inner optimization subproblem with its dual and Equation (5) becomes:

$$\max_{\alpha\in\mathcal{A}} \min_{\mu\in\mathcal{M}} \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t G(\alpha,\hat{\mathbf{y}}_t) = \min_{\mu\in\mathcal{M}} \max_{\alpha\in\mathcal{A}} \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t G(\alpha,\hat{\mathbf{y}}_t).$$

Here, we use the fact that the objective function is convex in $\mu$ and concave in $\alpha$. ■

Recall that $G(\alpha,\hat{\mathbf{y}})$ is concave in $\alpha$. Thus, the constraints in Equation (5) are convex. It is evident that the objective in Equation (5) is linear in both $\alpha$ and $\theta$. Therefore, Equation (5) is a *convex* problem. In other words, WELLSVM is a convex relaxation of Equation (2).

### 3.3 Tighter than SDP Relaxations

In this section, we compare our minimax relaxation with SDP relaxations. It is notable that the SVM without offset is always employed by previous SDP relaxations (Xu et al., 2005; Xu and Schuurmans, 2005; De Bie and Cristianini, 2006).

Recall the symbols in Section 3.1. Define

$$\mathcal{Y}_0 = \big\{ \mathbf{M} \mid \mathbf{M} = \mathbf{M}_{\hat{\mathbf{y}}}, \; \hat{\mathbf{y}} \in \mathcal{B} \big\}.$$

The original mixed-integer program in Equation (3) is the same as

$$\min_{\mathbf{M}\in\mathcal{Y}_0}\max_{\alpha\in\mathcal{A}} \bar{G}(\alpha,\mathbf{M}). \tag{7}$$

Define $\mathcal{Y}_1 = \left\{\mathbf{M} \mid \mathbf{M} = \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t \mathbf{M}_{\hat{\mathbf{y}}_t},\ \ \mu \in \mathcal{M}\right\}$. Our minimax relaxation in Equation (6) can be written as

$$\begin{aligned}\min_{\mu\in\mathcal{M}}\max_{\alpha\in\mathcal{A}} \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t \bar{G}(\alpha,\mathbf{M}_{\hat{\mathbf{y}}_t}) &= \min_{\mu\in\mathcal{M}}\max_{\alpha\in\mathcal{A}} \bar{G}\left(\alpha, \sum_{t:\hat{\mathbf{y}}_t\in\mathcal{B}} \mu_t \mathbf{M}_{\hat{\mathbf{y}}_t}\right) \\ &= \min_{\mathbf{M}\in\mathcal{Y}_1}\max_{\alpha\in\mathcal{A}} \bar{G}(\alpha,\mathbf{M}).\end{aligned} \tag{8}$$

On the other hand, the SDP relaxations in Xu et al. (2005); Xu and Schuurmans (2005) and De Bie and Cristianini (2006) are of the form

$$\min_{\mathbf{M}\in\mathcal{Y}_2}\max_{\alpha\in\mathcal{A}} \bar{G}(\alpha,\mathbf{M}),$$

where $\mathcal{Y}_2 = \left\{\mathbf{M} \mid \mathbf{M} \succeq 0, \mathbf{M} \in \mathcal{M}_{\mathcal{B}}\right\}$, and $\mathcal{M}_{\mathcal{B}}$ is a convex set related to $\mathcal{B}$. For example, in the context of clustering, Xu et al. (2005) used $\mathcal{B} = \{\hat{\mathbf{y}} | -\beta \le \mathbf{1}'\hat{\mathbf{y}} \le \beta\}$, where $\beta$ is a parameter controlling the class imbalance, and $\mathcal{M}_{\mathcal{B}}$ is defined as

$$\begin{aligned}\mathcal{M}_{\mathcal{B}}^{\text{clustering}} = \Big\{\mathbf{M} = [m_{ij}] \mid\ & -1 \le m_{ij} \le 1; m_{ii} = 1, m_{ij} = m_{ji}, \\ & m_{ik} \ge m_{ij} + m_{jk} - 1, m_{jk} \ge -m_{ij} - m_{ik} - 1, \\ & -\beta \le \sum_{i=1}^{N} m_{ij} \le \beta,\ \ \forall i,j,k = 1,\ldots,N\Big\}.\end{aligned}$$

It is easy to verify that $\mathcal{Y}_0 \subseteq \mathcal{Y}_2$ and $\mathcal{Y}_2$ is convex. Similarly, in semi-supervised learning, Xu and Schuurmans (2005) and De Bie and Cristianini (2006) defined $\mathcal{M}_{\mathcal{B}}$ as a subset[2] of $\mathcal{M}_{\mathcal{B}}^{\text{clustering}}$. Again, $\mathcal{Y}_0 \subseteq \mathcal{Y}_2$ and $\mathcal{Y}_2$ is convex.

**Theorem 1** *The relaxation of* WELLSVM *is at least as tight as the SDP relaxations in Xu et al. (2005); Xu and Schuurmans (2005) and De Bie and Cristianini (2006).*

**Proof** Note that $\mathcal{Y}_1$ is the convex hull of $\mathcal{Y}_0$, that is, the smallest convex set containing $\mathcal{Y}_0$ (Boyd and Vandenberghe, 2004). Therefore, Equation (8) gives the tightest convex relaxation of Equation (7), that is, $\mathcal{Y}_1 \subseteq \mathcal{Y}_2$. In other words, our relaxation is at least as tight as SDP relaxations. ■

### 3.4 Cutting Plane Algorithm by Label Generation

It appears that existing convex optimization techniques can be readily used to solve the convex problem in Equation (6), or equivalently Equation (5). However, note that there can be an exponential number of constraints in Equation (5), and so a direct optimization is computationally intractable.

2. For a more precise definition, interested readers are referred to Xu and Schuurmans (2005) and De Bie and Cristianini (2006).

**Algorithm 1** Cutting plane algorithm for WELLSVM.

1: Initialize $\hat{\mathbf{y}}$ and $\mathcal{C} = \emptyset$.
2: **repeat**
3: Update $\mathcal{C} \leftarrow \{\hat{\mathbf{y}}\} \bigcup \mathcal{C}$.
4: Obtain the optimal $\alpha$ from Equation (9).
5: Generate a violated $\hat{\mathbf{y}}$.
6: **until** $G(\alpha, \hat{\mathbf{y}}) > \min_{\mathbf{y} \in \mathcal{C}} G(\alpha, \mathbf{y}) - \varepsilon$ (where $\varepsilon$ is a small constant) or the decrease of objective value is smaller than a threshold.

Fortunately, typically not all these constraints are active at optimality, and including only a subset of them can lead to a very good approximation of the original optimization problem. Therefore, we can apply the cutting plane method (Kelley, 1960).

The cutting plane algorithm is described in Algorithm 1. First, we initialize a label vector $\hat{\mathbf{y}}$ and the working set $\mathcal{C}$ to $\{\hat{\mathbf{y}}\}$, and obtain $\alpha$ from

$$\min_{\mu \in \mathcal{M}} \max_{\alpha \in \mathcal{A}} \sum_{t: \hat{\mathbf{y}}_t \in \mathcal{C}} \mu_t G(\alpha, \hat{\mathbf{y}}_t) \tag{9}$$

via standard supervised learning methods. Then, a violated label vector $\hat{\mathbf{y}}$ in Equation (5) is *generated* and added to $\mathcal{C}$. The process is repeated until the termination criterion is met. Since the size of the working set $\mathcal{C}$ is often much smaller than that of $\mathcal{B}$, one can use existing convex optimization techniques to obtain $\alpha$ from Equation (9).

For the non-convex optimization methods reviewed in Section 2.1, a new label assignment for the unlabeled data is also generated in each iteration. However, they are very different from our proposal. First, those algorithms do not take the previous label assignments into account, while, as will be seen in Section 4.1.2, our WELLSVM aims to learn a combination of previous label assignments. Moreover, they update the label assignment to approach a locally optimal solution, while our WELLSVM aims to obtain a tight convex relaxation solution.

### 3.5 Computational Complexity

The key to analyzing the running time of Algorithm 1 is its convergence rate, and we have the following Theorem.

**Theorem 2** *Let $p^{(t)}$ be the optimal objective value of Equation (9) at the $t$-th iteration. Then,*

$$p^{(t+1)} \leq p^{(t)} - \eta, \tag{10}$$

*where* $\eta = \left(\frac{-c+\sqrt{c^2+4\varepsilon}}{2}\right)^2$, *and* $c = M\sqrt{2/\lambda}$.

Proof is in Appendix A. From Theorem 2, we can obtain the following convergence rate.

**Proposition 2** *Algorithm 1 converges in no more than $\frac{p^{(1)}-p^*}{\eta}$ iterations, where $p^*$ is the optimal objective value of* WELLSVM.

According to Assumption 1, we have $p^* = \min_{\hat{\mathbf{y}} \in \mathcal{B}} \max_{\alpha \in \mathcal{A}} G(\alpha, \hat{\mathbf{y}}) \geq lb$ and $p^{(1)} = \max_{\alpha \in \mathcal{A}} G(\alpha, \hat{\mathbf{y}}) \leq ub$. Moreover, recall that $lb$ and $ub$ are polynomial in $N$. Thus, Proposition 2 shows that with the use of the cutting plane algorithm, the number of active constraints only scales polynomially in $N$. In particular, as discussed in Section 3.1, for the ν-SVM, $lb = -\frac{1}{2}(\upsilon + \frac{1}{C})$ and $ub = 0$, both of which are unrelated to $N$. Thus, the number of active constraints only scales as $O(1)$.

Proposition 2 can be further refined by taking the search effort of a violated label into account. The proof is similar to that of Theorem 2.

**Proposition 3** *Let $\varepsilon_r \geq \varepsilon$, $\forall r = 1, 2, \ldots$, be the magnitude of the violation of a violated label in the r-th iteration, that is, $\varepsilon_r = \min_{\mathbf{y} \in C_r} G(\alpha, \mathbf{y}) - G(\alpha, \hat{\mathbf{y}}^r)$, where $C_r$ and $\hat{\mathbf{y}}^r$ denote the set of violated labels and the violated label obtained in the r-th iteration, respectively. Let $\eta_r = \left(\frac{-c + \sqrt{c^2 + 4\varepsilon_r}}{2}\right)^2$. Then, Algorithm 1 converges in no more than R iterations where $\sum_{r=1}^{R} \eta_r \geq p^{(1)} - p^*$.*

Hence, the more effort is spent on finding a violated label, the faster is the convergence. This represents a trade-off between the convergence rate and cost in each iteration.

We will show in Section 4 that step 4 of Algorithm 1 can be addressed via multiple kernel learning techniques which only involve a series of SVM subproblems that can be solved efficiently by state-of-the-art SVM software such as LIBSVM (Fan et al., 2005) and LIBLINEAR (Hsieh et al., 2008), while step 5 can be efficiently addressed by sorting. Therefore, the total time complexity of WELLSVM scales as the existing SVM solvers, and is significantly faster than SDP relaxations.

## 4. Three Weak-Label Learning Problems

In this section, we present the detailed formulations of WELLSVM on three common weak-label learning tasks, namely, semi-supervised learning (Section 4.1), multi-instance learning (Section 4.2), and clustering (Section 4.3).

### 4.1 Semi-Supervised Learning

In semi-supervised learning, not all the training labels are known. Let $\mathcal{D}_{\mathcal{L}} = \{\mathbf{x}_i, y_i\}_{i=1}^{l}$ and $\mathcal{D}_{\mathcal{U}} = \{\mathbf{x}_j\}_{j=l+1}^{N}$ be the sets of labeled and unlabeled examples, respectively, and $\mathcal{L} = \{1, \ldots, l\}$ (resp. $\mathcal{U} = \{l+1, \ldots, N\}$) be the index set of the labeled (resp. unlabeled) examples. In semi-supervised learning, unlabeled data are typically much more abundant than labeled data, that is, $N - l \gg l$. Hence, one can obtain a trivially "optimal" solution with infinite margin by assigning all the unlabeled examples to the same label. To prevent such a useless solution, Joachims (1999) introduced the balance constraint

$$\frac{\mathbf{1}'\hat{\mathbf{y}}_{\mathcal{U}}}{N-l} = \frac{\mathbf{1}'\mathbf{y}_{\mathcal{L}}}{l},$$

where $\hat{\mathbf{y}} = [\hat{y}_1, \cdots, \hat{y}_N]'$ is the vector of learned labels on both labeled and unlabeled examples, $\mathbf{y}_{\mathcal{L}} = [y_1, \ldots, y_l]'$, and $\hat{\mathbf{y}}_{\mathcal{U}} = [\hat{y}_{l+1}, \ldots, \hat{y}_N]'$. Let $\Omega = \frac{1}{2}\|\mathbf{w}\|^2$ and $\ell_f(\mathcal{D})$ be the sum of hinge loss values on both labeled and unlabeled data, Equation (2) leads to

$$\begin{aligned} \min_{\hat{\mathbf{y}} \in \mathcal{B}} \min_{\mathbf{w}, \xi} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C_1 \sum_{i=1}^{l} \xi_i + C_2 \sum_{i=l+1}^{N} \xi_i \\ \text{s.t.} \quad & \hat{y}_i \mathbf{w}' \phi(\mathbf{x}_i) \geq 1 - \xi_i, \ \ \xi_i \geq 0, \ \ i = 1 \ldots, N, \end{aligned}$$

where $\mathcal{B} = \{\hat{\mathbf{y}} \mid \hat{\mathbf{y}} = [\hat{\mathbf{y}}_{\mathcal{L}}; \hat{\mathbf{y}}_{\mathcal{U}}], \hat{\mathbf{y}}_{\mathcal{L}} = \mathbf{y}_{\mathcal{L}}, \hat{\mathbf{y}}_{\mathcal{U}} \in \{\pm 1\}^{N-l}; \frac{\mathbf{1}'\hat{\mathbf{y}}_{\mathcal{U}}}{N-l} = \frac{\mathbf{1}'\mathbf{y}_{\mathcal{L}}}{l}\}$, and $C_1, C_2$ trade off model complexity and empirical losses on the labeled and unlabeled data, respectively. The inner minimization problem can be rewritten in its dual, as:

$$\min_{\hat{\mathbf{y}} \in \mathcal{B}} \max_{\alpha \in \mathcal{A}} \quad G(\alpha, \hat{\mathbf{y}}) := \mathbf{1}'\alpha - \frac{1}{2}\alpha'\Big(\mathbf{K} \odot \hat{\mathbf{y}}\hat{\mathbf{y}}'\Big)\alpha, \tag{11}$$

where $\alpha = [\alpha_1, \ldots, \alpha_N]'$ is the vector of dual variables, and $\mathcal{A} = \{\alpha \;\big|\; C_1 \geq \alpha_i \geq 0, C_2 \geq \alpha_j \geq 0, i \in \mathcal{L}, j \in \mathcal{U}\}$.

Using Proposition 1, we have

$$\min_{\mu \in \mathcal{M}} \max_{\alpha \in \mathcal{A}} \; \mathbf{1}'\alpha - \frac{1}{2}\alpha'\Big(\sum_{t:\hat{\mathbf{y}}_t \in \mathcal{B}} \mu_t \mathbf{K} \odot \hat{\mathbf{y}}_t\hat{\mathbf{y}}_t'\Big)\alpha, \tag{12}$$

which is a convex relaxation of Equation (11). Note that $G(\alpha, \hat{\mathbf{y}})$ can be rewritten as $\bar{G}(\alpha, \mathbf{M}_{\mathbf{y}}) = \mathbf{1}'\alpha - \frac{1}{2}\alpha'\Big(\mathbf{K} \odot \mathbf{M}_{\mathbf{y}}\Big)\alpha$, where $\bar{G}$ is concave in $\alpha$ and linear in $\mathbf{M}_{\mathbf{y}}$. Hence, according to Theorem 1, WELLSVM is at least as tight as the SDP relaxations in Xu and Schuurmans (2005) and De Bie and Cristianini (2006).

Notice the similarity with standard SVM, which involves a single kernel matrix $\mathbf{K} \odot \hat{\mathbf{y}}\hat{\mathbf{y}}'$. Hence, Equation (12) can be regarded as *multiple kernel learning* (MKL) (Lanckriet et al., 2004), where the target kernel matrix is a convex combination of $|\mathcal{B}|$ base kernel matrices $\{\mathbf{K} \odot \hat{\mathbf{y}}_t\hat{\mathbf{y}}_t'\}_{t:\hat{\mathbf{y}}_t \in \mathcal{B}}$, each of which is constructed from a feasible label vector $\hat{\mathbf{y}}_t \in \mathcal{B}$.

#### 4.1.1 ALGORITHM

From Section 3, the cutting plane algorithm is used to solve Equation (12). There are two important issues that have to be addressed in the use of cutting plane algorithms. First, how to efficiently solve the MKL optimization problem? Second, how to efficiently find a violated $\hat{\mathbf{y}}$? These will be addressed in Sections 4.1.2 and 4.1.3, respectively.

#### 4.1.2 MULTIPLE LABEL-KERNEL LEARNING

In recent years, a lot of efforts have been devoted on efficient MKL approaches. Lanckriet et al. (2004) first proposed the use of quadratically constrained quadratic programming (QCQP) in MKL. Bach et al. (2004) showed that an approximate solution can be efficiently obtained by using sequential minimization optimization (SMO) (Platt, 1999). Recently, Sonnenburg et al. (2006) proposed a semi-infinite linear programming (SILP) formulation which allows MKL to be iteratively solved with standard SVM solver and linear programming. Rakotomamonjy et al. (2008) proposed a weighted 2-norm regularization with additional constraints on the kernel weights to encourage a sparse kernel combination. Xu et al. (2009) proposed the use of the extended level method to improve its convergence, which is further refined by the MKLGL algorithm (Xu et al., 2010). Extension to nonlinear MKL combinations is also studied recently in Kloft et al. (2009).

Unlike standard MKL problems which try to find the optimal kernel function/matrix for a given set of labels, here, we have to find the optimal label kernel matrix. In this paper, we use an adaptation of the MKLGL algorithm (Xu et al., 2010) to solve this multiple label-kernel learning (MLKL) problem. More specifically, suppose that the current working set is $\mathcal{C} = \{\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_T\}$. Note that the feature map corresponding to the base kernel matrix $\mathbf{K} \odot \hat{\mathbf{y}}_t\hat{\mathbf{y}}_t'$ is $\mathbf{x}_i \mapsto \hat{y}_{ti}\phi(\mathbf{x}_i)$. The MKL problem

in Equation (12) thus corresponds to the following primal optimization problem:

$$\begin{aligned} \min_{\mu\in\mathcal{M},\mathbf{W}=[\mathbf{w}_1,\ldots,\mathbf{w}_T],\xi} \quad & \frac{1}{2}\sum_{t=1}^{T}\frac{1}{\mu_t}||\mathbf{w}_t||^2 + C_1\sum_{i=1}^{l}\xi_i + C_2\sum_{i=l+1}^{N}\xi_i \\ \text{s.t.} \quad & \sum_{t=1}^{T}\hat{y}_{ti}\mathbf{w}_t'\phi(\mathbf{x}_i) \geq 1-\xi_i,\ \ \xi_i \geq 0,\ \ i=1,\ldots,N. \end{aligned} \tag{13}$$

It is easy to verify that its dual can be written as

$$\min_{\mu\in\mathcal{M}}\max_{\alpha\in\mathcal{A}} \quad \mathbf{1}'\alpha - \tfrac{1}{2}\alpha'\left(\textstyle\sum_{t=1}^{T}\mu_t\mathbf{K}\odot\hat{\mathbf{y}}_t\hat{\mathbf{y}}_t'\right)\alpha,$$

which is the same as Equation (12). Following MKLGL, we can solve Equation (12) (or, equivalently, Equation (13)) by iterating the following two steps until convergence.

1. Fix the mixing coefficients $\mu$ of the base kernel matrices and solve Equation (13). By setting $\tilde{\mathbf{w}} = [\frac{\mathbf{w}_1}{\sqrt{\mu_1}},\ldots,\frac{\mathbf{w}_T}{\sqrt{\mu_T}}]'$, $\tilde{\mathbf{x}}_i = [\sqrt{\mu_1}\phi(\mathbf{x}_i), \sqrt{\mu_2}\hat{y}_{1i}\hat{y}_{2i}\phi(\mathbf{x}_i),\ldots,\sqrt{\mu_T}\hat{y}_{1i}\hat{y}_{Ti}\phi(\mathbf{x}_i)]'$ and $\tilde{\mathbf{y}} = \hat{\mathbf{y}}_1$, Equation (13) can be rewritten as

$$\begin{aligned} \min_{\tilde{\mathbf{w}},\xi} \quad & \frac{1}{2}||\tilde{\mathbf{w}}||^2 + C_1\sum_{i=1}^{l}\xi_i + C_2\sum_{i=l+1}^{N}\xi_i \\ \text{s.t.} \quad & \tilde{y}_i\tilde{\mathbf{w}}'\tilde{\mathbf{x}}_i \geq 1-\xi_i,\ \ \xi_i \geq 0,\ \ i=1,\ldots,N, \end{aligned}$$

   which is similar to the primal of the standard SVM and can be efficiently handled by state-of-the-art SVM solvers.

2. Fix $\mathbf{w}_t$'s and update $\mu$ in closed-form, as

$$\mu_t = \frac{\|\mathbf{w}_t\|}{\sum_{t'=1}^{T}\|\mathbf{w}_{t'}\|},\ \ t=1,\ldots,T.$$

In our experiments, this always converges in fewer than 100 iterations. With the use of warm-start, even faster convergence can be expected.

#### 4.1.3 FINDING A VIOLATED LABEL ASSIGNMENT

The following optimization problem corresponds to finding the most violated $\hat{\mathbf{y}}$

$$\min_{\hat{\mathbf{y}}\in\mathcal{B}} \quad G(\alpha,\hat{\mathbf{y}}) = \mathbf{1}'\alpha - \frac{1}{2}\alpha'\left(\mathbf{K}\odot\hat{\mathbf{y}}\hat{\mathbf{y}}'\right)\alpha. \tag{14}$$

The first term in the objective does not relate to $\hat{\mathbf{y}}$, so Equation (14) is rewritten as

$$\max_{\hat{\mathbf{y}}\in\mathcal{B}} \ \frac{1}{2}\alpha'\left(\mathbf{K}\odot\hat{\mathbf{y}}\hat{\mathbf{y}}'\right)\alpha.$$

However, this is a concave QP and cannot be solved efficiently. Note that while the use of the most violated constraint may lead to faster convergence, the cutting plane algorithm only requires the

addition of a violated constraint at each iteration (Kelley, 1960; Tsochantaridis et al., 2006). Hence, we propose in the following a simple and efficient method for finding a violated label assignment.

Consider the following equivalent problem:

$$\max_{\hat{\mathbf{y}} \in \mathcal{B}} \hat{\mathbf{y}}' \mathbf{H} \hat{\mathbf{y}}, \tag{15}$$

where $\mathbf{H} = \mathbf{K} \odot (\alpha\alpha')$ is a psd matrix. Let $\bar{\mathbf{y}} \in \mathcal{C}$ be the following suboptimal solution of Equation (15)

$$\bar{\mathbf{y}} = \arg\max_{\hat{\mathbf{y}} \in \mathcal{C}} \hat{\mathbf{y}}' \mathbf{H} \hat{\mathbf{y}}.$$

Consider an optimal solution of the following optimization problem

$$\mathbf{y}^* = \operatorname{argmax}_{\hat{\mathbf{y}} \in \mathcal{B}} \hat{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}. \tag{16}$$

We have the following proposition.

**Proposition 4** *$\mathbf{y}^*$ is a violated label assignment if $\bar{\mathbf{y}}' \mathbf{H} \mathbf{y}^* \neq \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}$.*

**Proof** From $\hat{\mathbf{y}}' \mathbf{H} \mathbf{y}^* \neq \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}$, we have $\mathbf{y}^* \neq \bar{\mathbf{y}}$. Suppose that $(\mathbf{y}^*)' \mathbf{H} \mathbf{y}^* \leq \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}$, then $(\mathbf{y}^*)' \mathbf{H} \mathbf{y}^* + \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}} - 2(\mathbf{y}^*)' \mathbf{H} \bar{\mathbf{y}} \leq 2\bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}} - 2(\mathbf{y}^*)' \mathbf{H} \bar{\mathbf{y}} < 0$ which contradicts with $(\mathbf{y}^*)' \mathbf{H} \mathbf{y}^* + \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}} - 2(\mathbf{y}^*)' \mathbf{H} \bar{\mathbf{y}} = (\mathbf{y}^* - \bar{\mathbf{y}})' \mathbf{H} (\mathbf{y}^* - \bar{\mathbf{y}}) \geq 0$. So, $(\mathbf{y}^*)' \mathbf{H} \mathbf{y}^* > \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}$ which indicates $\mathbf{y}^*$ is a violated label assignment. ■

As for solving Equation (16), it is a integer linear program for $\hat{\mathbf{y}}$. We can rewrite this as

$$\begin{aligned} \max_{\hat{\mathbf{y}}} \quad & \mathbf{r}' \hat{\mathbf{y}} = \mathbf{r}'_{\mathcal{L}} \hat{\mathbf{y}}_{\mathcal{L}} + \mathbf{r}'_{\mathcal{U}} \hat{\mathbf{y}}_{\mathcal{U}} \\ \text{s.t.} \quad & \hat{\mathbf{y}}_{\mathcal{L}} = \mathbf{y}_{\mathcal{L}}, \hat{\mathbf{y}}_{\mathcal{U}} \in \{\pm 1\}^{N-l}, \frac{\mathbf{1}' \hat{\mathbf{y}}_{\mathcal{U}}}{N-l} = \frac{\mathbf{1}' \mathbf{y}_{\mathcal{L}}}{l}, \end{aligned} \tag{17}$$

where $\mathbf{r} = \mathbf{H} \bar{\mathbf{y}}$. Since $\hat{\mathbf{y}}_{\mathcal{L}}$ is constant, we have the following proposition.

**Proposition 5** *At optimality, $\hat{y}_i \geq \hat{y}_j$ if $r_i > r_j$, $i, j \in \mathcal{U}$.*

**Proof** Assume, to the contrary, that the optimal $\hat{\mathbf{y}}$ does not have the same sorted order as $\mathbf{r}$. Then, there are two label vectors $\hat{y}_i$ and $\hat{y}_j$, with $r_i > r_j$ but $\hat{y}_i < \hat{y}_j$. Then $r_i \hat{y}_i + r_j \hat{y}_j < r_i \hat{y}_j + r_j \hat{y}_i$ as $(r_i - r_j)(\hat{y}_i - \hat{y}_j) < 0$. Thus, $\hat{\mathbf{y}}$ is not optimal, a contradiction. ■

Thus, with Proposition 5, we can solve Equation (17) by first sorting in ascending order. The label assignment of $\hat{y}_i$'s aligns with the sorted values of $r_i$'s for $i \in \mathcal{U}$. To satisfy the balance constraint $\frac{\mathbf{1}' \hat{\mathbf{y}}_{\mathcal{U}}}{N-l} = \frac{\mathbf{1}' \mathbf{y}_{\mathcal{L}}}{l}$, the first $\left\lceil \frac{1}{2} \left( (N-l)(1 - \frac{1}{l} \mathbf{1}' \mathbf{y}_{\mathcal{L}}) \right) \right\rceil$ of $\hat{y}_i$'s are assigned $-1$, while the last $(N-l) - \left\lceil \frac{1}{2} \left( (N-l)(1 - \frac{1}{l} \mathbf{1}' \mathbf{y}_{\mathcal{L}}) \right) \right\rceil$ of them are assigned 1. Therefore, the label assignment in problem Equation (17) can be determined exactly and efficiently by sorting.

To find a violated label, we first get the $\bar{\mathbf{y}} \in \mathcal{C}$, which takes $O(N^2)$ (resp. $O(N)$) time when a nonlinear (resp. linear)[3] kernel is used; next we obtain the $\mathbf{y}^*$ in Equation (16), which takes

3. When the linear kernel is used, Equation (15) can be rewritten as $\max_{\hat{\mathbf{y}} \in \mathcal{C}} (\alpha \odot \hat{\mathbf{y}})' \mathbf{X}' \mathbf{X} (\alpha \odot \hat{\mathbf{y}})$, where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$. Hence, one can first compute $\mathbf{o} = \mathbf{X}(\alpha \odot \hat{\mathbf{y}})$ and then compute $\mathbf{o}'\mathbf{o}$. This takes a total of $O(N)$ time. A similar trick can be used in checking if $\mathbf{y}^*$ is a violated label assignment.

$O(N\log N)$ time; and finally check if $\mathbf{y}^*$ is a violated label assignment using Proposition 4, which takes $O(N^2)$ (resp. $O(N)$) time for a nonlinear (resp. linear) kernel. In total, this takes $O(N^2)$ (resp. $O(N\log N)$) time for nonlinear (resp. linear) kernel. Therefore, our proposal is computationally efficient.

Finally, after finishing the training process, we use $f(\mathbf{x}) = \sum_{t=1}^{T} \mathbf{w}_t' \phi(\mathbf{x})$ as the prediction function. Algorithm 2 summarizes the pseudocode of WELLSVM for semi-supervised learning.

**Algorithm 2** WELLSVM for semi-supervised learning.

1: Initialize $\hat{\mathbf{y}}$ and $\mathcal{C} = \emptyset$.
2: **repeat**
3: Update $\mathcal{C} \leftarrow \{\mathbf{y}^*\} \bigcup \mathcal{C}$.
4: Obtain the optimal $\{\mu, \mathbf{W}\}$ or $\alpha$ from Equation (13).
5: Find the optimal solution $\mathbf{y}^*$ of Equation (16).
6: **until** $G(\alpha, \mathbf{y}^*) > \min_{\mathbf{y}\in\mathcal{C}} G(\alpha, \mathbf{y}) - \varepsilon$ or the decrease of objective value is smaller than a threshold.
7: Output $f(\mathbf{x}) = \sum_{t=1}^{T} \mathbf{w}_t' \phi(\mathbf{x})$ as our prediction function.

### 4.2 Multi-Instance Learning

In this section, we consider the second weakly labeled learning problem, namely, multi-instance learning (MIL), where examples are bags containing multiple instances. More formally, we have a data set $\mathcal{D} = \{\mathbf{B}_i, y_i\}_{i=1}^m$, where $\mathbf{B}_i = \{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,m_i}\}$ is the input bag, $y_i \in \{\pm 1\}$ is the output and $m$ is the number of bags. Without loss of generality, we assume that the positive bags are ordered before the negative bags, that is, $y_i = 1$ for all $1 \le i \le p$ and $-1$ otherwise. Here, $p$ and $m-p$ are the numbers of positive and negative bags, respectively. In traditional MIL, a bag is labeled positive if it contains at least one key (or positive) instance, and negative otherwise. Thus, we only have the bag labels available, while the instance labels are only implicitly known.

Identification of the key instances from positive bags can be very useful in CBIR. Specifically, in CBIR, the whole image (bag) can be represented by multiple semantic regions (instances). Explicit identification of the regions of interest (ROIs) can help the user in recognizing images he/she wants quickly especially when the system returns a large amount of images. Consequently, the problem of determining whether a region is ROI can be posed as finding the key instances in MIL.

Traditional MIL implies that the label of a bag is determined by its most representative key instance, that is, $f(\mathbf{B}_i) = \max\{f(\mathbf{x}_{i,1}), \cdots, f(\mathbf{x}_{i,m_i})\}$. Let $\Omega = \frac{1}{2}\|\mathbf{w}\|_2^2$ and $\ell_f(\mathcal{D})$ be the sum of hinge losses on the bags, Equation (2) then leads to the MI-SVM proposed in Andrews et al. (2003):

$$\begin{aligned} \min_{\mathbf{w},\xi} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C_1 \sum_{i=1}^{p} \xi_i + C_2 \sum_{i=p+1}^{m} \xi_i \\ \text{s.t.} \quad & y_i \max_{1\le j\le m_i} \mathbf{w}'\phi(\mathbf{x}_{i,j}) \ge 1 - \xi_i,\ \ \xi_i \ge 0,\ \ i = 1, \ldots, m. \end{aligned} \tag{18}$$

Here, $C_1$ and $C_2$ trade off the model complexity and empirical losses on the positive and negative bags, respectively.

For a positive bag $\mathbf{B}_i$, we use the binary vector $\mathbf{d}_i = [d_{i,1}, \cdots, d_{i,m_i}]' \in \{0,1\}^{m_i}$ to indicate which instance in $\mathbf{B}_i$ is its key instance. Following the traditional MIL setup, we assume that each pos-

itive bag has only one key instance,[4] and so $\sum_{j=1}^{m_i} d_{i,j} = 1$. In the following, let $\mathbf{s} = [\mathbf{d}_1; \ldots; \mathbf{d}_p]$, and $\Delta$ be its domain. Moreover, note that $\max_{1 \le j \le m_i} \mathbf{w}'\phi(\mathbf{x}_{i,j})$ in Equation (18) can be written as $\max_{\mathbf{d}_i} \sum_{j=1}^{m_i} d_{i,j}\mathbf{w}'\phi(\mathbf{x}_{i,j})$.

For a negative bag $\mathbf{B}_i$, all its instances are negative and the corresponding constraint Equation (18) can be replaced by $-\mathbf{w}'\phi(\mathbf{x}_{i,j}) \ge 1 - \xi_i$ for every instance $\mathbf{x}_{i,j}$ in $\mathbf{B}_i$. Moreover, we relax the problem by allowing the slack variables $\xi_i$'s to be different for different instances in $\mathbf{B}_i$. This leads to a set of slack variables $\{\xi_{s(i,j)}\}_{i=p+1,\ldots,m; j=1,\ldots,m_i}$, where the indexing function $s(i,j) = J_{i-1} - J_p + j + p$ ranges from $p+1$ to $q = N - J_p + p$ and $J_i = \sum_{t=1}^{i} m_t$ ($J_0$ is set to 0). Combining all these together, Equation (18) can be rewritten as

$$\begin{aligned} \min_{\mathbf{s}\in\Delta} \min_{\mathbf{w},\xi} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C_1 \sum_{i=1}^{p} \xi_i + C_2 \sum_{i=p+1}^{m} \sum_{j=1}^{m_i} \xi_{s(i,j)} \\ \text{s.t.} \quad & \sum_{j=1}^{m_i} \mathbf{w}' d_{i,j} \phi(\mathbf{x}_{i,j}) \ge 1 - \xi_i, \ \ \xi_i \ge 0, \ \ i = 1, \ldots, p, \\ & -\mathbf{w}'\phi(\mathbf{x}_{i,j}) \ge 1 - \xi_{s(i,j)}, \ \ \xi_{s(i,j)} \ge 0, \ \ i = p+1, \ldots, m, j = 1, \ldots, m_i. \end{aligned}$$

The inner minimization problem is usually written in its dual, as

$$\max_{\alpha\in\mathcal{A}} \quad G(\alpha, \mathbf{s}) = \mathbf{1}'\alpha - \tfrac{1}{2}(\alpha \odot \hat{\mathbf{y}})' \left(\mathbf{K}^{\mathbf{s}}\right) (\alpha \odot \hat{\mathbf{y}}), \tag{19}$$

where $\alpha = [\alpha_1, \ldots, \alpha_q]' \in \mathbb{R}^q$ is the vector of dual variables, $\mathcal{A} = \{\alpha \mid C_1 \ge \alpha_i \ge 0, C_2 \ge \alpha_j \ge 0, i = 1, \ldots, p; j = p+1, \ldots, q\}$, $\hat{\mathbf{y}} = [\mathbf{1}_p, -\mathbf{1}_{q-p}] \in \mathbb{R}^q$, $\mathbf{K}^{\mathbf{s}} \in \mathbb{R}^{q \times q}$ is the kernel matrix where $\mathbf{K}^{\mathbf{s}}_{ij} = (\psi^{\mathbf{s}}_i)'(\psi^{\mathbf{s}}_j)$ with

$$\psi^{\mathbf{s}}_i = \begin{cases} \sum_{j=1}^{m_i} d_{i,j} \phi(\mathbf{x}_{i,j}) & i = 1, \ldots, p, \\ \phi(\mathbf{x}_{s(i,j)}) & i = p+1, \ldots, m; j = 1, \ldots, m_i. \end{cases} \tag{20}$$

Thus, Equation (19) is a mixed-integer programming problem. With Proposition 1, we have

$$\min_{\mu\in\mathcal{M}} \max_{\alpha\in\mathcal{A}} \ \mathbf{1}'\alpha - \frac{1}{2}(\alpha \odot \hat{\mathbf{y}})' \sum_{t:\mathbf{s}_t\in\Delta} \left(\mu_t \mathbf{K}^{\mathbf{s}_t}\right) (\alpha \odot \hat{\mathbf{y}}), \tag{21}$$

which is a convex relaxation of Equation (19).

### 4.2.1 ALGORITHM

Similar to semi-supervised learning, the cutting plane algorithm is used for solving Equation (21). Recall that there are two issues in the use of cutting-plane algorithms, namely, efficient multiple label-kernel learning and the finding of a violated label assignment. For the first issue, suppose that the current $\mathcal{C}$ is $\{\mathbf{s}_1, \ldots, \mathbf{s}_T\}$, the MKL problem in Equation (21) corresponds to the following

4. Sometimes, one can allow for more than one key instances in a positive bag (Wang et al., 2008; Xu and Frank, 2004; Zhou and Zhang, 2007; Zhou et al., 2012). The proposed method can be extended to this case by setting $\sum_{j=1}^{m_i} d_{i,j} = v$, where $v$ is the known number of key instances.

primal problem:

$$\begin{aligned}\min_{\mu\in\mathcal{M},\mathbf{W}=[\mathbf{w}_1;\ldots;\mathbf{w}_T],\xi} \quad & \frac{1}{2}\sum_{t=1}^{T}\frac{1}{\mu_t}||\mathbf{w}_t||^2 + C_1\sum_{i=1}^{p}\xi_i + C_2\sum_{i=p+1}^{m}\sum_{j=1}^{m_i}\xi_{s(i,j)} \\ \text{s.t.} \quad & \sum_{t=1}^{T}\left(\sum_{j=1}^{m_i}\mathbf{w}_t' d_{i,j}^t\phi(\mathbf{x}_{i,j})\right) \geq 1-\xi_i,\ \xi_i\geq 0,\ i=1,\ldots,p, \\ & -\sum_{t=1}^{T}\mathbf{w}_t'\phi(\mathbf{x}_{s(i,j)}) \geq 1-\xi_{s(i,j)},\ \xi_{s(i,j)}\geq 0,\ i=p+1,\ldots,m;\ j=1,\ldots,m_i.\end{aligned} \tag{22}$$

Therefore, we can still apply the MKLGL algorithm to solve MKL problem in Equation (21) efficiently. As for the second issue, one needs to solve the following problem:

$$\min_{\mathbf{s}\in\Delta} \quad \mathbf{1}'\alpha - \frac{1}{2}(\alpha\odot\hat{\mathbf{y}})'\mathbf{K}^{\mathbf{s}}(\alpha\odot\hat{\mathbf{y}}),$$

which is equivalent to

$$\max_{\mathbf{s}\in\Delta} \quad \sum_{i,j=1}^{q}\alpha_i\alpha_j\hat{y}_i\hat{y}_j(\psi_i^{\mathbf{s}})'(\psi_j^{\mathbf{s}}).$$

According to the definition of $\psi$ in Equation (20), this can be rewritten as

$$\max_{\mathbf{s}\in\Delta} \left\|\sum_{i=1}^{p}\alpha_i\sum_{j=1}^{m_i}d_{i,j}\phi(\mathbf{x}_{i,j}) - \sum_{i=p+1}^{m}\sum_{j=1}^{m_i}\alpha_{s(i,j)}\phi(\mathbf{x}_{s(i,j)})\right\|^2,$$

which can be reformulated as

$$\max_{\mathbf{s}\in\Delta} \quad \mathbf{s}'\mathbf{H}\mathbf{s}+\tau'\mathbf{s}, \tag{23}$$

where $\mathbf{H}\in\mathbb{R}^{J_p\times J_p}$ and $\tau\in\mathbb{R}^{J_p}$. Let $v(i,j)=J_{i-1}+j$, $i\in 1,\ldots,p$, $j\in 1,\ldots,m_i$, we have $H_{v(i,j),v(\hat{i},\hat{j})} = \alpha_i\alpha_{\hat{i}}\phi(\mathbf{x}_{i,j})'\phi(\mathbf{x}_{\hat{i},\hat{j}})$ and $\tau_{v(i,j)} = -2\alpha_i\phi(\mathbf{x}_{i,j})'(\sum_{i=p+1}^{m}\sum_{j=1}^{m_i}\alpha_{s(i,j)}\phi(\mathbf{x}_{s(i,j)}))$. It is easy to verify that $\mathbf{H}$ is psd.

Equation (23) is also a concave QP whose globally optimal solution, or equivalently the most violated $\mathbf{s}$, is intractable in general. In the following, we adapt a variant of the simple yet efficient method proposed in Section 4.1.3 to find a violated $\mathbf{s}$. Let $\bar{\mathbf{s}}\in\mathcal{C}$, where $\mathcal{C}=\{\mathbf{s}_1,\ldots,\mathbf{s}_T\}$, be the following suboptimal solution of Equation (23): $\bar{\mathbf{s}} = \text{argmax}_{\mathbf{s}\in\mathcal{C}}\mathbf{s}'\mathbf{H}\mathbf{s}+\tau'\mathbf{s}$. Let $\mathbf{s}^*$ be an optimal solution of the following optimization problem

$$\mathbf{s}^* = \text{argmax}_{\mathbf{s}\in\Delta}\mathbf{s}'\mathbf{H}\bar{\mathbf{s}} + \frac{\tau'\mathbf{s}}{2}. \tag{24}$$

**Proposition 6** *$\mathbf{s}^*$ is a violated label assignment when $(\mathbf{s}^*)'\mathbf{H}\bar{\mathbf{s}}+\frac{\tau'\mathbf{s}^*}{2} > \bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\frac{\tau'\bar{\mathbf{s}}}{2}$.*

**Proof** From $(\mathbf{s}^*)'\mathbf{H}\bar{\mathbf{s}}+\frac{\tau'\mathbf{s}^*}{2} > \bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\frac{\tau'\bar{\mathbf{s}}}{2}$, we have $\mathbf{s}^*\neq\bar{\mathbf{s}}$. Suppose that $(\mathbf{s}^*)'\mathbf{H}\mathbf{s}^*+\tau'\mathbf{s}^* \leq \bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}$. Then

$$\begin{aligned}&\left((\mathbf{s}^*)'\mathbf{H}\mathbf{s}^*+\tau'\mathbf{s}^*\right)+\left(\bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}\right)-\left[2(\mathbf{s}^*)'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}+\tau'\mathbf{s}^*\right] \\ \leq\ & 2\left[\bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}-(\mathbf{s}^*)'\mathbf{H}\bar{\mathbf{s}}-\frac{\tau'\bar{\mathbf{s}}}{2}-\frac{\tau'\mathbf{s}^*}{2}\right]<0,\end{aligned}$$

which contradicts

$$\begin{aligned}\left((\mathbf{s}^*)'\mathbf{H}\mathbf{s}^*+\tau'\mathbf{s}^*\right)+\left(\bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}\right)-\left[2(\mathbf{s}^*)'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}+\tau'\mathbf{s}^*\right] &= (\mathbf{s}^*-\bar{\mathbf{s}})'\mathbf{H}(\mathbf{s}^*-\bar{\mathbf{s}})\\ &\geq 0.\end{aligned}$$

So $(\mathbf{s}^*)'\mathbf{H}\mathbf{s}^*+\tau'(\mathbf{s}^*)>\bar{\mathbf{s}}'\mathbf{H}\bar{\mathbf{s}}+\tau'\bar{\mathbf{s}}$, which indicates that $\mathbf{s}^*$ is a violated label assignment. ■

Similar to Equation (16), Equation (24) is also a linear integer program but with different constraints. We now show that the optimal $\mathbf{s}^*$ in Equation (24) can still be solved via sorting. Notice that Equation (24) can be reformulated as

$$\begin{aligned}\max_{\mathbf{s}} \quad & \mathbf{r}'\mathbf{s} \\ \text{s.t.} \quad & \mathbf{1}'\mathbf{d}_i=1, \mathbf{d}_i\in\{0,1\}^{m_i}, i=1,\ldots,p,\end{aligned} \tag{25}$$

where $\mathbf{r}=\mathbf{H}\bar{\mathbf{s}}+\frac{\tau}{2}$. As can be seen, $\mathbf{d}_i$'s are decoupled in both the objective and constraints of Equation (25). Therefore, one can obtain its optimal solution by solving the $p$ subproblems individually

$$\begin{aligned}\max_{\mathbf{d}_i} \quad & \sum_{j=1}^{m_i} r_{J_{i-1}+j}d_{i,j} \\ \text{s.t.} \quad & \mathbf{1}'\mathbf{d}_i=1, \mathbf{d}_i\in\{0,1\}^{m_i}.\end{aligned}$$

It is evident that the optimal $\mathbf{d}_i$ can be obtained by assigning $d_{i,\hat{i}}=1$, where $\hat{i}$ is the index of the largest element among $[r_{J_{i-1}+1},\ldots,r_{J_{i-1}+m_i}]$, and the rest to zero. Similar to semi-supervised learning, the complexity to find a violated $\mathbf{s}$ scales as $O(N^2)$ (resp. $O(N\log N)$) when the nonlinear (resp. linear) kernel is used, and so is computationally efficient.

On prediction, each instance $\mathbf{x}$ can be treated as a bag, and its output from the WELLSVM is given by $f(\mathbf{x})=\sum_{t=1}^{T}\mathbf{w}_t'\phi(\mathbf{x})$. Algorithm 3 summarizes the pseudo codes of WELLSVM for multi-instance learning.

**Algorithm 3** WELLSVM for multi-instance learning.

1: Initialize $\mathbf{s}^*$ and $\mathcal{C}=\emptyset$.
2: **repeat**
3: Update $\mathcal{C}\leftarrow\{\mathbf{s}^*\}\bigcup\mathcal{C}$.
4: Obtain the optimal $\{\mu,\mathbf{W}\}$ or $\alpha$ from Equation (22).
5: Find the optimal solution $\mathbf{s}^*$ of Equation (24).
6: **until** $G(\alpha,\mathbf{s}^*)>\min_{\mathbf{s}\in\mathcal{C}}G(\alpha,\mathbf{s})-\varepsilon$ or the decrease of objective value is smaller than a threshold.
7: Output $f(\mathbf{x})=\sum_{t=1}^{T}\mathbf{w}_t'\phi(\mathbf{x})$ as the prediction function.

### 4.3 Clustering

In this section, we consider the third weakly labeled learning task, namely, clustering, where all the class labels are unknown. Similar to semi-supervised learning, one can obtain a trivially "optimal" solution with infinite margin by assigning all patterns to the same cluster. To prevent such a useless solution, Xu et al. (2005) introduced a class balance constraint

$$-\beta\leq\mathbf{1}'\hat{\mathbf{y}}\leq\beta,$$

where $\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_N]'$ is the vector of unknown labels, and $\beta \geq 0$ is a user-defined constant controlling the class imbalance.

Let $\Omega(f) = \frac{1}{2}\|\mathbf{w}\|_2^2$ and $\ell_f(\mathcal{D})$ be the sum of hinge losses on the individual examples. Equation (2) then leads to

$$\begin{aligned} \min_{\hat{\mathbf{y}} \in \mathcal{B}} \min_{\mathbf{w}, \xi} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{N} \xi_i \\ \text{s.t} \quad & \hat{y}_i \mathbf{w}' \phi(\mathbf{x}_i) \geq 1 - \xi_i, \ \ \xi_i \geq 0, \ \ i = 1 \ldots, N, \end{aligned} \tag{26}$$

where $\mathcal{B} = \{\hat{\mathbf{y}} \mid \hat{y}_i \in \{+1, -1\}, i = 1, \ldots, N; -\beta \leq \mathbf{1}'\hat{\mathbf{y}} \leq \beta\}$. The inner minimization problem is usually rewritten in its dual

$$\begin{aligned} \min_{\hat{\mathbf{y}} \in \mathcal{B}} \max_{\alpha} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j \left(\hat{y}_i \hat{y}_j \phi(\mathbf{x}_i)' \phi(\mathbf{x}_j)\right) \\ \text{s.t.} \quad & C \geq \alpha_i \geq 0, \ \ i = 1 \ldots, N, \end{aligned} \tag{27}$$

where $\alpha_i$ is the dual variable for each inequality constraint in Equation (26). Let $\alpha = [\alpha_1, \cdots, \alpha_N]'$ be the vector of dual variables, and $\mathcal{A} = \{\alpha \,\big|\, C\mathbf{1} \geq \alpha \geq \mathbf{0}\}$. Then Equation (27) can be rewritten in matrix form as

$$\min_{\hat{\mathbf{y}} \in \mathcal{B}} \max_{\alpha \in \mathcal{A}} \quad G(\alpha, \hat{\mathbf{y}}) := \mathbf{1}'\alpha - \frac{1}{2}\alpha'\Big(\mathbf{K} \odot \hat{\mathbf{y}}\hat{\mathbf{y}}'\Big)\alpha. \tag{28}$$

This, however, is still a mixed integer programming problem.

With Proposition 1, we have

$$\min_{\mu \in \mathcal{M}} \max_{\alpha \in \mathcal{A}} \ \mathbf{1}'\alpha - \frac{1}{2}\alpha'\Big(\sum_{t:\hat{\mathbf{y}}_t \in \mathcal{B}} \mu_t \mathbf{K} \odot \hat{\mathbf{y}}_t \hat{\mathbf{y}}_t'\Big)\alpha \tag{29}$$

as a convex relaxation of Equation (28). Note that $G(\alpha, \hat{\mathbf{y}})$ can be reformulated by $\bar{G}(\alpha, \mathbf{M}_\mathbf{y}) = \mathbf{1}'\alpha - \frac{1}{2}\alpha'\Big(\mathbf{K} \odot \mathbf{M}_\mathbf{y}\Big)\alpha$, where $\bar{G}$ is concave in $\alpha$ and linear in $\mathbf{M}_\mathbf{y}$. Hence, according to Theorem 1, WELLSVM is at least as tight as the SDP relaxation in Xu et al. (2005).

#### 4.3.1 ALGORITHM

The cutting plane algorithm can still be applied for clustering. Similar to semi-supervised learning, the MKL can be formulated as the following primal problem:

$$\begin{aligned} \min_{\mu \in \mathcal{M}, \mathbf{W} = [\mathbf{w}_1; \ldots; \mathbf{w}_T], \xi} \quad & \frac{1}{2}\sum_{t=1}^{T} \frac{1}{\mu_t}||\mathbf{w}_t||^2 + C\sum_{i=1}^{N} \xi_i \\ \text{s.t.} \quad & \sum_{t=1}^{T} \hat{y}_{ti} \mathbf{w}_t' \phi(\mathbf{x}_i) \geq 1 - \xi_i, \ \ \xi_i \geq 0, \ \ i = 1, \ldots, N, \end{aligned} \tag{30}$$

and its dual is

$$\min_{\mu \in \mathcal{M}} \max_{\alpha \in \mathcal{A}} \quad \mathbf{1}'\alpha - \tfrac{1}{2}\alpha'\Big(\textstyle\sum_{t=1}^{T} \mu_t \mathbf{K} \odot \hat{\mathbf{y}}_t \hat{\mathbf{y}}_t'\Big)\alpha,$$

which is the same as Equation (29). Therefore, MKLGL algorithm can still be applied for solving the MKL problem in Equation (29) efficiently.

As for finding a violated label assignment, let $\bar{\mathbf{y}} \in \mathcal{C}$ be

$$\bar{\mathbf{y}} = \arg\max_{\hat{\mathbf{y}} \in \mathcal{C}} \hat{\mathbf{y}}' \mathbf{H} \hat{\mathbf{y}},$$

where $\mathbf{H} = \mathbf{K} \odot (\alpha\alpha')$ is a positive semidefinite matrix. Consider an optimal solution of the following optimization problem

$$\mathbf{y}^* = \operatorname{argmax}_{\hat{\mathbf{y}} \in \mathcal{B}} \hat{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}. \tag{31}$$

With Proposition 4, we obtain that $\mathbf{y}^*$ is a violated label assignment if $\bar{\mathbf{y}}' \mathbf{H} \mathbf{y}^* \geq \bar{\mathbf{y}}' \mathbf{H} \bar{\mathbf{y}}$.

Note that Equation (31) is a linear program for $\hat{\mathbf{y}}$ and can be formulated as

$$\begin{aligned} \max_{\hat{\mathbf{y}}} \quad & \mathbf{r}' \hat{\mathbf{y}} \\ \text{s.t.} \quad & -\beta \leq \hat{\mathbf{y}}' \mathbf{1} \leq \beta, \hat{\mathbf{y}} \in \{-1, +1\}^N, \end{aligned} \tag{32}$$

where $\mathbf{r} = \mathbf{H} \bar{\mathbf{y}}$. From Proposition 5, we can solve Equation (32) by first sorting $r_i$'s in ascending order. The label assignment of $\hat{y}_i$'s aligns with the sorted values of $r_i$'s. To satisfy the balance constraint $-\beta \leq \mathbf{1}' \hat{\mathbf{y}} \leq \beta$, the first $\frac{N-\beta}{2}$ of $\hat{y}_i$'s are assigned $-1$, the last $\frac{N-\beta}{2}$ of them are assigned 1, and the rest $\hat{y}_i$'s are assigned $-1$ (resp. 1) if the corresponded $r_i$'s are negative (resp. non-negative). It is easy to verify that such an assignment satisfies the balance constraint and the objective $\mathbf{r}' \hat{\mathbf{y}}$ is maximized. Similar to semi-supervised learning, the complexity to find a violated label scales as $O(N^2)$ (resp. $O(N \log N)$) when the nonlinear (resp. linear) kernel is used, and so is computationally efficient. Finally, we use $f(\mathbf{x}) = \sum_{t=1}^{T} \mathbf{w}_t' \mathbf{x}$ as the prediction function. Algorithm 4 summarizes the pseudo codes of WELLSVM for clustering.

**Algorithm 4** WELLSVM for clustering.

1: Initialize $\hat{\mathbf{y}}$ and $\mathcal{C} = \emptyset$.
2: **repeat**
3: Update $\mathcal{C} \leftarrow \{\mathbf{y}^*\} \bigcup \mathcal{C}$.
4: Obtain the optimal $\{\mu, \mathbf{W}\}$ or $\alpha$ from Equation (30).
5: Find the optimal solution $\mathbf{y}^*$ of Equation (31).
6: **until** $G(\alpha, \mathbf{y}^*) > \min_{\mathbf{y} \in \mathcal{C}} G(\alpha, \mathbf{y}) - \varepsilon$ or the decrease of objective value is smaller than a threshold.
7: Output $f(\mathbf{x}) = \sum_{t=1}^{T} \mathbf{w}_t' \phi(\mathbf{x})$ as the prediction function.

## 5. Experiments

In this section, comprehensive evaluations are performed to verify the effectiveness of the proposed WELLSVM. Experiments are conducted on all the three aforementioned weakly labeled learning tasks: semi-supervised learning (Section 5.1), multi-instance learning (Section 5.2) and clustering (Section 5.3). For nonlinear kernel, the WELLSVM adapts $\nu$-SVM with square hinge loss (Tsang et al., 2006) and is implemented using the LIBSVM (Fan et al., 2005); For linear kernel, it adapts standard SVM without offset and is implemented using the LIBLINEAR (Hsieh et al., 2008). Experiments are run on a 3.20GHz Intel Xeon(R)2 Duo PC running Windows 7 with 8GB main memory.

For all the other methods that will be used for comparison, the default stopping criteria in the corresponding packages are used. For the WELLSVM, both the $\varepsilon$ and stopping threshold in Algorithm 1 are set to $10^{-3}$.

### 5.1 Semi-Supervised Learning

We first evaluate the WELLSVM on semi-supervised learning with a large collection of real-world data sets. 16 UCI data sets, which cover a wide range of properties, and 2 large-scale data sets[5] are used. Table 1 shows some statistics of these data sets.

| | Data | # Instances | # Features | | Data | # Instances | # Features |
|---|---|---|---|---|---|---|---|
| 1 | *Echocardiogram* | 132 | 8 | 10 | *Clean1* | 476 | 166 |
| 2 | *House* | 232 | 16 | 11 | *Isolet* | 600 | 51 |
| 3 | *Heart* | 270 | 9 | 12 | *Australian* | 690 | 42 |
| 4 | *Heart-stalog* | 270 | 13 | 13 | *Diabetes* | 768 | 8 |
| 5 | *Haberman* | 306 | 14 | 14 | *German* | 1,000 | 59 |
| 6 | *LiveDiscorders* | 345 | 6 | 15 | *Krvskp* | 3,196 | 36 |
| 7 | *Spectf* | 349 | 44 | 16 | *Sick* | 3,772 | 31 |
| 8 | *Ionosphere* | 351 | 34 | 17 | *real-sim* | 72,309 | 20,958 |
| 9 | *House-votes* | 435 | 16 | 18 | *rcv1* | 677,399 | 47,236 |

Table 1: Data sets used in the experiments.

#### 5.1.1 SMALL-SCALE EXPERIMENTS

For each UCI data set, 75% of the examples are randomly chosen for training, and the rest for testing. We investigate the performance of each approach with varying amount of labeled data (namely, 5%, 10% and 15% of all the labeled data). The whole setup is repeated 30 times and the average accuracies (with standard deviations) on the test set are reported.

We compare WELLSVM with 1) the standard SVM (using labeled data only), and three state-of-the-art semi-supervised SVMs ($S^3$VMs), namely 2) Transductive SVM (TSVM)[6] (Joachims, 1999); 3) Laplacian SVM (LapSVM)[7] (Belkin et al., 2006); and 4) UniverSVM (USVM)[8] (Collobert et al., 2006). Note that TSVM and USVM adopt the same objective as WELLSVM, but with different optimization strategies (local search and constrained convex-concave procedure, respectively). LapSVM is another $S^3$VM based on the manifold assumption (Belkin et al., 2006). The SDP-based $S^3$VMs (Xu and Schuurmans, 2005; De Bie and Cristianini, 2006) are not compared, as they do not converge after 3 hours on even the smallest data set (*Echocardiogram*).

Parameters of the different methods are set as follows. $C_1$ is fixed at 1 and $C_2$ is selected in the range $\{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$. The linear and Gaussian kernels are used for all SVMs, where the width $\sigma$ of the Gaussian kernel $k(\mathbf{x}, \hat{\mathbf{x}}) = \exp(-||\mathbf{x} - \hat{\mathbf{x}}||^2/2\sigma^2)$ is picked from $\{0.25\sqrt{\gamma}, 0.5\sqrt{\gamma}, \sqrt{\gamma}, 2\sqrt{\gamma}, 4\sqrt{\gamma}\}$, with $\gamma$ being the average distance between all instance pairs. The

5. Data sets can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`.
6. Transductive SVM can be found at `http://svmlight.joachims.org/`.
7. Laplacian SVM can be found at `http://manifold.cs.uchicago.edu/manifold_regularization/software.html`.
8. UniverSVM can be found at `http://mloss.org/software/view/19/`.

| Data | SVM | TSVM | LapSVM | USVM | WELLSVM |
|---|---|---|---|---|---|
| *Echocardiogram* | 0.80 ± 0.07 (2.5) | 0.74 ± 0.08 (4) | 0.64 ± 0.22 (5) | **0.81 ± 0.06** (1) | 0.80 ± 0.07 (2.5) |
| *House* | **0.90 ± 0.04** (3) | **0.90 ± 0.05** (3) | **0.90 ± 0.04** (3) | **0.90 ± 0.03** (3) | **0.90 ± 0.04** (3) |
| *Heart* | 0.70 ± 0.08 (5) | 0.75 ± 0.08 (3) | 0.73 ± 0.09 (4) | 0.76 ± 0.07 (2) | **0.77 ± 0.08** (1) |
| *Heart-statlog* | 0.73 ± 0.10 (4.5) | **0.75 ± 0.10** (1.5) | 0.74 ± 0.11 (3) | **0.75 ± 0.12** (1.5) | 0.73 ± 0.12 (4.5) |
| *Haberman* | 0.65 ± 0.07 (3) | 0.61 ± 0.06 (4) | 0.57 ± 0.11 (5) | **0.75 ± 0.05** (1.5) | **0.75 ± 0.05** (1.5) |
| *LiverDisorders* | 0.56 ± 0.05 (2) | 0.55 ± 0.05 (3.5) | 0.55 ± 0.05 (3.5) | **0.59 ± 0.05** (1) | 0.53 ± 0.07 (5) |
| *Spectf* | 0.73 ± 0.05 (2) | 0.68 ± 0.10 (4) | 0.61 ± 0.08 (5) | **0.74 ± 0.05** (1) | 0.70 ± 0.07 (3) |
| *Ionosphere* | 0.67 ± 0.06 (4) | **0.82 ± 0.11** (1) | 0.65 ± 0.05 (5) | 0.77 ± 0.07 (2) | 0.70 ± 0.08 (3) |
| *House-votes* | 0.88 ± 0.03 (3) | **0.89 ± 0.05** (1.5) | 0.87 ± 0.03 (4) | 0.83 ± 0.03 (5) | **0.89 ± 0.03** (1.5) |
| *Clean1* | 0.58 ± 0.06 (4) | 0.60 ± 0.08 (3) | 0.54 ± 0.05 (5) | **0.65 ± 0.05** (1) | 0.63 ± 0.07 (2) |
| *Isolet* | 0.97 ± 0.02 (3) | **0.99 ± 0.01** (1) | 0.97 ± 0.02 (3) | 0.70 ± 0.09 (5) | 0.97 ± 0.02 (3) |
| *Australian* | 0.79 ± 0.05 (4) | **0.82 ± 0.07** (1) | 0.78 ± 0.08 (5) | 0.80 ± 0.05 (3) | 0.81 ± 0.04 (2) |
| *Diabetes* | 0.67 ± 0.04 (4) | 0.67 ± 0.04 (4) | 0.67 ± 0.04 (4) | **0.70 ± 0.03** (1) | 0.69 ± 0.03 (2) |
| *German* | **0.70 ± 0.03** (2) | 0.69 ± 0.03 (4) | 0.62 ± 0.05 (5) | **0.70 ± 0.02** (2) | **0.70 ± 0.02** (2) |
| *Krvskp* | 0.91 ± 0.02 (3.5) | **0.92 ± 0.03** (1.5) | 0.80 ± 0.02 (5) | 0.91 ± 0.03 (3.5) | **0.92 ± 0.02** (1.5) |
| *Sick* | **0.94 ± 0.01** (2) | 0.89 ± 0.01 (5) | 0.90 ± 0.02 (4) | **0.94 ± 0.01** (2) | **0.94 ± 0.01** (2) |
| SVM: win/tie/loss | | 5/7/4 | 8/7/1 | 2/9/5 | **3/6/7** |
| ave. acc. | 0.763 | 0.767 | 0.723 | 0.770 | **0.778** |
| ave. rank | 3.2188 | 2.8125 | 4.2813 | 2.2188 | 2.4688 |

Table 2: Accuracies on the various data sets with 5% labeled examples. The best performance on each data set is bolded. The win/tie/loss counts (paired $t$-test at 95% significance level) are listed. The method with the largest number of (#wins - #losses) against SVM as well as the best average accuracy is also highlighted. Number in parentheses denotes the ranking (computed as in Demsar 2006) of each method on the data set.

initial label assignment of WELLSVM is obtained from the predictions of a standard SVM. For LapSVM, the number of nearest neighbors in the underlying data graph is selected from $\{3,5,7,9\}$. All parameters are determined by using the five-fold cross-validated accuracy.

Table 2 shows the results on the UCI data sets with 5% labeled examples. As can be seen, WELLSVM obtains highly competitive performance with the other methods, and achieves the best improvement against SVM in terms of both the counts of (#wins $-$ #loses) as well as average accuracy. The Friedman test (Demsar, 2006) shows that both WELLSVM and USVM perform significantly better than SVM at the 90% confidence level, while TSVM and LapSVM do not.

As can be seen, there are cases where unlabeled data cannot help for TSVM, USVM and WELLSVM. Besides the local minimum problem, another possible reason may be that there are multiple large margin separators coinciding well with labeled data and the labeled examples are too few to provide a reliable selection for these separators (Li and Zhou, 2011). Moreover, overall, LapSVM cannot obtain good performance, which may be due to that the manifold assumption does not hold on these data (Chapelle et al., 2006b).

Tables 3 and 4 show the results on the UCI data sets with 10% and 15% labeled examples, respectively. As can be seen, as the number of labeled examples increases, SVM gets much better performance. As a result, both TSVM and USVM cannot beat the SVM. On the other hand, the Friedman test shows that WELLSVM still performs significantly better than SVM with 10% labeled examples at the 90% confidence level. With 15% labeled examples, no $S^3$VM performs significantly better than SVM.

Figure 1 compares the average CPU time of WELLSVM with the other $S^3$VMs different numbers of labeled examples. As can be seen, TSVM is the slowest while USVM is the most efficient.

| Data | SVM | TSVM | LapSVM | USVM | WELLSVM |
|---|---|---|---|---|---|
| *Echocardiogram* | 0.81 ± 0.05 (2.5) | 0.76 ± 0.12 (4) | 0.69 ± 0.14 (5) | **0.82 ± 0.05** (1) | 0.81 ± 0.05 (2.5) |
| *House* | 0.90 ± 0.04 (2.5) | **0.92 ± 0.05** (1) | 0.89 ± 0.04 (4) | 0.83 ± 0.03 (5) | 0.90 ± 0.04 (2.5) |
| *Heart* | 0.76 ± 0.05 (3.5) | 0.75 ± 0.05 (5) | 0.76 ± 0.06 (3.5) | **0.78 ± 0.05** (1.5) | **0.78 ± 0.04** (1.5) |
| *Heart-statlog* | 0.79 ± 0.03 (4) | 0.74 ± 0.05 (5) | 0.80 ± 0.04 (2.5) | 0.80 ± 0.04 (2.5) | **0.81 ± 0.04** (1) |
| *Haberman* | **0.75 ± 0.04** (2) | 0.60 ± 0.07 (4.5) | 0.60 ± 0.07 (4.5) | **0.75 ± 0.04** (2) | **0.75 ± 0.04** (2) |
| *LiverDisorders* | **0.59 ± 0.06** (1) | 0.57 ± 0.05 (2.5) | 0.55 ± 0.06 (4) | 0.53 ± 0.06 (5) | 0.57 ± 0.05 (2.5) |
| *Spectf* | 0.74 ± 0.05 (2) | **0.76 ± 0.06** (1) | 0.64 ± 0.06 (5) | 0.72 ± 0.06 (3.5) | 0.72 ± 0.07 (3.5) |
| *Ionosphere* | 0.78 ± 0.07 (4) | **0.90 ± 0.04** (1) | 0.66 ± 0.06 (5) | 0.88 ± 0.05 (2) | 0.82 ± 0.05 (3) |
| *House-votes* | **0.92 ± 0.03** (1.5) | 0.91 ± 0.03 (3.5) | 0.88 ± 0.04 (5) | 0.91 ± 0.03 (3.5) | **0.92 ± 0.03** (1.5) |
| *Clean1* | 0.69 ± 0.05 (3.5) | 0.71 ± 0.05 (2) | 0.63 ± 0.07 (5) | **0.72 ± 0.05** (1) | 0.69 ± 0.04 (3.5) |
| *Isolet* | 0.99 ± 0.01 (2.5) | **1.00 ± 0.01** (1) | 0.96 ± 0.02 (4) | 0.52 ± 0.03 (5) | 0.99 ± 0.01 (2.5) |
| *Australian* | 0.81 ± 0.03 (5) | **0.84 ± 0.03** (1.5) | 0.82 ± 0.04 (4) | **0.84 ± 0.03** (1.5) | 0.83 ± 0.03 (3) |
| *Diabetes* | 0.70 ± 0.03 (4.5) | 0.70 ± 0.05 (4.5) | 0.71 ± 0.04 (3) | 0.72 ± 0.03 (2) | **0.74 ± 0.03** (1) |
| *German* | 0.67 ± 0.03 (3.5) | 0.67 ± 0.03 (3.5) | 0.66 ± 0.04 (5) | **0.70 ± 0.02** (1.5) | **0.70 ± 0.02** (1.5) |
| *Krvskp* | 0.93 ± 0.01 (3) | 0.93 ± 0.01 (3) | 0.86 ± 0.04 (5) | 0.93 ± 0.01 (3) | **0.94 ± 0.01** (1) |
| *Sick* | **0.93 ± 0.01** (2) | 0.89 ± 0.01 (5) | 0.92 ± 0.01 (4) | **0.93 ± 0.01** (2) | **0.93 ± 0.01** (2) |
| SVM: win/tie/loss | | 5/8/3 | 10/5/1 | 5/6/5 | **0/9/7** |
| avg. acc. | 0.799 | 0.789 | 0.753 | 0.774 | **0.807** |
| avg. rank | 2.9375 | 3.0000 | 4.2813 | 2.6250 | 2.1563 |

Table 3: Accuracies on the various data sets with 10% labeled examples.

| Data | SVM | TSVM | LapSVM | USVM | WELLSVM |
|---|---|---|---|---|---|
| *echocardiogram* | 0.83 ± 0.04 (2.5) | 0.76 ± 0.07 (4) | 0.75 ± 0.08 (5) | **0.85 ± 0** (1) | 0.83 ± 0.04 (2.5) |
| *house* | 0.92 ± 0.04 (2.5) | **0.94 ± 0.04** (1) | 0.83 ± 0.11 (5) | 0.91 ± 0.04 (4) | 0.92 ± 0.03 (2.5) |
| *heart* | 0.78 ± 0.06 (3) | 0.78 ± 0.05 (3) | **0.79 ± 0.05** (1) | 0.78 ± 0.07 (3) | 0.78 ± 0.06 (3) |
| *heart-statlog* | 0.76 ± 0.06 (2) | 0.74 ± 0.06 (4) | **0.79 ± 0.05** (1) | 0.73 ± 0.07 (5) | 0.75 ± 0.06 (3) |
| *haberman* | 0.72 ± 0.03 (3) | 0.62 ± 0.07 (5) | 0.63 ± 0.11 (4) | **0.74 ± 0** (1.5) | **0.74 ± 0** (1.5) |
| *liverDisorders* | **0.61 ± 0.05** (1) | 0.54 ± 0.06 (4) | 0.53 ± 0.07 (5) | 0.58 ± 0 (2) | 0.56 ± 0.06 (3) |
| *spectf* | 0.77 ± 0.03 (2) | **0.79 ± 0.04** (1) | 0.6 ± 0.1 (5) | 0.74 ± 0 (4) | 0.75 ± 0.06 (3) |
| *ionosphere* | 0.76 ± 0.04 (5) | **0.9 ± 0.04** (1) | 0.83 ± 0.04 (4) | 0.89 ± 0.04 (2) | 0.84 ± 0.03 (3) |
| *house-votes* | **0.92 ± 0.02** (1.5) | **0.92 ± 0.03** (1.5) | 0.9 ± 0.03 (3) | 0.83 ± 0.03 (5) | 0.89 ± 0.02 (4) |
| *clean1* | 0.71 ± 0.04 (4) | 0.74 ± 0.04 (2) | 0.63 ± 0.07 (5) | **0.76 ± 0.06** (1) | 0.72 ± 0.04 (3) |
| *isolet* | 0.98 ± 0.01 (3.5) | **0.99 ± 0.01** (1.5) | 0.98 ± 0.01 (3.5) | 0.54 ± 0.02 (5) | **0.99 ± 0.01** (1.5) |
| *australian* | **0.86 ± 0.02** (1.5) | 0.85 ± 0.03 (3) | 0.83 ± 0.02 (4.5) | 0.83 ± 0.03 (4.5) | **0.86 ± 0.03** (1.5) |
| *diabetes* | **0.75 ± 0.03** (1.5) | 0.73 ± 0.02 (3.5) | 0.73 ± 0.03 (3.5) | 0.72 ± 0.04 (5) | **0.75 ± 0.03** (1.5) |
| *german* | 0.71 ± 0.01 (2) | 0.7 ± 0.03 (3.5) | 0.68 ± 0.04 (5) | 0.7 ± 0.04 (3.5) | **0.72 ± 0.01** (1) |
| *krvskp* | **0.95 ± 0.01** (1.5) | 0.93 ± 0.01 (4) | 0.91 ± 0.01 (5) | 0.94 ± 0.01 (3) | **0.95 ± 0.01** (1.5) |
| *sick* | **0.94 ± 0** (2) | 0.9 ± 0.01 (4.5) | 0.9 ± 0.12 (4.5) | **0.94 ± 0** (2) | **0.94 ± 0** (2) |
| SVM: win/tie/loss | | 8/3/5 | 11/2/3 | 6/6/4 | **2/9/5** |
| avg. acc. | 0.809 | 0.801 | 0.771 | 0.780 | **0.811** |
| avg. rank | 2.4063 | 2.9063 | 4.0000 | 3.2188 | 2.3438 |

Table 4: Accuracies on various data sets with 15% labeled examples.

WELLSVM is comparable to LapSVM. Figure 2 shows the objective values of WELLSVM on five representative UCI data sets. We can observe that the number of iterations is always fewer than 25. As mentioned above, the SDP-based $S^3$VMs (Xu and Schuurmans, 2005; De Bie and Cristianini, 2006), in contrast, cannot converge in 3 hours even on the smallest data set *Echocardiogram*. Hence, WELLSVM scales much better than these SDP-based approaches.

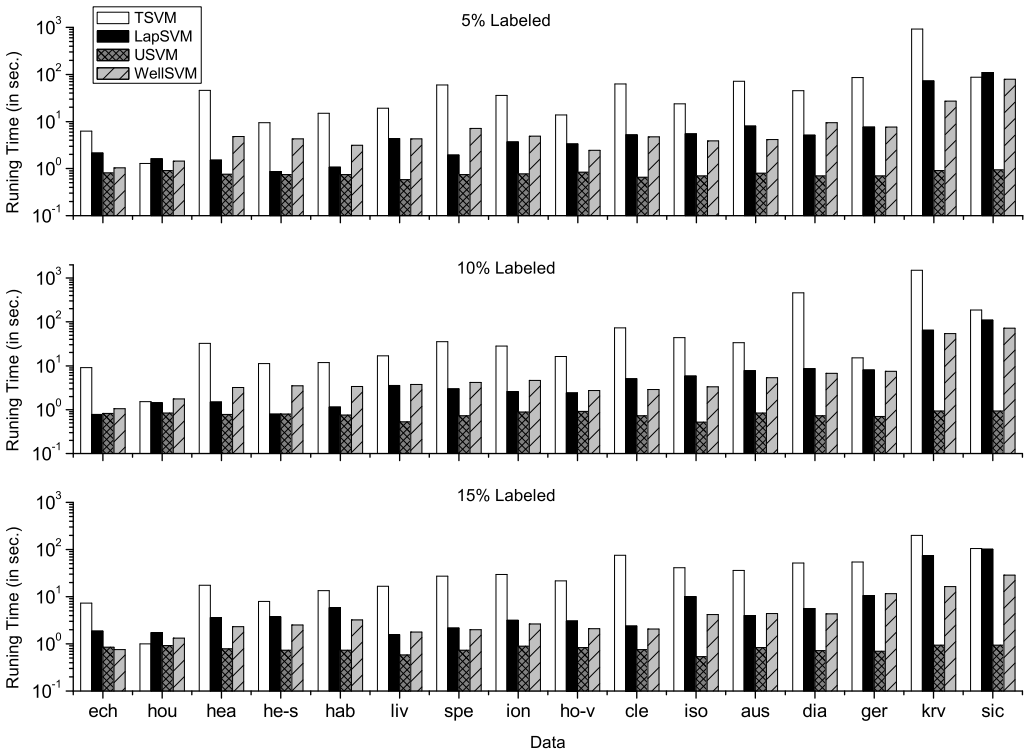



Figure 1: CPU time on the UCI data sets.

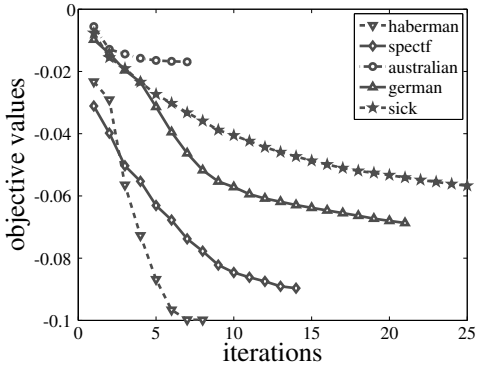



Figure 2: Number of WELLSVM iterations on the UCI data sets.

### 5.1.2 LARGE-SCALE EXPERIMENTS

In this section, we study the scalability of the proposed WELLSVM and other state-of-the-art approaches on two large data sets, *real-sim* and *RCV1*. The *real-sim* data has 20,958 features and 72,309 instances. while the *RCV1* data has 47,236 features and 677,399 instances. The linear kernel is used. The $S^3$VMs compared in Section 5.1.1 are for general kernels and cannot converge in 24 hours. Hence, to conduct a fair comparison, an efficient linear $S^3$VM solver, namely, SVMlin[9]

9. SVMlin can be found at `http://vikas.sindhwani.org/svmlin.html`.

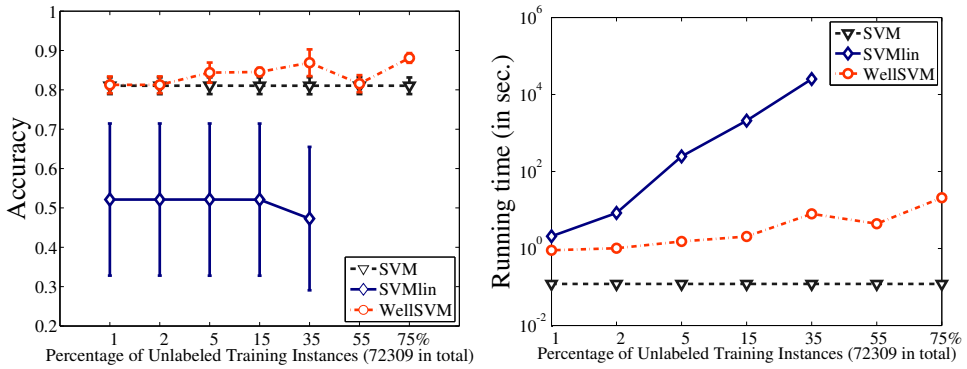



Figure 3: Semi-supervised learning results on the *real-sim* data with different amounts of unlabeled examples.

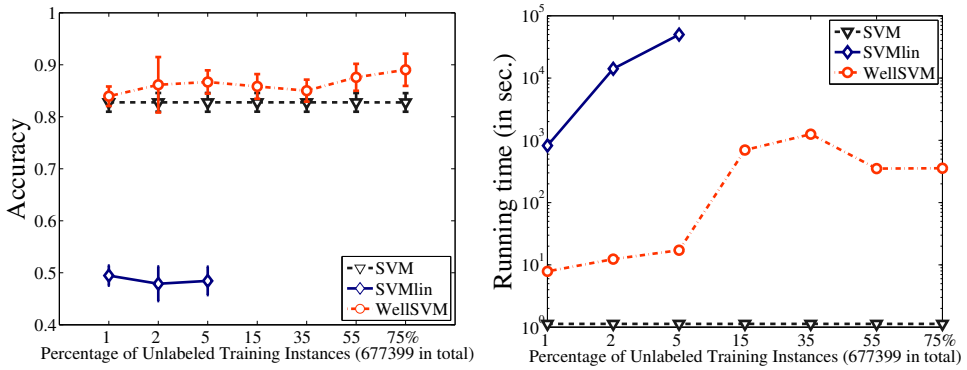



Figure 4: Semi-supervised learning results on the *RCV1* data with different number of unlabeled examples.

using deterministic annealing (Sindhwani and Keerthi, 2006), is employed. All the parameters are determined in the same manner as in Section 5.1.1.

In the first experiment, we study the performance at different numbers of unlabeled examples. Specifically, 1%, 2%, 5%, 15%, 35%, 55% and 75% of the data (with 50 of them labeled) are used for training, and 25% of the data are for testing. This is repeated 10 times and the average performance is reported.

Figure 3 shows the results. As can be seen, WELLSVM is always superior to SVMlin, and achieves highly competitive or even better accuracy than the SVM as the number of unlabeled examples increases. Moreover, WELLSVM is much faster than SVMlin. As the number of unlabeled examples increases, the difference becomes more prominent. This is mainly because SVMlin employs gradient descent while WELLSVM (which is based on LIBLINEAR (Hsieh et al., 2008)) uses coordinate descent, which is known to be one of the fastest solvers for large-scale linear SVMs (Shalev-Shwartz et al., 2007).

Figure 4 shows the results on the larger *RCV1* data set. As can be seen, WELLSVM obtains good accuracy at different numbers of unlabeled examples. More importantly, WELLSVM scales well on *RCV1*. For example, WELLSVM takes fewer than 1,000 seconds with more than 500,000 instances. On the other hand, SVMlin cannot converge in 24 hours when more than 5% examples are used for training.

| # of labeled examples | | 25 | 50 | 100 | 150 | 200 |
|---|---|---|---|---|---|---|
| *real-sim* | SVM | $0.78 \pm 0.03$ | $0.81 \pm 0.02$ | $0.84 \pm 0.02$ | $0.86 \pm 0.01$ | $0.88 \pm 0.01$ |
| | WELLSVM | $\mathbf{0.81 \pm 0.08}$ | $\mathbf{0.84 \pm 0.02}$ | $\mathbf{0.89 \pm 0.01}$ | $\mathbf{0.9 \pm 0.01}$ | $\mathbf{0.91 \pm 0.01}$ |
| *rcv1* | SVM | $0.77 \pm 0.03$ | $0.83 \pm 0.01$ | $0.87 \pm 0.01$ | $0.89 \pm 0.01$ | $0.9 \pm 0.01$ |
| | WELLSVM | $\mathbf{0.83 \pm 0.03}$ | $\mathbf{0.9 \pm 0.02}$ | $\mathbf{0.91 \pm 0.01}$ | $\mathbf{0.92 \pm 0.01}$ | $\mathbf{0.93 \pm 0.01}$ |

Table 5: Accuracy (with standard derivations) on the *real-sim* and *rcv1* data sets, with different numbers of labeled examples. Results for which the performance of WELLSVM is **significantly** better than SVM are in bold.

Our next experiment studies how the performance of WELLSVM changes with different numbers of labeled examples. Following the setup in Section 5.1.1, 75% of the examples are used for training while the rest are for testing. Different numbers (namely, 25,50,100,150, and 200) of labeled examples are randomly chosen. Since SVMlin cannot handle such a large training set, the SVM is used instead. The above process is repeated 30 times. Table 5 shows the average testing accuracy. As can be seen, WELLSVM is significantly better than SVM in all cases. The high standard deviation of WELLSVM on *real-sim* with 25 labeled examples may be due to the fact that the large amount of unlabeled instances lead to a large variance in deriving a large margin classifier, whereas the amount of labeled examples is too small to reduce the variance.

### 5.1.3 COMPARISON WITH OTHER BENCHMARKS IN THE LITERATURE

In this section, we further evaluate the proposed WELLSVM with other published results in the literature. First, we experiment on the benchmark data sets in Chapelle et al. (2006b) by using their same setup. Results on the average test error are shown in Table 6. As can be seen, WELLSVM is highly competitive.

| | g241c | g241d | Digit1 | USPS | COIL | BCI | Text |
|---|---|---|---|---|---|---|---|
| SVM | 47.32 | 46.66 | 30.60 | **20.03** | 68.36 | 49.85 | 45.37 |
| TSVM | **24.71** | 50.08 | 17.77 | 25.20 | **67.50** | 49.15 | 40.37 |
| WELLSVM | 37.37 | **43.33** | **16.94** | 22.74 | 70.73 | **48.50** | **33.70** |

Table 6: Test errors (%) on the SSL benchmark data sets (using 10 labeled examples). The SVM and TSVM results are from Table 21.9 in Chapelle et al. (2006b).

Next, we compare WELLSVM with the SVM and other state-of-the-art $S^3$VMs reported in Chapelle et al. (2008). These include

1. $\nabla S^3$VM (Chapelle and Zien, 2005), which minimizes the $S^3$VM objective by gradient descent;
2. Continuation $S^3$VM ($cS^3$VM) (Chapelle et al., 2006a), which first relaxes the $S^3$VM objective to a continuous function and then employs gradient descent;
3. USVM (Collobert et al., 2006);
4. TSVM (Joachims, 1999);

| | SVM | ∇S$^3$VM | cS$^3$VM | USVM | TSVM | ∇DA | Newton | BB | WELLSVM |
|---|---|---|---|---|---|---|---|---|---|
| 2moons | 35.6 | 65.0 | 49.8 | 66.3 | 68.7 | 30.0 | 33.5 | **0.0** | 33.5 |
| g50c | 8.2 | 8.3 | 8.3 | 8.5 | 8.4 | **6.7** | 7.5 | - | 7.6 |
| text | 14.8 | **5.7** | 5.8 | 8.5 | 8.1 | 6.5 | 14.5 | - | 8.7 |
| uspst | 20.7 | 14.1 | 15.6 | 14.9 | 14.5 | **11.0** | 19.2 | - | 14.3 |
| coil20 | 32.7 | 23.9 | 23.6 | 23.6 | 21.8 | **18.9** | 24.6 | - | 23.0 |

Table 7: Test errors (%) of the WELLSVM and various S$^3$VM variants. Results of the S$^3$VMs compared are from Table 11 in Chapelle et al. (2008). BB can only be run on the *2moons* data set due to its high computational cost. Note that in Chapelle et al. (2008), USVM is called CCCP and TSVM is called S$^3$VM$^{light}$.

5. Deterministic annealing S$^3$VM with gradient minimization (∇DA) (Sindhwani et al., 2006), which is based on the global optimization heuristic of deterministic annealing;
6. Newton S$^3$VM (Newton) (Chapelle, 2007), which uses the second-order Newton's method; and
7. Branch-and-bound (BB) (Chapelle et al., 2007).

Results are shown in Table 7. As can be seen, BB attains the best performance. Overall, WELLSVM performs slightly worse than ∇DA, but is highly competitive compared with the other S$^3$VM variants.

Finally, we compare WELLSVM with MMC (Xu et al., 2005), a SDP-based S$^3$VM, on the data sets used there. Table 8 shows the results. Again, WELLSVM is highly competitive.

| | HWD 1-7 | HWD 2-3 | Australian | Flare | Vote | Diabetes |
|---|---|---|---|---|---|---|
| MMC | 3.2 | **4.7** | **32.0** | 34.0 | 14.0 | **35.6** |
| WELLSVM | **2.7** | 5.3 | 40.0 | **28.9** | **11.6** | 41.3 |

Table 8: Test errors (%) of WELLSVM and MMC (a SDP-based S$^3$VM) on the data sets used in Xu et al. (2005). The MMC results are copied from their Table 2.

### 5.2 Multi-Instance Learning for Locating ROIs

In this section, we evaluate the proposed method on multi-instance learning, with application to ROI-location in CBIR image data. We employ the image database in Zhou et al. (2005), which consists of 500 COREL images from five image categories: *castle*, *firework*, *mountain*, *sunset* and *waterfall*. Each image is of size $160 \times 160$, and is converted to the multi-instance feature representation by the bag generator SBN (Maron and Ratan, 1998). Each region (instance) in the image (bag) is of size $20 \times 20$. Some of these regions are labeled manually as ROIs. A summary of the data set is shown in Table 9. It is very labor-expensive to collect large image data with all the regions labeled. Hence, we will leave the experiments on large-scale data sets as a future direction.

The one-vs-rest strategy is used. Specifically, a training set of 50 images is created by randomly sampling 10 images from each of the five categories. The remaining 450 images constitute the test set. This training/test split is randomly generated 10 times, and the average performance is reported.

| concept | #images | average #ROIs per image |
|---|---|---|
| *castle* | 100 | 19.39 |
| *firework* | 100 | 27.23 |
| *mountain* | 100 | 24.93 |
| *sunset* | 100 | 2.32 |
| *waterfall* | 100 | 13.89 |

Table 9: Some statistics of the image data set.

Although many multi-instance methods have been proposed, they mainly focus on improving the classification performance, whereas only some of them are used to identify the ROIs. We list these state-of-the-art methods (Andrews et al., 2003; Maron and Ratan, 1998; Zhang and Goldman, 2002; Zhou et al., 2005) as well as related SVM-based methods for comparisons in experiments. Specifically, the WELLSVM is compared with the following SVM variants: 1) MI-SVM (Andrews et al., 2003); 2) mi-SVM (Andrews et al., 2003); and 3) SVM with multi-instance kernel (MI-Kernel) (Gärtner et al., 2002). The Gaussian kernel is used for all the SVMs, where its width $\sigma$ is picked from $\{0.25\sqrt{\gamma}, 0.5\sqrt{\gamma}, \sqrt{\gamma}, 2\sqrt{\gamma}, 4\sqrt{\gamma}\}$ with $\gamma$ being the average distance between instances; $C_1$ is picked from $\{C_2, 4C_2, 10C_2\}$; and $C_2$ is from $\{1, 10, 100\}$. We also compare with three state-of-art non-SVM-based methods that can locate ROIs, namely, Diverse Density (DD) (Maron and Ratan, 1998), EM-DD (Zhang and Goldman, 2002) and C*k*NN-ROI (Zhou et al., 2005). All the parameters are selected by ten-fold cross-validation (except for C*k*NN-ROI, in which its parameters are based on the best setting reported in Zhou et al. (2005)).

In each image classified as relevant by the algorithm, the image region with the maximum prediction value is taken as its ROI.[10] The following two measures are used in evaluating the performance of ROI location.

1. $$\text{success rate of relevant images} = \frac{\text{number of ROI successes}}{\text{number of relevant images}}. \tag{33}$$

   Here, for each image predicted as relevant by the algorithm, the ROI returned by the algorithm is counted as a success if it is a real ROI.

2. The ROI success rate computed based on those images that are predicted as relevant, that is,

$$\text{success rate of ROIs} = \frac{\text{number of ROI successes}}{\text{number of images predicted as relevant}}. \tag{34}$$

Notice that there is a tradeoff between these two measures. When an algorithm classifies many images as relevant, the success rate of relevant images (Equation (33)) is high while the success rate of ROIs (Equation (34)) can be low, since there are many relevant images predicted by the algorithm. On the other hand, when an algorithm classifies many images as irrelevant, the success rate of ROIs is high while the success rate of relevant images is low since many relevant images are missing. To compromise these two goals, we introduce a novel *success rate* of ROIs

$$\textit{success rate} = \frac{2\#\text{ROI successes}}{\#\text{relevant images} + \#\text{predicted relevant images}}.$$

10. Alternatively, if we allow an algorithm to output multiple ROI's for an image, a heuristic thresholding of the prediction values will be needed. For simplicity, we defer such a setup as future work.

| | method | *castle* | *firework* | *mountain* | *sunset* | *waterfall* |
|---|---|---|---|---|---|---|
| SVM methods | WELLSVM | $0.57 \pm 0.12$ | $\mathbf{0.68 \pm 0.17}$ | $\mathbf{0.59 \pm 0.10}$ | $0.32 \pm 0.07$ | $\mathbf{0.39 \pm 0.13}$ |
| | mi-SVM | $0.51 \pm 0.04$ | $0.56 \pm 0.07$ | $0.18 \pm 0.09$ | $0.32 \pm 0.01$ | $0.37 \pm 0.08$ |
| | MI-SVM | $0.52 \pm 0.22$ | $0.63 \pm 0.26$ | $0.18 \pm 0.13$ | $0.29 \pm 0.10$ | $0.06 \pm 0.02$ |
| | MI-Kernel | $0.56 \pm 0.08$ | $0.57 \pm 0.11$ | $0.23 \pm 0.20$ | $0.24 \pm 0.03$ | $0.20 \pm 0.11$ |
| non-SVM methods | DD | $0.24 \pm 0.16$ | $0.15 \pm 0.28$ | $0.56 \pm 0.11$ | $0.30 \pm 0.18$ | $0.26 \pm 0.24$ |
| | EM-DD | $\mathbf{0.69 \pm 0.06}$ | $0.65 \pm 0.24$ | $0.54 \pm 0.18$ | $\mathbf{0.36 \pm 0.15}$ | $0.30 \pm 0.12$ |
| | C*k*NN-ROI | $0.48 \pm 0.05$ | $0.65 \pm 0.09$ | $0.47 \pm 0.06$ | $0.31 \pm 0.04$ | $0.20 \pm 0.05$ |

Table 10: Success rate in locating the ROIs. The best performance and those which are comparable to the best performance (paired $t$-test at 95% significance level) on each data set are bolded.

This is similar to the F-score in information retrieval as

$$\begin{aligned}\frac{1}{\textit{success rate}} &= \frac{\#\text{relevant images} + \#\text{predicted relevant images}}{2\#\text{ROI successes}} \\ &= \frac{1}{2}\left(\frac{1}{\frac{\#\text{ROI successes}}{\#\text{relevant images}}} + \frac{1}{\frac{\#\text{ROI successes}}{\#\text{predicted relevant images}}}\right).\end{aligned}$$

Intuitively, when an algorithm correctly recognizes all the relevant images and their ROIs, the success rate will be high.

Table 10 shows the success rates (with standard deviations) of the various methods. As can be seen, WELLSVM achieves the best performance among all the SVM-based methods. As for its performance comparison with the other non-SVM methods, WELLSVM is still always better than DD and C*k*NN-ROI, and is highly comparable to EM-DD. In particular, EM-DD achieves the best performance on *castle* and *sunset*, while WELLSVM achieves the best performance on the remaining three categories (*firework*, *mountain* and *waterfall*). Figure 5 shows some example images with the located ROIs. It can be observed that WELLSVM can correctly identify more ROIs than the other SVM-based methods.

### 5.3 Clustering

In this section, we further evaluate our WELLSVM on clustering problems where all the labels are unknown. As in semi-supervised learning, 16 UCI data sets and 2 large data sets are used for comparison.

#### 5.3.1 SMALL-SCALE EXPERIMENTS

The WELLSVM is compared with the following methods: 1) $k$-means clustering (KM); 2) kernel $k$-means clustering (KKM); 3) normalized cut (NC) (Shi and Malik, 2000); 4) GMMC (Valizadegan and Jin, 2007); 5) IterSVR[11] (Zhang et al., 2007); and 6) CPMMC[12] (Zhao et al., 2008). In the preliminary experiment, we also compared with the original SDP-based approach in Xu et al. (2005).

11. IterSVR can be found at `http://www.cse.ust.hk/~twinsen`.
12. CPMMC can be found at `http://binzhao02.googlepages.com/`.

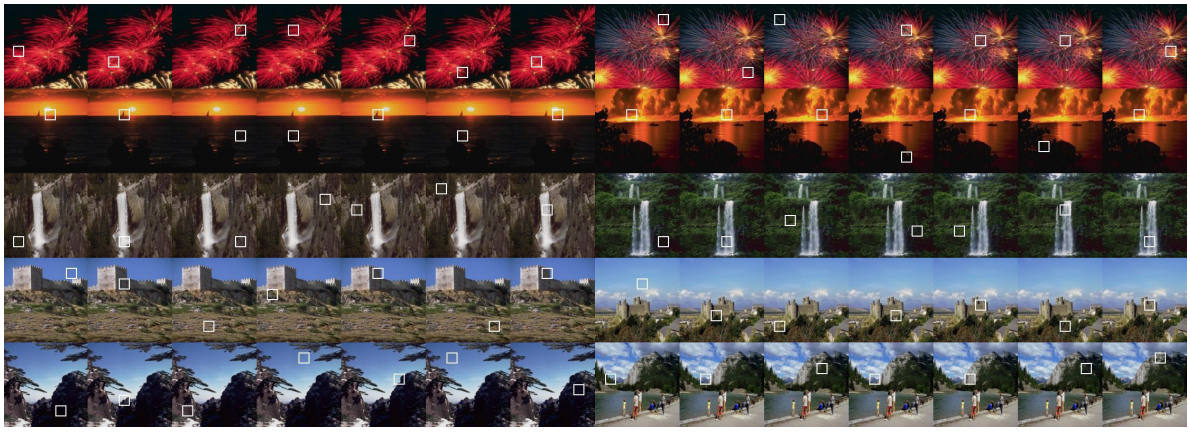


Figure 5: ROIs located by (from left to right) DD, EM-DD, C*k*NN-ROI, MI-SVM, mi-SVM, MI-Kernel, and WELLSVM. Each row shows one category (top to bottom: *firework*, *sunset*, *waterfall*, *castle* and *mountain*).

However, similar to the experimental results in semi-supervised learning, it does not converge after 3 hours on the smallest data set *echocardiogram*. Hence, GMMC, which is also based on SDP but about 100 times faster than Xu et al. (2005), is used in the comparison.

For GMMC, IterSVR, CPMMC and WELLSVM, the $C$ parameter is selected in a range $\{0.1, 0.5, 1, 5, 10, 100\}$. For the UCI data sets, both the linear and Gaussian kernels are used. In particular, the width $\sigma$ of the Gaussian kernel is picked from $\{0.25\sqrt{\gamma}, 0.5\sqrt{\gamma}, \sqrt{\gamma}, 2\sqrt{\gamma}, 4\sqrt{\gamma}\}$, where $\gamma$ is the average distance between instances. The parameter of normalized cut is chosen from the same range of $\sigma$. Since $k$-means and IterSVR are susceptible to the problem of local minimum, these two methods are run 10 times and the average performance reported. We set the balance constraint in the same manner as in Zhang et al. (2007), that is, $\beta$ is set as $0.03N$ for balanced data and $0.3N$ for imbalanced data. To initialize WELLSVM, 20 random label assignments are generated and the one with the maximum kernel alignment (Cristianini et al., 2002) is chosen. We also use this to initialize KM, KKM and IterSVR, and the resultant variants are denoted KM-r, KKM-r and IterSVR-r, respectively. All the methods are reported with the best parameter setting.

We follow the strategy in Xu et al. (2005) to evaluate the clustering accuracy. We first remove the labels for all instances, and then obtain the clusters by the various clustering algorithms. Finally, the misclassification error is measured w.r.t. the true labels.

We first study the clustering accuracy on 16 UCI data sets that cover a wide range of properties. Results are shown in Table 11. As can be seen, WELLSVM outperforms existing clustering approaches on most data sets. Specifically, WELLSVM obtains the best performance on 10 out of 16 data sets. GMMC is not as good as WELLSVM. This may due to that the convex relaxation proposed in GMMC is not the same as the original SDP-based approach (Xu et al., 2005) and WELLSVM.

The CPU time on the UCI data sets are shown in Figure 6. As can be seen, local optimization methods, such as IterSVR and CPMMC, are often efficient. As for the global optimization method, WELLSVM scales much better than GMMC. On average, WELLSVM is about 10 times faster. These results validate that WELLSVM achieves much better scalability than the SDP-based

| Data | KM | KM-r | KKM | KKM-r | NC | GMMC | Iter SVR | Iter SVR-r | CP MMC | WELL SVM |
|---|---|---|---|---|---|---|---|---|---|---|
| *Echocardiogram* | 0.76 | 0.76 | 0.76 | 0.77 | 0.76 | 0.7 | 0.74 | 0.78 | 0.82 | **0.84** |
| *House* | 0.89 | 0.89 | 0.89 | 0.88 | 0.89 | 0.78 | 0.87 | 0.87 | 0.53 | **0.90** |
| *Heart* | 0.66 | 0.59 | 0.69 | 0.59 | 0.57 | **0.7** | 0.59 | 0.59 | 0.56 | 0.59 |
| *Heart-statlog* | 0.68 | 0.79 | 0.78 | 0.79 | 0.79 | 0.77 | 0.76 | 0.76 | 0.56 | **0.81** |
| *Haberman* | 0.6 | 0.59 | 0.69 | 0.64 | 0.7 | 0.6 | 0.62 | 0.57 | **0.74** | **0.74** |
| *LiverDisorders* | 0.55 | 0.54 | 0.56 | 0.56 | 0.57 | 0.55 | 0.53 | 0.51 | **0.58** | **0.58** |
| *Spectf* | 0.58 | 0.57 | **0.77** | **0.77** | 0.63 | 0.64 | 0.53 | 0.53 | 0.73 | 0.73 |
| *Ionosphere* | 0.7 | 0.71 | 0.73 | **0.74** | 0.7 | 0.73 | 0.71 | 0.65 | 0.64 | 0.72 |
| *House-votes* | **0.87** | **0.87** | **0.87** | **0.87** | 0.86 | 0.6 | 0.83 | 0.82 | 0.61 | **0.87** |
| *Clean1* | 0.54 | 0.54 | 0.59 | 0.62 | 0.52 | **0.66** | 0.61 | 0.53 | 0.56 | 0.55 |
| *Isolet* | 0.98 | 0.96 | 0.89 | 0.95 | 0.98 | 0.56 | **1.00** | **1.00** | 0.5 | 0.98 |
| *Australian* | 0.54 | 0.55 | 0.57 | 0.57 | 0.56 | 0.6 | 0.56 | 0.51 | 0.56 | **0.83** |
| *Diabetes* | 0.67 | 0.67 | **0.69** | **0.69** | 0.66 | **0.69** | 0.66 | 0.66 | 0.65 | **0.69** |
| *German* | 0.57 | 0.56 | 0.68 | 0.62 | 0.66 | 0.56 | 0.56 | 0.64 | **0.7** | **0.7** |
| *Krvskp* | 0.52 | 0.51 | 0.55 | 0.55 | **0.56** | - | 0.51 | 0.51 | 0.52 | 0.54 |
| *Sick* | 0.68 | 0.63 | 0.88 | 0.77 | 0.84 | - | 0.63 | 0.59 | **0.94** | **0.94** |

Table 11: Clustering accuracies on various data sets. "-" indicates that the method does not converge in 2 hours or out-of-memory problem occurs.

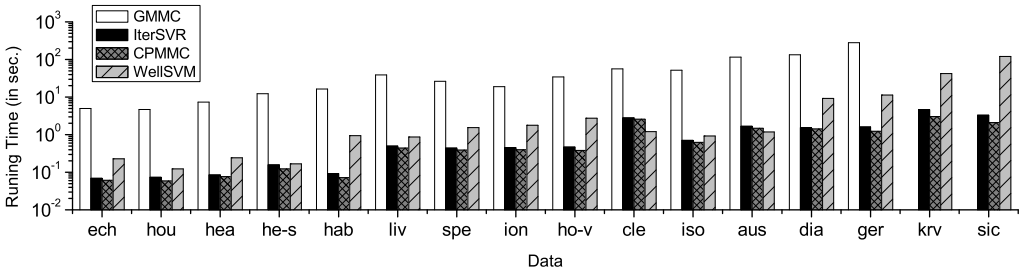



Figure 6: CPU time (in seconds) on the UCI data sets.

GMMC approach. However, in general, convex methods are still slower than non-convex optimization methods on the small data sets.

### 5.3.2 LARGE-SCALE EXPERIMENTS

In this section, we further evaluate the scalability of WELLSVM on large data sets when the linear kernel is used. In this case, the WELLSVM only involves solving a sequence of linear SVMs. As packages specially designed for the linear SVM (such as LIBLINEAR) are much more efficient than those designed for general kernels (such as LIBSVM), it can be expected that the linear WELLSVM is also scalable on large data sets.

The *real-sim* data contains 72,309 instances and has 20,958 features. To study the effect of sample size on performance, different sampling rates (1%, 2%, 5% and $10\%, 20\%, \ldots, 100\%$) are considered. For each sampling rate (except for 100%), we perform random sampling 5 times, and report the average performance. Since $k$-means depends on random initialization, we run it 10 times

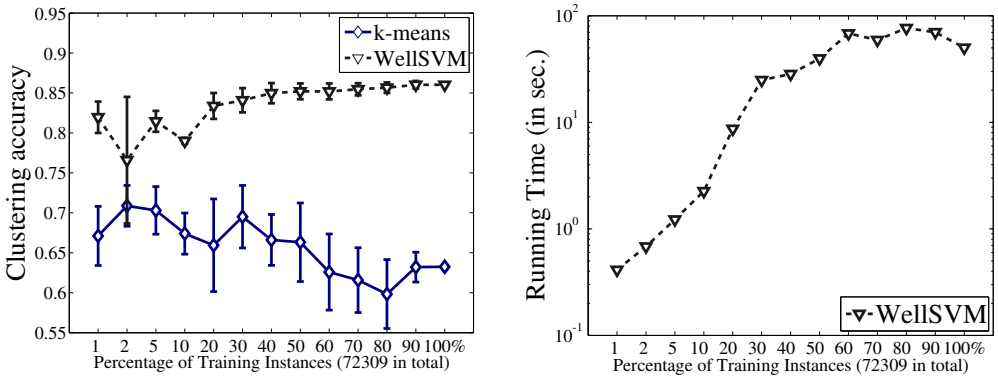



Figure 7: Clustering results on the *real-sim* data with different numbers of examples.

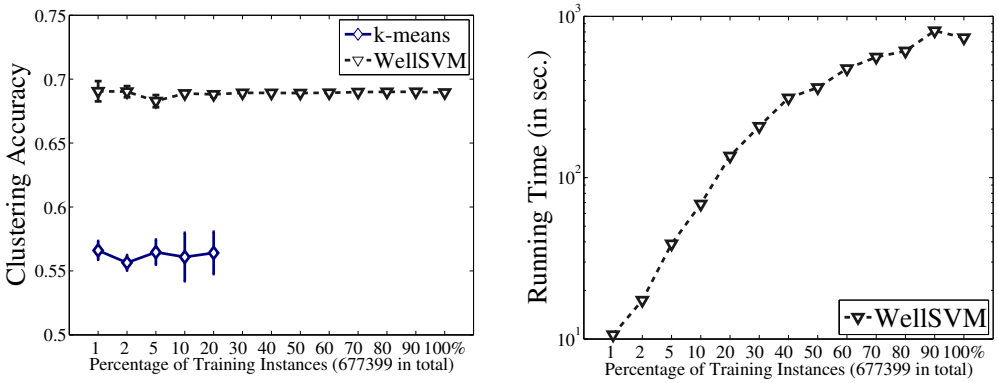



Figure 8: Clustering results on the *RCV1* data with different numbers of examples.

for each sampling rate, and report its average accuracy. Figure 7 shows the accuracy and running time.[13] As can be seen, WELLSVM outperforms *k*-means and can be used on large data sets.

The *RCV1* data is very high-dimensional and contains more than 677,000 instances. Following the same setup as for the *Real-sim* data, WELLSVM is compared with *k*-means under different sampling rates. Figure 8 shows the results. Note that *k*-means does not converge in 24 hours when more than 20% training instances are used. As can be seen, WELLSVM obtains better performance than *k*-means and WELLSVM scales quite well on *RCV1*. It takes fewer than 1,000 seconds for *RCV1* with more than 677,000 instances and 40,000 dimensions.

## 6. Conclusion

Learning from *weakly labeled data*, where the training labels are incomplete, is generally regarded as a crucial yet challenging machine learning task. However, because of the underlying mixed integer programming problem, this limits its scalability and accuracy. To alleviate these difficulties, we proposed a convex WELLSVM based on a novel "label generation" strategy. It can be shown that WELLSVM is at least as tight as existing SDP relaxations, but is much more scalable. Moreover,

13. *k*-means is implemented in matlab, and so its running time is not compared with WELLSVM, whose core procedure is implemented in C++.

since it can be reduced to a sequence of standard SVM training, it can directly benefit from advances in the development of efficient SVM software.

In contrast to traditional approaches that are tailored for a specific weak-label learning problem, our WELLSVM formulation can be used on a general class of weak-label learning problems. Specifically, WELLSVM on three common weak-label learning tasks, namely (i) semi-supervised learning where labels are partially known; (ii) multi-instance learning where labels are implicitly known; and (iii) clustering where labels are totally unknown, can all be put under the same formulation. Experimental results show that the WELLSVM obtains good performance and is readily scalable on large data sets. We believe that similar conclusions can be reached on other weak-label learning tasks, such as the noisy-tolerant problem (Angluin and Laird, 1988).

The focus of this paper is on binary weakly labeled problems. For multi-class weakly labeled problems, they can be easily handled by decomposing into multiple binary problems (Crammer and Singer, 2002). However, one exception is clustering problems, in which existing decomposition methods cannot be applied as there is no label. Extension to this more challenging multi-class clustering scenario will be considered as a future work.

## Acknowledgments

The authors want to thank the editor and reviewers for helpful comments and suggestions. We also thank Teng Zhang, Linli Xu and Kai Zhang for help in the experiments. This work was partially supported by the National Fundamental Research Program of China (2010CB327903), the National Science Foundation of China (61073097, 61021062), the program for outstanding PhD candidate of Nanjing University, Singapore A*star under Grant SERC 112 280 4005, and the Research Grants Council of the Hong Kong Special Administrative Region under Grant 614012. Z.-H. Zhou is the corresponding author of this paper.

## Appendix A. Proof of Theorem 2

**Proof** Let $\{\bar{\alpha}^{(t)}, \bar{\mu}^{(t)}\}$ be the optimal solution of Equation (9), which can be viewed as a saddle-point problem. Let $J(\alpha, \mu) = \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \mu_{\hat{\mathbf{y}}} g_{\hat{\mathbf{y}}}(\alpha)$. Using the saddle-point property (Boyd and Vandenberghe, 2004), we have

$$J(\alpha, \bar{\mu}^{(t)}) \geq J(\bar{\alpha}^{(t)}, \bar{\mu}^{(t)}) \geq J(\bar{\alpha}^{(t)}, \mu), \ \ \forall \alpha, \mu.$$

In other words, $\bar{\alpha}^{(t)}$ minimizes $J(\alpha, \bar{\mu}^{(t)})$. Note that $g_{\mathbf{y}}(\alpha)$ is $\lambda$-strongly convex and $\sum_{\mathbf{y} \in \mathcal{C}^{(t)}} \bar{\mu}_{\mathbf{y}}^{(t)} = 1$, thus $J(\alpha, \bar{\mu}^{(t)})$ is also $\lambda$-strongly convex. Using the Taylor expansion, we have

$$J(\alpha, \bar{\mu}^{(t)}) - J(\bar{\alpha}^{(t)}, \bar{\mu}^{(t)}) \geq \frac{\lambda}{2} \|\alpha - \bar{\alpha}^{(t)}\|^2, \ \ \forall \alpha \in \mathcal{A}.$$

Using the definition of $J(\alpha, \mu)$, we then have

$$\sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\alpha) - \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\bar{\alpha}^{(t)}) \geq \frac{\lambda}{2} \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} \|\alpha - \bar{\alpha}^{(t)}\|^2 = \frac{\lambda}{2} \|\alpha - \bar{\alpha}^{(t)}\|^2. \tag{35}$$

Let $\hat{\mathbf{y}}^{(t+1)}$ be the violated label vector selected at iteration $t+1$ in Algorithm 1, that is, $\mathcal{C}^{t+1} = \mathcal{C}^{(t)} \bigcup \hat{\mathbf{y}}^{(t+1)}$. From the definition, we have

$$\begin{aligned} g_{\hat{\mathbf{y}}^{(t+1)}}(\bar{\alpha}^{(t)}) = -G(\bar{\alpha}^{(t)}, \hat{\mathbf{y}}^{(t+1)}) &\geq \max_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} -G(\bar{\alpha}^{(t)}, \hat{\mathbf{y}}) + \varepsilon = \max_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} g_{\hat{\mathbf{y}}}(\bar{\alpha}^{(t)}) + \varepsilon \\ &\geq \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\bar{\alpha}^{(t)}) + \varepsilon = -p^{(t)} + \varepsilon. \end{aligned} \tag{36}$$

Consider the following optimization problem and let $\hat{p}^{(t+1)}$ be its optimal objective value

$$\hat{p}^{(t+1)} = -\min_{\alpha \in \mathcal{A}} \max_{0 \leq \theta \leq 1} \theta \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\alpha) + (1-\theta) g_{\hat{\mathbf{y}}^{(t+1)}}(\alpha). \tag{37}$$

When $\theta = 1$, it reduces to Equation (9) at iteration $t$, and so $\hat{p}^{(t+1)} \leq p^{(t)}$. On the other hand, note that $\theta \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} + (1-\theta) = \theta + (1-\theta) = 1$, the optimal solution in Equation (37) is suboptimal to that of Equation (9) at iteration $t+1$. Then we have $p^{(t+1)} \leq \hat{p}^{(t+1)}$. Let $\hat{p}^{(t+1)} = p^{(t)} - \eta$, now we aims at showing $\eta \geq (\frac{-c+\sqrt{c^2+4\varepsilon}}{2})^2$ which obviously induces our final inequality Equation (10).

Let $\{\tilde{\alpha}^{(t)}, \tilde{\theta}\}$ be the optimal solution of Equation (37), we have following inequalities

$$p^{(t)} - \eta \leq -\sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\tilde{\alpha}^{(t)}), \tag{38}$$

$$p^{(t)} - \eta \leq -g_{\hat{\mathbf{y}}^{(t+1)}}(\tilde{\alpha}^{(t)}). \tag{39}$$

Using Equations (35), (36), (38) and (39), we have

$$\eta \geq \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\tilde{\alpha}^{(t)}) - \sum_{\hat{\mathbf{y}} \in \mathcal{C}^{(t)}} \bar{\mu}_{\hat{\mathbf{y}}}^{(t)} g_{\hat{\mathbf{y}}}(\bar{\alpha}^{(t)}) \geq \frac{\lambda}{2} \|\tilde{\alpha}^{(t)} - \bar{\alpha}^{(t)}\|^2, \tag{40}$$

$$\varepsilon - \eta \leq g_{\hat{\mathbf{y}}^{(t+1)}}(\bar{\alpha}^{(t)}) - g_{\hat{\mathbf{y}}^{(t+1)}}(\tilde{\alpha}^{(t)}) \leq M \|\tilde{\alpha}^{(t)} - \bar{\alpha}^{(t)}\|. \tag{41}$$

On combining Equations (40) and (41), we obtain

$$\varepsilon - \eta \leq M \sqrt{\frac{2\eta}{\lambda}},$$

and then finally we have $\eta \geq \left(\frac{-c+\sqrt{c^2+4\varepsilon}}{2}\right)^2$, where $c = \frac{M}{\sqrt{\lambda/2}}$. ■

## References

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 577–584. MIT Press, Cambridge, MA, 2003.

D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the 21st International Conference on Machine Learning*, pages 41–48, Banff, Canada, 2004.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

S. S. Bucak, R. Jin, and A. K. Jain. Multi-label learning with incomplete class assignments. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 2801–2808, Colorado Springs, CO, 2011.

O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5):1155–1178, 2007.

O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, The Savannah Hotel, Barbados, 2005.

O. Chapelle, M. Chi, and A. Zien. A continuation method for semi-supervised SVMs. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 185–192, Pittsburgh, PA, 2006a.

O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, USA, 2006b.

O. Chapelle, V. Sindhwani, and S. S. Keerthi. Branch and bound for semi-supervised support vector machines. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 217–224. MIT Press, Cambridge, MA, 2007.

O. Chapelle, V. Sindhwani, and S. S. Keerthi. Optimization techniques for semi-supervised support vector machines. *Journal of Machine Learning Research*, 9:203–233, 2008.

P. M. Cheung and J. T. Kwok. A regularization framework for multiple-instance learning. In *Proceedings of the 23th International Conference on Machine Learning*, pages 193–200, Pittsburgh, PA, USA, 2006.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Large scale transductive SVMs. *Journal of Machine Learning Research*, 7:1687–1712, 2006.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2002.

N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In T. G. Dietterich, Z. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 367–373. MIT Press, Cambridge, MA, 2002.

T. De Bie and N. Cristianini. Semi-supervised learning using semi-definite programming. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

R.E. Fan, P.H. Chen, and C.J. Lin. Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 6:1889–1918, 2005.

T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *Proceedings of the 19th International Conference on Machine Learning*, pages 179–186, Sydney, Australia, 2002.

Y. Guo. Max-margin multiple-instance learning via semidefinite programming. In *Proceedings of the 1st Asian Conference on Machine Learning*, pages 98–108, Nanjing, China, 2009.

R. Horst and N.V. Thoai. DC programming: Overview. *Journal of Optimization Theory and Applications*, 103(1):1–43, 1999.

C. J. Hsieh, K. W. Chang, C. J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the 25th International Conference on Machine Learning*, pages 408–415, Helsinki, Finland, 2008.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Upper Saddle River, NJ, USA, 1988.

T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning*, pages 200–209, Bled, Slovenia, 1999.

T. Joachims. Training linear SVMs in linear time. In *Proceedings of the 12th International Conference on Knowledge Discovery and Data mining*, pages 217–226, Philadelphia, PA, 2006.

J. E. Kelley. The cutting plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

S.-J. Kim and S. Boyd. A minimax theorem with applications to machine learning, signal processing, and finance. *SIAM Journal on Optimization*, 19(3):1344–1367, 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, Cambridge, MA, 2009.

G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

Y.-F. Li and Z.-H. Zhou. Towards making unlabeled data never hurt. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1081–1088, Bellevue, WA, 2011.

Y.-F. Li, J.T. Kwok, I.W. Tsang, and Z.-H. Zhou. A convex method for locating regions of interest with multi-instance learning. In *Proceedings of the 20th European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 15–30, Bled, Slovenia, 2009a.

Y.-F. Li, J.T. Kwok, and Z.-H. Zhou. Semi-supervised learning using label mean. In *Proceedings of the 26th International Conference on Machine Learning*, pages 633–640, Montreal, Canada, 2009b.

Y.-F. Li, I.W. Tsang, J.T. Kwok, and Z.-H. Zhou. Tighter and convex maximum margin clustering. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 344–351, Clearwater Beach, FL, 2009c.

Y.-F. Li, J.-H. Hu, Y. Jiang, and Z.-H. Zhou. Towards discovering what patterns trigger what labels. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 1012–1018, Toronto, Canada, 2012.

M.S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear algebra and its applications*, 284(1):193–228, 1998.

O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *Proceedings of the 15th International Conference on Machine Learning*, pages 341–349, Madison, WI, 1998.

T.M. Mitchell. The discipline of machine learning. Technical report, Machine Learning Department, Carnegie Mellon University, 2006.

Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*, volume 13. Society for Industrial Mathematics, 1987.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.

A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

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

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, Corvallis, OR, 2007.

V.S. Sheng, F. Provost, and P.G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th International Conference on Knowledge Discovery and Data mining*, pages 614–622, Las Vegas, NV, 2008.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

V. Sindhwani and S.S. Keerthi. Large scale semi-supervised linear SVMs. In *Proceedings of the 29th annual International Conference on Research and Development in Information Retrieval*, pages 477–484, Seattle, WA, 2006.

V. Sindhwani, S.S. Keerthi, and O. Chapelle. Deterministic annealing for semi-supervised kernel machines. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 841–848, Pittsburgh, PA, 2006.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, 2006.

A. Subramanya and J. Bilmes. Entropic graph regularization in non-parametric semi-supervised classification. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1803–1811. MIT Press, Cambridge, MA, 2009.

Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou. Multi-label learning with weak label. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 593–598, Atlanta, GA, 2010.

I. W. Tsang, J. T. Kwok, and P. Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2006.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(2):1453, 2006.

H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1417–1424. MIT Press, Cambridge, MA, 2007.

V.N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

H. Y. Wang, Q. Yang, and H. Zha. Adaptive p-posterior mixture-model kernels for multiple instance learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1136–1143, Helsinki, Finland, 2008.

L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 904–910, Pittsburgh, PA, 2005.

L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1537–1544. MIT Press, Cambridge, MA, 2005.

X. Xu and E. Frank. Logistic regression and boosting for labeled bags of instances. In *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 272–281, Sydney, Australia, 2004.

Z. Xu, R. Jin, I. King, and M. R. Lyu. An extended level method for efficient multiple kernel learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1825–1832. MIT Press, Cambridge, MA, 2009.

Z. Xu, R. Jin, H. Yang, I. King, and M. Lyu. Simple and efficient multiple kernel learning by group lasso. In *Proceedings of 27th International Conference on Machine Learning*, pages 1–8, Haifa, Israel, 2010.

S.-J. Yang, Y. Jiang, and Z.-H. Zhou. Multi-instance multi-label learning with weak label. In *Proceedings of 23rd International Joint Conference on Artificial Intelligence*, Beijing, China, 2013.

K. Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1119–1126, Corvallis, OR, 2007.

K. Zhang, J.T. Kwok, and B. Parvin. Prototype vector machine for large scale semi-supervised learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1233–1240, Montreal, Canada, 2009a.

K. Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. *IEEE Transactions on Neural Networks*, 20(4):583–596, 2009b.

Q. Zhang and S. A. Goldman. EM-DD: An improved multiple-instance learning technique. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 1073–1080. MIT Press, Cambridge, MA, 2002.

B. Zhao, F. Wang, and C. Zhang. Efficient maximum margin clustering via cutting plane algorithm. In *Proceedings of the 8th International Conference on Data Mining*, pages 751–762, Atlanta, GA, 2008.

Z.-H. Zhou and M. Li. Semi-supervised learning by disagreement. *Knowledge and Information Systems*, 24(3):415–439, 2010.

Z.-H. Zhou and J.-M. Xu. On the relation between multi-instance learning and semi-supervised learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1167–1174, Corvallis, OR, 2007.

Z.-H. Zhou and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In B. Schölkopf, J. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems 19*, pages 1609–1616. MIT Press, Cambridge, MA, 2007.

Z.-H. Zhou, X.-B. Xue, and Y. Jiang. Locating regions of interest in CBIR with multi-instance learning techniques. In *Proceedings of the 18th Australian Joint Conference on Artificial Intelligence*, pages 92–101, Sydney, Australia, 2005.

Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.

X. Zhu. Semi-supervised learning literature survey. Technical report, Computer Science, University of Wisconsin-Madison, 2006.

Journal of Machine Learning Research 14 (2013) 2189-2212 Submitted 12/12; Published 7/13

# Language-Motivated Approaches to Action Recognition

**Manavender R. Malgireddy** MRM42@BUFFALO.EDU
**Ifeoma Nwogu** INWOGU@BUFFALO.EDU
**Venu Govindaraju** GOVIND@BUFFALO.EDU
*Department of Computer Science and Engineering*
*University at Buffalo, SUNY*
*Buffalo, NY 14260, USA*

**Editors:** Isabelle Guyon and Vassilis Athitsos

## Abstract

We present language-motivated approaches to detecting, localizing and classifying activities and gestures in videos. In order to obtain statistical insight into the underlying patterns of motions in activities, we develop a dynamic, hierarchical Bayesian model which connects low-level visual features in videos with poses, motion patterns and classes of activities. This process is somewhat analogous to the method of detecting topics or categories from documents based on the word content of the documents, except that our documents are dynamic. The proposed generative model harnesses both the temporal ordering power of dynamic Bayesian networks such as hidden Markov models (HMMs) and the automatic clustering power of hierarchical Bayesian models such as the latent Dirichlet allocation (LDA) model. We also introduce a probabilistic framework for detecting and localizing pre-specified activities (or gestures) in a video sequence, analogous to the use of filler models for keyword detection in speech processing. We demonstrate the robustness of our classification model and our spotting framework by recognizing activities in unconstrained real-life video sequences and by spotting gestures via a one-shot-learning approach.

**Keywords:** dynamic hierarchical Bayesian networks, topic models, activity recognition, gesture spotting, generative models

## 1. Introduction

Vision-based activity recognition is currently a very active area of computer vision research, where the goal is to automatically recognize different activities from a video. In a simple case where a video contains only one activity, the goal is to classify that activity, whereas, in a more general case, the objective is to detect the start and end locations of different specific activities occurring in a video. The former, simpler case is known as *activity classification* and latter as *activity spotting*. The ability to recognize activities in videos, can be helpful in several applications, such as monitoring elderly persons; surveillance systems in airports and other important public areas to detect abnormal and suspicious activities; and content based video retrieval, amongst other uses.

There are several challenges in recognizing human activities from videos and these include videos taken with moving background such as trees and other objects; different lighting conditions (day time, indoor, outdoor, night time); different view points; occlusions; variations within each activity (different persons will have their own style of performing an activity); large number of activities; and limited quantities of labeled data amongst others.

©2013 Manavender R. Malgireddy, Ifeoma Nwogu and Venu Govindaraju.

Recent advances in applied machine learning, especially in natural language and text processing, have led to a new modeling paradigm where high-level problems can be modeled using combinations of lower-level segmental units. Such units can be learned from large data sets and represent the universal set of alphabets to fully describe a vocabulary. For example, in a high-level problem such as speech recognition, a phoneme is defined as the smallest segmental unit employed to form an utterance (speech vector). Similarly, in language based documents processing, words in the document often represent the smallest segmental unit while in image-based object identification, the bag-of-words (or bag-of-features) technique learns the set of small units required to segment and label the object parts in the image. These features can then be input to generative models based on hierarchical clustering paradigms, such as topic modeling methods, to represent different levels of abstractions.

Motivated by the successes of this modeling technique in solving general high-level problems, we define an activity as a sequence of contiguous sub-actions, where the sub-action is a discrete unit that can be identified in a action stream. For example, in a natural setting, when a person waves goodbye, the sub-actions involved could be (i) raising a hand from rest position to a vertical upright position; (ii) moving the arm from right to left; and (iii) moving the arm from left to right. The entire activity or gesture[1] therefore consists of the first sub-action occurring once and the second and third sub-actions occurring multiple times. Extracting the complete vocabulary of sub-actions in activities is a challenging problem since the exhaustive list of sub-actions involved in a set of given activities is not necessarily known beforehand. We therefore propose machine learning models and algorithms to (i) compose a compact, near-complete vocabulary of sub-actions in a given set of activities; (ii) recognize the specific actions given a set of known activities; and (iii) efficiently learn a generative model to be used in recognizing or spotting a pre-specified action, given a set of activities.

We therefore hypothesize that the use of sub-actions in combination with the use of a generative model for representing activities will improve recognition accuracy and can also aid in activity spotting. We will perform experiments using various available publicly available benchmark data sets to evaluate our hypothesis.

## 2. Background and Related Work

Although extensive research has gone into the study of the classification of human activities in video, fewer attempts have been made to spot actions from an activity stream. A recent, more complete survey on activity recognition research is presented by Aggarwal and Ryoo (2011). We divide the related work in activity recognition into two main categories: activity classification and activity spotting.

### 2.1 Activity Classification

Approaches for activity classification can be grouped into three categories: (i) space-time approaches: a video is represented as a collection of space-time feature points and algorithms are designed to learn a model for each activity using these features; (ii) sequential approaches: features are extracted from video frames sequentially and a state-space model such as a hidden Markov

1. When referring to activity spotting purposes, we use the term gestures instead of activities, only to be consistent with the terminology of the *ChaLearn Gesture Challenge*.

model (HMM) is learned over the features; (iii) hierarchical approaches: an activity is modeled hierarchically, as combination of simpler low level activities. We will briefly describe each of these approaches along with the relevant literature, in sections below.

### 2.1.1 SPACE-TIME APPROACHES

Space-time approaches represent a video as a collection of feature points and use these points for classification. A typical space-time approach for activity recognition involves the detection of interest points and the computation of various descriptors for each interest point. The collection of these descriptors (bag-of-words) is therefore the representation of a video. The descriptors of labeled training data are presented to a classifier during training. Hence, when an unlabeled, unseen video is presented, similar descriptors are extracted as mentioned above and presented to a classifier for labeling. Commonly used classifiers in the space-time approach to activity classification include support vector machines (SVM), K-nearest neighbor (KNN), etc.

Spatio-temporal interest points were initially introduced by Laptev and Lindeberg (2003) and since then, other interest-point-based detectors such as those based on spatio-temporal Hessian matrix (Willems et al., 2008) and Gabor filters (Bregonzio et al., 2009; Dollár et al., 2005) have been proposed. Various other descriptors such as those based on histogram-of-gradients (HoG) (Dalal and Triggs, 2005) or histogram-of-flow (HoF) (Laptev et al., 2008), three-dimensional histogram-of-gradients (HoG3D) (Kläser et al., 2008), three-dimensional scale-invariant feature transform (3D-SIFT) (Scovanner et al., 2007) and local trinary patterns (Yeffet and Wolf, 2009), have also been proposed to describe interest points. More recently, descriptors based on tracking interest points have been explored (Messing et al., 2009; Matikainen et al., 2009). These use standard Kanade-Lucas-Tomasi (KLT) feature trackers to track interest points over time.

In a recent paper by Wang et al. (2009), the authors performed an evaluation of local spatio-temporal features for action recognition and showed that dense sampling of feature points significantly improved classification results when compared to sparse interest points. Similar results were also shown for image classification (Nowak et al., 2006).

### 2.1.2 SEQUENTIAL APPROACHES

Sequential approaches represent an activity as an ordered sequence of features, here the goal is to learn the order of specific activity using state-space models. HMMs and other dynamic Bayesian networks (DBNs) are popular state-space models used in activity recognition. If an activity is represented as a set of hidden states, each hidden state can produce a feature at each time frame, known as the observation. HMMs were first applied to activity recognition in 1992 by Yamato et al. (1992). They extracted features at each frame of a video by first binarizing the frame and dividing it into $(M \times N)$ meshes. The feature for each mesh was defined as the ratio of black pixels to the total number of pixels in the mesh and all the mesh features were concatenated to form a feature vector for the frame. An HMM was then learned for each activity using the standard Expectation-Maximization (EM) algorithm. The system was able to detect various tennis strokes such as forehand stroke, smash, and serve from one camera viewpoint. The major drawback of the conventional HMM was its inability to handle activities with multiple persons. A variant of HMM called coupled HMM (CHMM) was introduced by Oliver et al. (2000), which overcame this drawback by coupling HMMs, where each HMM in the CHMM modeled one person's activity. In their experiments they coupled two HMMs to model human-human interactions, but again this was

somewhat limited in its applications. An approach to extend both HMM and CHMMs by explicitly modeling the duration of an activity using states was also proposed by Natarajan and Nevatia (2007). Each state in a coupled hidden semi-Markov model (CHSMMs) had its own duration and the sequence of these states defined the activity. Their experiments showed that CHSMM modeled an activity better than the CHMM.

#### 2.1.3 HIERARCHICAL APPROACHES

The main idea of hierarchical approaches is to perform recognition of higher-level activities by modeling them as a combination of other simpler activities. The major advantage of these approaches over sequential approaches is their ability to recognize activities with complex structures. In hierarchical approaches, multiple layers of state-based models such as HMMs and other DBNs are used to recognize higher level activities. In most cases, there are usually two layers. The bottom layer takes features as inputs and learns atomic actions called *sub-actions*. The results from this layer are fed into the second layer and used for the actual activity recognition. A layered hidden Markov model (LHMM) (Oliver et al., 2002) was used in an application for office awareness. The lower layer HMMs classified the video and audio data with a time granularity of less than 1 second while the higher layer learned typical office activities such as phone conversation, face-to-face conversation, presentation, etc. Each layer of the HMM was designed and trained separately with fully labeled data. Hierarchical HMMs (Nguyen et al., 2005) were used to recognize human activities such as person having "short-meal", "snacks" and "normal meal". They also used a 2-layer architecture where lower layer HMM modeled simpler behaviors such as moving from one location in a room to another and the higher layer HMM used the information from layer one as its features. The higher layer was then used to recognize activities. A method based on modeling temporal relationships among a set of different temporal events (Gong and Xiang, 2003) was developed and used for a scene-level interpretation to recognize cargo loading and unloading events.

The main difference between the above mentioned methods and our proposed method, is that these approaches assume that the higher-level activities and atomic activities (sub-actions) are known *a priori*, hence, the parameters of the model can be learned directly based on this notion. While this approach might be suitable for a small number of activities, it does not hold true for real-word scenarios where there is often a large number of sub-actions along with many activities (such as is found in the HMDB data set which is described in more detail in Section 6.2). *For activity classification, we propose to first compute sub-actions by clustering dynamic features obtained from videos, and then learn a hierarchical generative model over these features, thus probabilistically learning the relations between sub-actions, that are necessary to recognize different activities including those in real-world scenarios.*

### 2.2 Activity Spotting

Only a few methods have been proposed for activity spotting. Among them is the work of Yuan et al. (2009), which represented a video as a 3D volume and activities-of-interest as sub-volumes. The task of activity spotting was therefore reduced to one of performing an optimal search for activities in the video. Another work in spotting by Derpanis et al. (2010) introduced a local descriptor of video dynamics based on visual spacetime oriented energy measures. Similar to the previous work, their input was also a video which was searched for a specific action. The limitation of these techniques is their inability to adapt to changes in view points, scale, appearance etc. Rather than

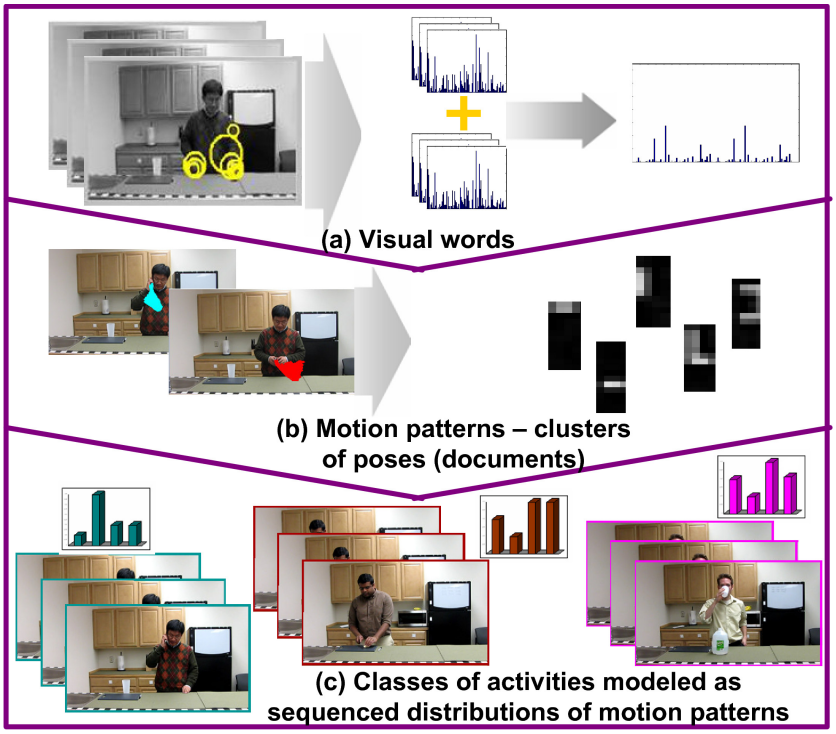



Figure 1: Our general framework abstracts low-level visual features from videos and connects them to poses, motion patterns and classes of activity. (a) A video sequence is divided into short segments with a few frames only. In each segment, the space time interest points are computed. At the interest points, HoG and HoF are computed, concatenated and quantized to represent our low-level *visual words*. We discover and model a distribution over visual words which we refer to as *poses* (not shown in image). (b) Atomic motions are discovered and modeled as distributions over poses. We refer to these atomic motions as *motion patterns* or *sub-actions*. (c) Each video segment is modeled as a distribution over motion patterns. The time component is incorporated by modeling the transitions between the video segments, so that a complete video is modeled as a dynamic network of motion patterns. The distributions and transitions of underlying motion patterns in a video determine the final activity label assigned to that video.

being defined on the motion patterns involved in an activity, these methods performed template matching type techniques, which do not readily generalize to new environments exhibiting a known activity. Both methods reported their results on the KTH and CMU data sets (described in more detail in Section 6), where the environment in which the activities were being performed did not readily change.

## 3. A Language-Motivated Hierarchical Model for Classification

Our proposed language-motivated hierarchical approach aims to perform recognition of higher-level activities by modeling them as a combination of other simpler activities. The major advantage of this

approach over the typical sequential approaches and other hierarchical approaches is its ability to recognize activities with complex structures. By employing a hierarchical approach, multiple layers of state-based dynamic models can be used to recognize higher level activities. The bottom layers take observed features as inputs in order to recognize atomic actions (sub-actions). The results from these lower layers are then fed to the upper layers and used to recognize the modular activity.

### 3.1 Hierarchical Activity Modeling using Multi-class Markov Chain Latent Dirichlet Allocation (MCMCLDA)

We propose a supervised dynamic, hierarchical Bayesian model, the multi-class Markov chain latent Dirichlet allocation (MCMCLDA), which captures the temporal information of an activity by modeling it as sequence of motion patterns, based on the Markov assumption. We develop this generative learning framework in order to obtain statistical insight into the underlying motions patterns (sub-actions) involved in an activity. An important aspect of this model is that motion patterns are shared across activities. So although the model is generative in structure, it can act discriminatively as it specifically learns which motion patterns are present in each activity. The fact that motion patterns are shared across activities was validated empirically (Messing et al., 2009) on the University of Rochester activities data set. Our proposed generative model harnesses both the temporal ordering power of DBNs and the automatic clustering power of hierarchical Bayesian models. The model correlates these motion patterns over time in order to define the signatures for classes of activities. Figure 1 shows an overview of the implementation network although we do not display *poses*, since they have no direct meaningful physical manifestations.

A given video is broken into motion segments comprising of either a combination of a fixed number of frames, or at the finest level, a single frame. Each motion segment can be represented as bag of vectorized descriptors (visual words) so that the input to the model (at time $t$) is the bag of visual words for motion segment $t$. Our model is similar in sprit to Hospedales et al. (2009), where the authors mine behaviors in video data from public scenes using an unsupervised framework. A major difference is that our MCMCLDA is a supervised version of their model in which motion-patterns/behaviors are shared across different classes, which makes it possible to handle a large number of different classes. If we assume that there exists only one class, then the motion-patterns are no longer shared, our model also becomes unsupervised and will thus be reduced to that of Hospedales et al. (2009).

We view MCMCLDA as a generative process and include a notation section before delving into the details of the LDA-type model:

$m$ = any single video in the corpus,
$z_t$ = motion pattern at time $t$ (a video is assumed to be made up of motion patterns),
$y_{t,i}$ = the hidden variable representing a pose at motion pattern $i$, in time $t$ (motion patterns are assumed to be made up of poses),
$x_{t,i}$ = the slices of the input video which we refer to as visual words and are the only observable variables,
$\phi_y$ = the visual word distribution for pose $y$,
$\theta_z$ = motion pattern specific pose distribution,
$c_m$ is the class label for the video $m$; (for one-shot learning, one activity is represented by one video $(N_m = 1)$),
$\psi_j$ = $j^{th}$ class-specific transition matrix for the transition from one motion pattern to the next,

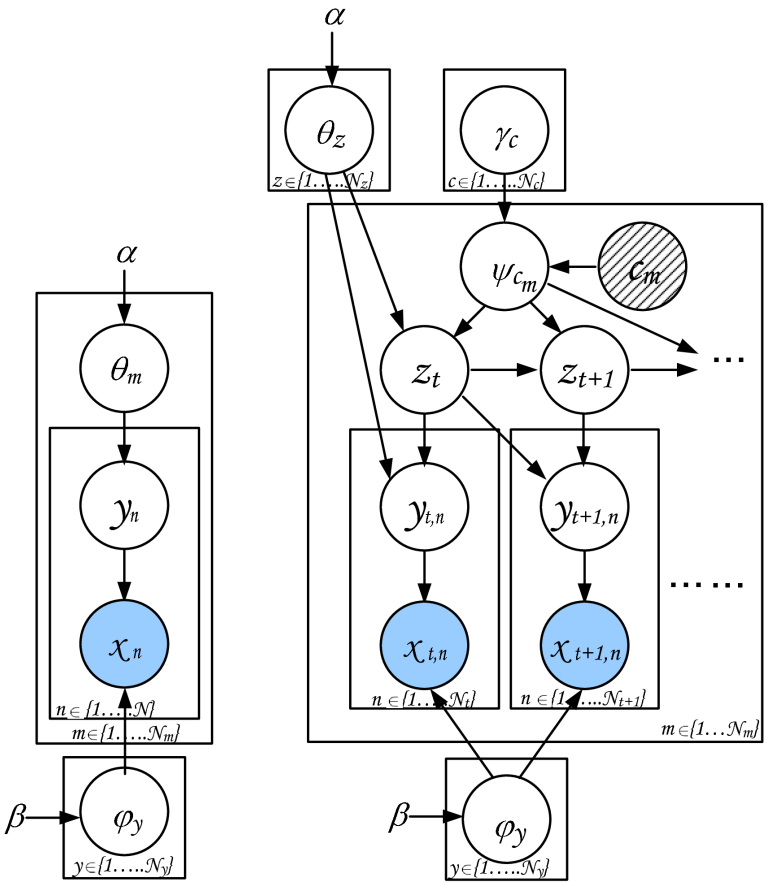



Figure 2: Left: plates diagram for standard LDA; right: plates diagram for a dynamic model which extends LDA to learning the states from sequence data

$\gamma_c$ = the transition matrix distribution for a video,
$\alpha$,$\beta$ = the hyperparameters of the priors.

The complete generative model is given by:

$$\begin{aligned}
\vec{\psi}_j^z &\sim Dir(\psi_j^z|\gamma_j),\\
\vec{\theta}_z &\sim Dir(\theta_z|\alpha),\\
\vec{\phi}_y &\sim Dir(\phi_y|\beta),\\
z_t &\sim Mult(z_t|\vec{\psi}_j^{z_{t-1}}),\\
y_{t,i} &\sim Mult(y_{t,i}|\vec{\theta}_{z_t}),\\
x_{t,i} &\sim Mult(x_{t,i}|\vec{\phi}_{y_{t,i}}),
\end{aligned}$$

where $Mult(\cdot)$ refers to a multinomial distribution.

Now, consider the Bayesian network of MCMCLDA shown in Figure 2. This can be interpreted as follows: For each video $m$ in the corpus, a motion pattern indicator $z_t$ is drawn from $p(z_t|z_{t-1},\vec{\psi}_{c_m})$, denoted by Mult$(\vec{\psi}_{c_m}^{z_{t-1}})$, where $c_m$ is the class label for the video $m$. Then the corresponding pattern specific pose distribution $\vec{\theta}_{z_t}$ is used to draw visual words for that segment. That is, for each visual word, a pose indicator $y_{t,i}$ is sampled according to pattern specific pose distribution $\vec{\theta}_{z_t}$, and then the corresponding pose-specific word distribution $\vec{\phi}_{y_{t,i}}$ is used to draw a visual word. The poses $\vec{\phi}_y$, motion patterns $\vec{\theta}_z$ and transition matrices $\vec{\psi}_j$ are sampled once for the entire corpus.

The joint distribution of all known and hidden variables given the hyperparameters for a video is:

$$p(\{x_t,y_t,z_t\}_1^T,\phi,\psi_j,\theta|\alpha,\beta,\gamma_j) = p(\phi|\beta)p(\theta|\alpha)p(\psi|\gamma_j)\prod_t\prod_i p(x_{t,i}|y_{t,i})p(y_{t,i}|z_t)p(z_t|z_{t-1}).$$

### 3.2 Parameter Estimation and Inference of the MCMCLDA Model

As in the case with LDA, exact inference is intractable. We therefore use collapsed Gibbs sampler for approximate inference and learning. The update equation for pose from which the Gibbs sampler draws the hidden pose $y_{t,i}$ is obtained by integrating out the parameters $\theta,\phi$ and noting that $x_{t,i}=x$ and $z_t=z$ :

$$p(y_{t,i}=y|\mathbf{y}_{\neg(\mathbf{t},\mathbf{i})},\mathbf{z},\mathbf{x}) \propto \frac{n_{x,y}^{\neg(t,i)}+\beta}{\sum_{x=1}^{N_x} n_{x,y}^{\neg(t,i)}+N_x\beta}(n_{y,z}^{\neg(t,i)}+\alpha), \tag{1}$$

where $n_{x,y}^{\neg(t,i)}$ denote the number of times that visual word $x$ is observed with pose $y$ excluding the token at $(t,i)$ and $n_{y,z}^{\neg(t,i)}$ refers to the number of times that pose $y$ is associated with motion pattern $z$ excluding the token at $(t,i)$. $N_x$ is size of codebook and $N_y$ is the number of poses.

The Gibbs sampler update for motion-pattern at time $t$ is derived by taking into account that at time $t$, there can be many different poses associated to a single motion-pattern $z_t$ and also the possible transition from $z_{t-1}$ to $z_{t+1}$. The update equation for $z_t$ can be expressed as :

$$p(z_t=z|\mathbf{y},\mathbf{z}_{\neg\mathbf{t}}) \propto p(y_t|z_t=z,z_{\neg t},y_{\neg t})p(z_t=z|z_{\neg t}^{c_m},c_m). \tag{2}$$

The likelihood term $p(y_t|z_t=z,z_{\neg t},y_{\neg t})$ cannot be reduced to the simplified form as in LDA as the difference between $n_{y,z}^{\neg t}$ and $n_{y,z}$ is not one, since there will be multiple poses associated to the motion-pattern $z_t$. $n_{y,z}$ denotes the number of times pose $y$ is associated with motion-pattern $z$ and $n_{y,z}^{\neg t}$ refers to the number of times pose $y$ is observed with motion-pattern $z$ excluding the poses (multiple) at time $t$. Taking the above condition into account, the likelihood term can be obtained as below:

$$p(y_t|z_t=z,z_{\neg t},y_{\neg t}) = \frac{\prod_y \Gamma(n_{y,z}+\alpha)}{\prod_y \Gamma(n_{y,z}^{\neg t}+\alpha)}\frac{\Gamma(\sum_y n_{y,z}^{\neg t}+N_y\alpha)}{\Gamma(\sum_y n_{y,z}+N_y\alpha)}.$$

Prior term $p(z_t = z|z_{\neg t}^{c_m}, c_m)$ is calculated as below depending on the values of $z_{t-1}, z_t$ and $z_{t+1}$.

$$
\begin{aligned}
&if \quad z_{t-1} \neq z: \\
&= \frac{n^{(c_m)}_{z_{t-1},z,\neg t} + \gamma_{c_m}}{\sum_z n^{(c_m)}_{z_{t-1},z,\neg t} + N_z\gamma_{c_m}} \frac{n^{(c_m)}_{z,z_{t+1},\neg t} + \gamma_{c_m}}{\sum_{z_{t+1}} n^{(c_m)}_{z,z_{t+1},\neg t} + N_z\gamma_{c_m}}, \\
&if \quad z_{t-1} = z = z_{t+1}: \\
&= \frac{n^{(c_m)}_{z_{t-1},z,\neg t} + 1 + \gamma_{c_m}}{\sum_z n^{(c_m)}_{z_{t-1},z,\neg t} + 1 + N_z\gamma_{c_m}} \frac{n^{(c_m)}_{z,z_{t+1},\neg t} + \gamma_{c_m}}{\sum_{z_{t+1}} n^{(c_m)}_{z,z_{t+1},\neg t} + N_z\gamma_{c_m}}, \\
&if \quad z_{t-1} = z \neq z_{t+1}: \\
&= \frac{n^{(c_m)}_{z_{t-1},z,\neg t} + \gamma_{c_m}}{\sum_z n^{(c_m)}_{z_{t-1},z,\neg t} + N_z\gamma_{c_m}} \frac{n^{(c_m)}_{z,z_{t+1},\neg t} + \gamma_{c_m}}{\sum_{z_{t+1}} n^{(c_m)}_{z,z_{t+1},\neg t} + 1 + N_z\gamma_{c_m}}.
\end{aligned}
$$

Here $n^{(c_m)}_{z_{t-1},z,\neg t}$ denotes the count from all the videos with the label $c_m$ where motion-pattern $z$ is followed by motion-pattern $z_{t-1}$ excluding the token at $t$. $n^{(c_m)}_{z,z_{t+1},\neg t}$ denotes the count from all the videos with label $c_m$ where motion-pattern $z_{t+1}$ is followed by motion-pattern $z_t$ excluding the token at $t$. $N_z$ is the number of motion-patterns. The Gibbs sampling algorithm iterates between Equations 1 and 2 and finds the approximate posterior distribution. To obtain the resulting model parameters $\{\phi, \theta, \phi\}$ from the Gibbs sampler, we use the expectation of their distribution (Heinrich, 2008), and collect $N_s$ such samples of the model parameters.

For inference, we need to find the best motion-pattern sequence for a new video. The Gibbs sampler draws $N_s$ samples of parameters during the learning phase. We assume that these are sufficient statistics for the model and that no further adaptation of parameters is necessary. We then adopt the Viterbi decoding algorithm to find the best motion-pattern sequence. We approximate the integral over $\phi, \theta, \psi$ using the point estimates obtained during learning. To formulate the recursive equation for the Viterbi algorithm, we can define the quantity

$$
\begin{aligned}
\delta_t(i) &= \max_{z_1,\ldots,z_{t-1}} \int_{\phi,\theta,\psi_{c_m}} p(z_{1:(t-1)}, z_t = i, x_{1:t}|\phi, \theta, \psi_{c_m}), \\
&\approx \max_{z_1,\ldots,z_{t-1}} \left( \frac{1}{N_s} \sum_s p(z_{1:(t-1)}, z_t = i, x_{1:t}|\phi^s, \theta^s, \psi^s_{c_m}) \right),
\end{aligned}
$$

that is $\delta_t(i)$ is the best score at time $t$, which accounts for first $t$ motion-segments and ends in motion-pattern $i$. By induction we have

$$
\delta_{t+1}(j) \approx \max_i \delta_t(i) \frac{1}{N_s} \sum_s p(z_{t+1} = j|z_t = i, \psi^s_{c_m}) p(x_{t+1}|z_{t+1} = j, \theta^s, \phi^s). \tag{3}
$$

To find the best motion-pattern, we need to keep track of the arguments that maximized Equation 3. For the classification task we calculate the likelihood $p^\star$ for each class and assign the label which has maximum value in:

$$
p^\star = \max_{1 \leq j \leq N_z} \delta_T(j).
$$

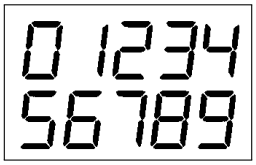


Figure 3: Digital digits for simulations

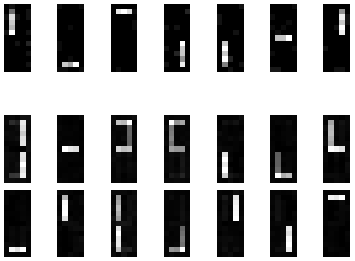


Figure 4: The top row shows seven poses discovered from clustering words. The middle and bottom rows show the fourteen motion patterns discovered and modeled from the poses. A motion pattern captures one or more strokes in the order they are written

## 4. Experiments and Results using MCMCLDA

In this section, we present our observations as well as the results of applying our proposed language-motivated hierarchical model to sub-action analysis as well as to activity classification, using both simulated data as well as a publicly available benchmark data set.

### 4.1 Study Performed on Simulated Digit Data

To flesh out the details of our proposed hierarchical classification model, we present a study performed on simulated data. The ten simulated dynamic activity classes were the writing of the ten digital digits, 0-9 as shown in Figure 3. The word vocabulary was made up of all the pixels in a $13 \times 5$ grid and the topics or poses represented the distribution over the words. An activity class therefore consisted of the steps needed to simulate the writing of each digit and the purpose of the simulation was to visually observe the clusters of motion patterns involved in the activities.

#### 4.1.1 ANALYSIS OF RESULTS

A total of seven clusters were discovered and modeled, as shown in Figure 4. These represent the simulated strokes (or topics) involved in writing each digit. There were fourteen motion patterns discovered, as shown in the two bottom rows of Figure 4. These are the probabilistic clusters of the stroke motions. An activity or digit written was therefore classified based on the sequences of distributions of these motion patterns over time.

### 4.2 Study Performed on the Daily Activities Data Set

The *Daily Activities data set* contains high resolution ($1280 \times 760$ at 30 fps) videos, with 10 different complex daily life activities such as *eating banana, answering phone, drinking water, etc.*. Each activity was performed by five subjects three times, yielding a total of 150 videos. The duration of each video varied between 10 and 60 seconds.

We generated visual words for the MCMCLDA model in a manner similar to Laptev (2005), where the Harris3D detector (Laptev and Lindeberg, 2003) was used to extract space-time interest points at multiple scales. Each interest point was described by the concatenation of HoF and HoG (Laptev, 2005) descriptors. After the extraction of these descriptors for all the training videos, we used the k-means clustering algorithm to form a codebook of descriptors (or visual words (VW)). Furthermore, we vector-quantized each descriptor by calculating its membership with respect to the codebook. We used the original implementation available online[2] with the standard parameter settings to extract interest points and descriptors.

Due to the limitations of the distributed implementation of space-time interest points (Laptev et al., 2008), we reduced the video resolution to 320 x 180. In our experimental setup, we used 100 videos for training and 50 videos for testing exactly as pre-specified by the original publishers of this data set (Messing et al., 2009). Both the training and testing sets had a uniform distribution of samples for each activity. We learned our MCMCLDA model on the training videos, with a motion segment size of 15 frames. We ran a Gibbs sampler for a total of 6000 iterations, ignoring the first 5000 sweeps as burn-in, then took 10 samples at a lag of 100 sweeps. The hyperparameters were fixed initially with values $(\alpha = 5, \beta = 0.01, \gamma = 1)$ and after burn-in, these values were empirically estimated using maximum-likelihood estimation (Heinrich, 2008) as $(\alpha = 0.34, \beta = 0.001$ and $\gamma = \{0.04, 0.05, 0.16, 0.22, 0.006, 0.04, 0.13, 0.05, 0.14, 0.45\})$. We set the number of motion-patterns, poses and codebook size experimentally as $N_z = 100, N_y = 100$ and $N_x = 1000$.

The confusion matrix computed from this experiment is given in Figure 5 and a comparison with other activity recognition methods on the Daily Activities data set is given in Table 1. Because the data set was already pre-divided, the other recognition methods reported in Table 1 were trained and tested on the same sets of training and testing videos.

Qualitatively, Figure 7 pictorially illustrates some examples of different activities having the same underlying shared motion patterns.

#### 4.2.1 ANALYSIS OF RESULTS

We present comparative results with other systems in Table 1. The results show that the approach based on computing a distribution mixture over motion orientations at each spatial location of the video sequence (Benabbas et al., 2010), slightly outperformed our hierarchical model. Interestingly, in our test, one activity, the *write on whiteboard (wW)* activity is quite confused with *use silverware (uS)* activity, significantly bringing down the overall accuracy. The confusion matrix for Benabbas et al. (2010) is presented in Figure 6 and it shows several of the classes being confused, no perfect recognition scores and also one of the class recognition rates being below 50%. Being a generative model, the MCMCLDA model performs comparably to other discriminative models in a class labeling task.

2. Implementation can be found at `http://www.irisa.fr/vista/Equipe/People/Laptev/download.html#stip`.
3. The authors also reported velocity trajectory feature augmented with prior spatial layout information, resulting in an accuracy of 89%.

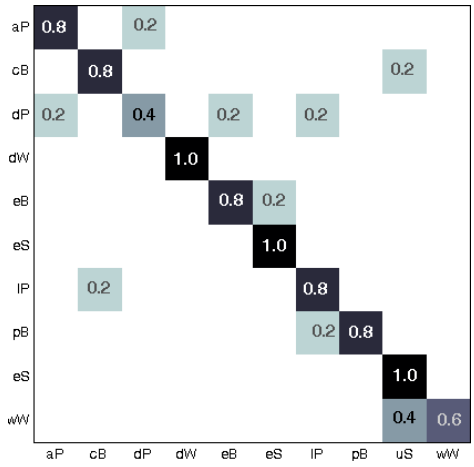



Figure 5: Confusion matrix for analyzing the University of Rochester daily activities data set. Overall accuracy is 80.0%. Zeros are omitted for clarity. The labels and their corresponding meaning are: ap-answer phone; cB-chop banana; dP-dial phone; dW-drink water; eB-eat banana; eS-eat snack; lP-lookup in phonebook; pB-peel banana; uS-use silverware; wW-write on whiteboard.

| **Technique** | **Focus** | **Accuracy** |
|---|---|---|
| Latent velocity trajectory features (Messing et al., 2009)[3] | motion feature enhancement | 67% |
| Naive-Bayes pairwise trajectory features (Matikainen et al., 2010) | motion feature enhancement | 70% |
| Salient region tracking features (Bilen et al., 2011) | motion feature enhancement | 74% |
| Video temporal cropping technique | motion feature enhancement | 80% |
| Our supervised dynamic hierarchical model | dynamic hierarchical modeling | 80% |
| Direction of motion features (Benabbas et al., 2010) | motion feature enhancement | 81% |

Table 1: The accuracy numbers reported in literature from applying different activity recognition techniques on the daily activities data set

Figure 7 pictorially illustrates some examples of different activities having the same underlying shared motion patterns. For example, the activity of answering the phone shares a common motion pattern (#85) with the activities of dialing the phone and drinking water. Semantically, we observe that this shared motion is related to the *lifting* sub-action.

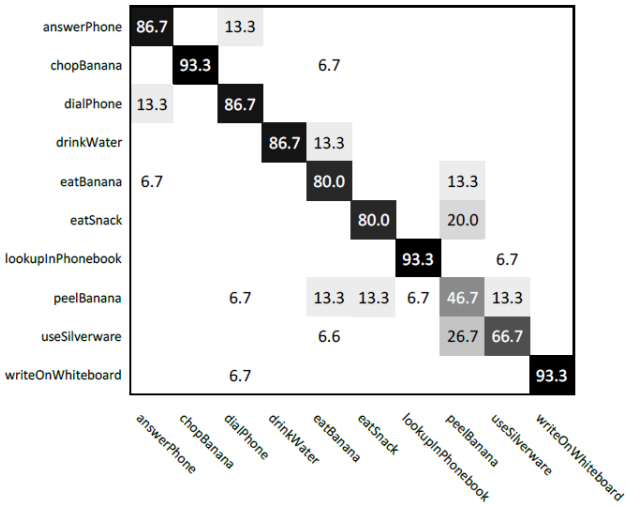



Figure 6: Confusion matrix results from Benabbas et al. (2010) on the University of Rochester daily activities data set.

## 5. A Language-Motivated Model for Gesture Recognition and Spotting

Few methods have been proposed for gesture spotting and among them include the work of Yuan et al. (2009), who represented a video as a 3D volume and activities-of-interest as sub-volumes. The task of gesture spotting was therefore reduced to performing an optimal search for gestures in the video. Another work in spotting was presented by Derpanis et al. (2010) who introduced a local descriptor of video dynamics based on visual space-time oriented energy measures. Similar to the previous work, their input was also a video in which a specific action was searched for. The limitation in these techniques is their inability to adapt to changes in view points, scale, appearance, etc. Rather than being defined on the motion patterns involved in an activity, these methods performed a type of 3D template matching on sequential data; such methods do not readily generalize to new environments exhibiting the known activity. *We therefore propose to develop a probabilistic framework for gesture spotting that can be learned with very little training data and can readily generalize to different environments*.

Justification: Although the proposed framework is a generative probabilistic model, it performs comparably to the state-of-the-art activity techniques which are typically discriminative in nature, as demonstrated in Tables 2 and 3. An additional benefit of the framework is its usefulness for gesture spotting based on learning from only one, or few training examples.

Background: In speech recognition, unconstrained keyword spotting refers to the identification of specific words uttered, when those words are not clearly separated from other words, and no grammar is enforced on the sentence containing them. Our proposed spotting framework uses the Viterbi decoding algorithm and is motivated by the *keyword-filler HMM for spotting keywords in continuous speech*. The current state of the art keyword filler HMM dates back to the seminal papers of Rohlicek et al. (1989) as well as Rose and Paul (1990), where the basic idea is to create one HMM of the keyword and a separate HMM of the filler or non keyword regions. These two models are then

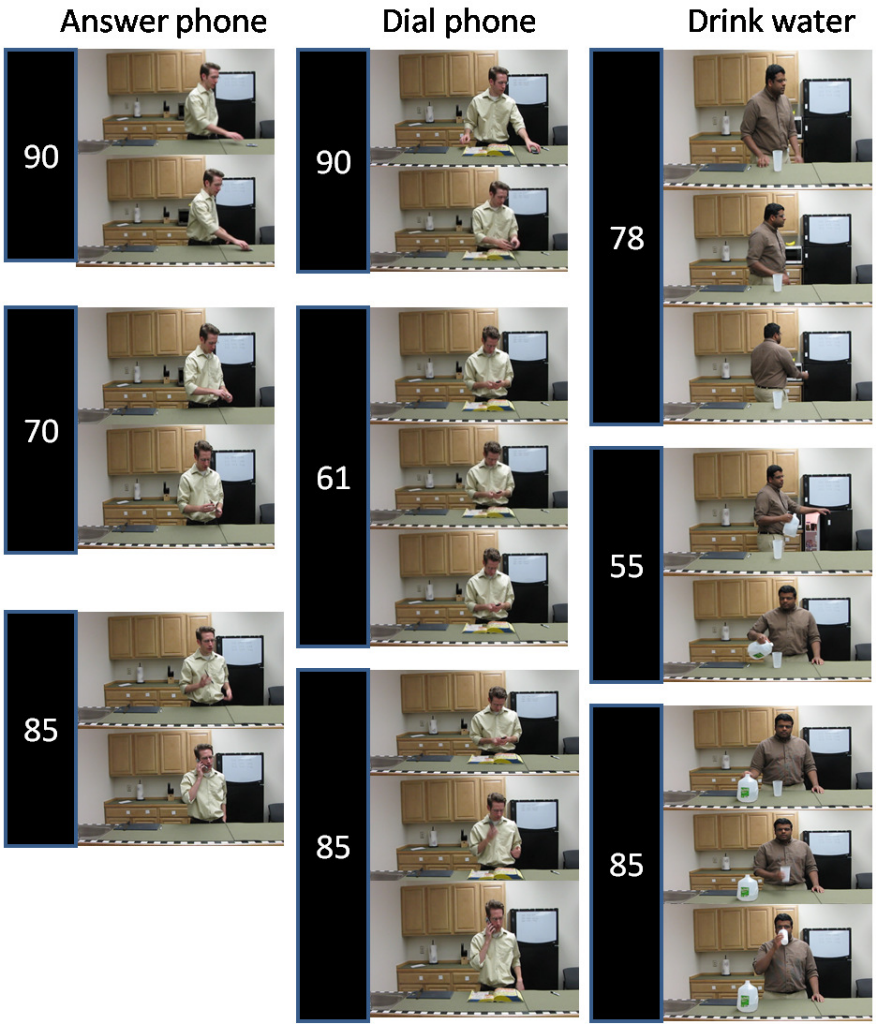



Figure 7: Different activities showing shared underlying motion patterns. The shared motion patterns are 85 and 90, amidst other underlying motion patterns shown

combined to form a composite filler HMM that is used to annotate speech parts using the Viterbi decoding scheme. Putative decisions arise when the Viterbi path crosses the keyword portion of the model. The ratio between the likelihood of the Viterbi path that passes through the keyword model and the likelihood of an alternate path that passes solely through the filler portion can be used to score the occurrence of keywords. In a similar manner, we compute the probabilistic signature for a gesture class, and using the filler model structure, we test for the presence of that gesture within a given video. For one-shot learning, the parameters of the single training video are considered to be sufficiently representative of the class.

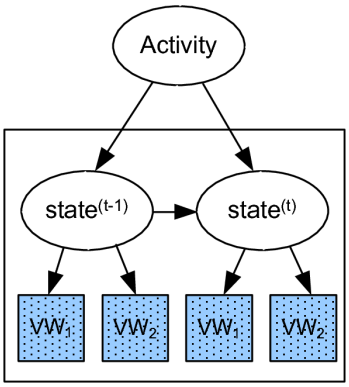



Figure 8: Plates model for mcHMM showing the relationship between activities or gestures, states and the two channels of observed visual words (VW).

### 5.1 Gesture Recognition using a Multichannel Dynamic Bayesian Network

In a general sense, the spotting model can be interpreted as an HMM (whose random variables involve hidden states and observed input nodes) but unlike the classic HMM, this model has multiple input channels, where each channel is represented as a distribution over the visual words corresponding to that channel. In contrast to the classic HMM, our model can have multiple observations per state and channel, and we refer to this as the multiple channel HMM (mcHMM). Figure 8 shows a graphical representation of the mcHMM.

### 5.2 Parameter Estimation for the Gesture Recognition Model

To determine the probabilistic signature of an activity class, one mcHMM is trained for each activity. The generative process for mcHMM involves first sampling a state from an activity, based on the transition matrix for that activity; then a frame-feature comprising of the distribution of visual words is sampled according to a multinomial distribution for that state[4] and this is repeated for each frame. Similar to a classic HMM, the parameters for the mcHMM are therefore:

1. Initial state distribution $\pi = \{\pi_i\}$,
2. State transition probability distribution $A = \{a_{ij}\}$,
3. Observation densities for each state and descriptor $B = \{b_i^d\}$.

The joint probability distribution of observations (O) and hidden state sequence (Q) given the parameters of the multinomial representing a hidden state ($\lambda$) can be expressed as:

$$P(O,Q|\lambda) = \pi_{q_1} b_{q_1}(O_1) \prod_{t=2}^{T} a_{q_{t-1}q_t} \cdot b_{q_t}(O_t),$$

where $b_{q_t}(O_t)$ is modeled as follows:

$$\begin{aligned} b_{q_t}(O_t) = & \quad \prod_{d=1}^{D} b_q^d(O_t^d), \\ = & \quad \prod_{d=1}^{D} Mult(O_t^d|b_q^d), \end{aligned}$$

4. States are modeled as multinomials since our input observables are discrete values.

and $D$ is the number of descriptors.

EM is implemented to find the maximum likelihood estimates. The update equations for the model parameters are:

$$\hat{\pi} = \sum_{r=1}^{R} \gamma_1^r(i),$$
$$\hat{a}_{ij} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T}\eta_t^r(i,j)}{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^r(i)},$$
$$\hat{b}_j^d(k) = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^r(j)\cdot\frac{n_t^{d,k}}{n_t^{d,\cdot}}}{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^r(j)},$$

where $R$ is number of videos and $\gamma_1(i)$ is the expected number of times the activity being modeled started with state $i$;

$\eta_t^r(i,j)$ is the expected number of transitions from state $i$ to state $j$ and $\gamma_t^r(i)$ is the expected number of transitions from state $i$;

$n_t^{d,k}$ is the number of times that visual word $k$ occurred in descriptor $d$ at time t and $n_t^{d,\cdot}$ is the total number of visual words that occurred in descriptor $d$ at time $t$.

### 5.3 Gesture Spotting via Inference on the Model

The gesture spotting problem is thus reduced to an inference problem where, given a new not-previously-seen test video, and the model parameters or probabilistic signatures of known activity classes, the goal is to establish which activity class distributions most likely generated the test video. This type of inference can be achieved using the Viterbi algorithm.

We constructed our spotting network such that there could be a maximum of five gestures in a video. This design choice was driven by our participation in the Chalearn competition where there was a maximum of five gestures in every test video. Each of these gesture classes was seen during training, hence, there were no random gestures inserted into the test video. This relaxed our network, compared to the original filler model in speech analysis, where there can exist classes that have not been previously seen. Figure 9 shows an example of the stacked mcHMMs involved the gesture spotting task. This toy example shown in the figure can spot gestures in a test video comprised of at most two gestures. This network has a non-emitting start state $S'$. This state does not have any observation density associated with it. From this state, we can enter any of $K$ gestures, which is shown by having edges from $S'$ to $K$ mcHMMs. All the gestures are then connected to non-emitting state $S''$ which represents the end of first gesture. Similarly we can enter the second gesture from $S''$ and end at $e'$ or directly go from $S''$ to $e'$ which handles the case for a video having only one gesture. This can be easily extended to the case where there are at most five gestures.

The Viterbi decoding algorithm was implemented to traverse the stacked network and putative decisions arose when the Viterbi path crosses the keyword portion of the model. The ratio between the likelihood of the Viterbi path that passed through the keyword model and the likelihood of an alternate path that passes through the non-keyword portion was then used to score the occurrence of a keyword, where a keyword here referred to a gesture class. An empirically chosen threshold value was thus used to select the occurrence of a keyword in the video being decoded.

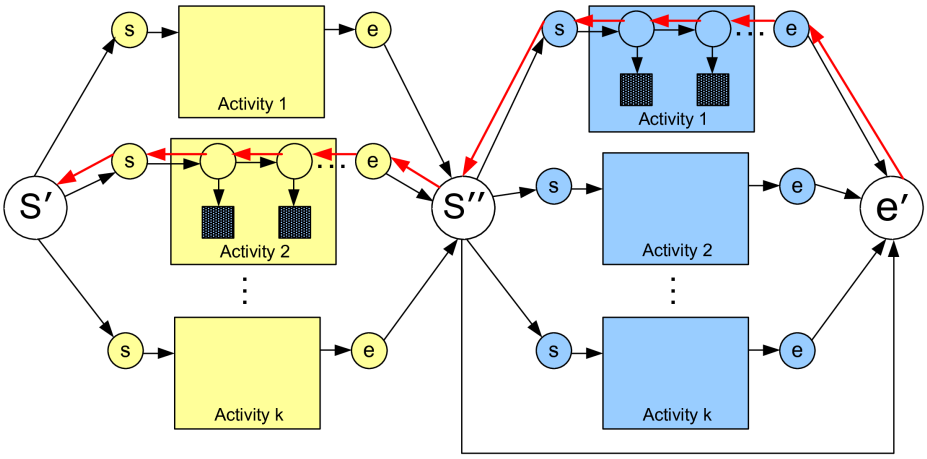



Figure 9: Activity spotting by computing likelihoods via Viterbi decoding. The toy example shown assumes there are at most two activities in any test video, where the first activity is from the set of activities that start from $s'$ and end at $s''$, followed by one from the set that start from $s''$ and end at $e'$. The image also shows an example of a putative decision path from $e'$ to $s'$, after the decoding is completed

## 6. Experiments and Results using mcHMM

In this section, we present our approach on generating visual words and our observations as well as the results of applying proposed mcHMM model to activity classification and gesture spotting, using publicly available benchmark data sets.

### 6.1 Generating Visual Words

An important step in generating visual words is the the need to extract interest points from frames sampled from the videos at 30 fps. Interest points were obtained from the KTH and HMDB data set by sampling dense points in every frame in the video and then tracking these points for the next $L$ frames. These are known as dense trajectories. For each of these tracks, motion boundary histogram descriptors based on HoG and HoF descriptors were extracted. These features are similar to the ones used in dense trajectories (Wang et al., 2011), although rather than sampling interest points at every $L$ frames or when the current point is lost before being tracked for $L$ frames, we sampled at every frame. By so doing, we obtained a better representation for each frame, whereas the original work used the features to represent the whole video and was not frame-dependent.

Because the HMDB data set is comprised of real-life scenes which contain people and activities occurring at multiple scales, the frame-size in the video was reduced by a factor of two repeatedly, and motion boundary descriptors were extracted at multiple scales. In the Chalearn data set, since the videos were comprised of RGB-depth frames, we extracted interest points by (i) taking the difference between two consecutive depth frames and/or (ii) calculating the centroid of the depth foreground in every frame and computing the extrema points (from that centroid) in the depth foreground. The second process ensured that extrema points such as the hands, elbows, top-of-the-head,

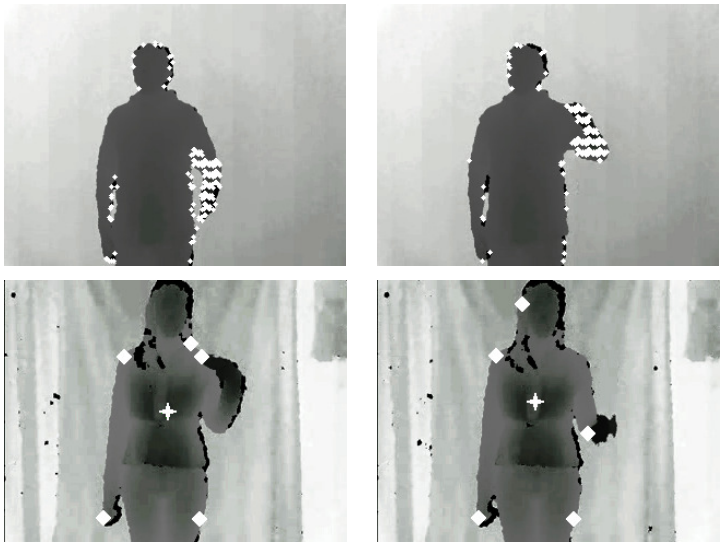


Figure 10: Interest points for 2 consecutive video frames. Top: Depth-subtraction interest points; bottom: extrema interest points (with centroid).

etc., were always included in the superset of interest points. The top and bottom image pairs in Figure 10 show examples of consecutive depth frames from the Chalearn data set, with the interest points obtained via the two different methods, superimposed. Again, HoG and HoF descriptors were extracted at each interest point so that similar descriptors could be obtained in all the cases. We used a patch size of $32 \times 32$ and a bin size of 8 for HoG and 9 for HoF implementation.

The feature descriptors were then clustered to obtain visual words. In general, from the literature (Wang et al., 2011; Laptev et al., 2008), in order to limit complexity, researchers randomly select a finite number of samples (roughly in the order of 100,000) and cluster these to form visual words. This could prove reasonable when the number of samples is a few orders of magnitude greater than 100,000. But in dealing with densely sampled interest points at every frame, the amount of descriptors generated especially at multiple scales become significantly large. We therefore divided the construction of visual words for HMDB data set into a two step process where visual words were first constructed for each activity class separately, and then the visual words obtained for each class were used as the input samples to cluster the final visual words. For the smaller data sets such as KTH and Chalearn Gesture Data Set, we randomly sampled 100,000 points and clustered them to form the visual words.

### 6.2 Study Performed on the HMDB and KTH Data Sets

In order to compare our framework to the other current state-of-the-art methods, we performed activity classification on video sequences created from the KTH database (Schüldt et al., 2004); KTH is a relatively simplistic data set comprised of 2391 video clips used to train/test six human actions. Each action is performed several times by 25 subjects in various outdoor and indoor scenarios. We split the data into training set of 16 subjects and test set of 9 subjects, which is exactly the same

| Method | Accuracy |
|---|---|
| (Laptev et al., 2008) | 91.8% |
| (Yuan et al., 2009) | 93.3% |
| (Wang et al., 2011) | 94.2% |
| (Gilbert et al., 2011) | 94.5% |
| (Kovashka and Grauman, 2010) | 94.53% |
| proposed mcHMM | **94.67** |

Table 2: Comparison of our proposed model and features for KTH data set

| Method | Accuracy |
|---|---|
| Best results on 51 activities (original) | |
| (Kuehne et al., 2011) | 23.18% |
| Proposed mcHMM on 51 activities (original) | **25.64%** |
| Best results on 10 activities (original) | |
| (Kuehne et al., 2011) | 54.3% |
| Proposed mcHMM on 10 activities (original) | **57.67%** |
| Proposed mcHMM on 10 activities (stabilized) | **66.67%** |

Table 3: Comparison of our proposed model and features for the HMDB data set

setup used by the authors of the initial paper (Schüldt et al., 2004). Table 2 shows the comparison of accuracies obtained.

Similarly, we performed activity classification tests on Human Motion Database (HMDB) (Kuehne et al., 2011). HMDB is currently the most realistic database for human activity recognition comprising of 6766 video clips and 51 activities extracted from a wide range of sources like YouTube, Google videos, digitized movies and other videos available on the Internet. We follow the original experimental setup using three train-test splits (Kuehne et al., 2011). Each split has 70 video for training and 30 videos for testing for each class. All the videos in the data set are stabilized to remove the camera motion and the authors of the initial paper (Kuehne et al., 2011) report results on both original and stabilized videos for 51 activities. The authors also selected 10 common actions from HMDB data set that were similar to action categories in the UCF50 data set (ucf, 2010) and compared the recognition performance. Table 3 summarizes the performance of proposed mcHMM method on 51 activities as well as 10 activities for both original and stabilized videos.

### 6.2.1 ANALYSIS OF RESULTS

For both the case of simple actions as found in the KTH data set and the case of significantly more complex actions as found in the HMDB data set, the mcHMM model performs comparably with other methods, outperforming them in the activity recognition task. Our evaluation against state-of-the-art data sets suggest that performance is not significantly affected over a range of factors such as camera position and motion as well as occlusions. This suggests that the overall framework (combination of dense descriptors and a state-based probabilistic model) is fairly robust with respect to these low-level video degradations. At the time of this submission, although we outperformed the only currently reported accuracy results on the HMDB data set, as shown by the ccuracy scores

reported, the framework is still limited in its representative power to capture the complexity of human actions.

### 6.3 Study Performed on the ChaLearn Gesture Data Set

Lastly, we present our results of gesture spotting from the ChaLearn gesture data set (Cha, 2011). The ChaLearn data set consisted of video frames with RGB-Depth information. Since the task-at-hand was gesture spotting via one-shot learning, only one video per class was provided to train an activity (or gesture). The data set was divided into three parts: development, validation and final. In the first phase of the competition, participants initially developed their techniques against the development data set. Ground truth was not provided during the development phase. Once the participants had a working model, they then ran their techniques against the validation data set and uploaded their predicted results to the competition website, where they could receive feedback (scores based on edit distances) on the correctness of the technique. In the last phase of the competition, the final data set was released so that participants could test against it and upload their predicted results. Similarly, edit scores were used to measure the correctness of the results and the final rankings were published on the competition website.

We reported results using two methods i) mcHMM ii) mcHMM with LDA (Blei et al., 2003). For mcHMM method, we constructed visual words as described in Section 6.1 and represented each frame as two histograms of visual words. This representation was input to the model to learn parameters of the mcHMM model. In the augmented framework, mcHMM + LDA, the process of applying LDA to the input data can be viewed as a type of dimensionality reduction step since the number of topics are usually significantly smaller than the number of unique words. In our work, a frame is analogous to a document and visual words are analogous to words in a text document. Hence, in the combined method, we performed the additional step of using LDA to represent each frame as a histogram of topics. These reduced-dimension features were input to the mcHMM model. Gesture spotting was then performed by creating a spotting network made up of connected mcHMM models, one for each gesture learned, as explained in Section 5.3.

For the mcHMM model, we experimentally fixed the number of states to 10. The number of visual words was computed as the number of classes multiplied by a factor of 10, for example if the number of classes is 12, then then number of visual words generated will be 120. The dimensionality of the input features to the mcHMM model was the number of visual words representing one training sample. For the augmented model the dimension of the features was reduced by a factor of 1.25, that is in the previous example, the length of feature vector would be reduced from 120 to 96. All the above parameters were experientially found using the development set. The same values were then used for the validation and final sets.

#### 6.3.1 ANALYSIS OF RESULTS

Table 4 shows the results of one-shot-learning on the ChaLearn data at the three different stages of the competition. We present results based on the two variants of our framework—the mcHMM model framework and the augmented mcHMM + LDA framework. Our results indicate that the framework augmented with LDA outperforms the unaugmented one, two out of three times. During implementation, the computational performance for the augmented framework was also significantly better than the unaugmented model due the reduced number of features needed for training and for inference. It is also interesting to observe how the edit distances reduced from the devel-

| Method | Data Set | edit distance |
|---|---|---|
| proposed mcHMM | Development | 0.26336 |
| proposed mcHMM + LDA | Development | 0.2409 |
| baseline | Validation | 0.59978 |
| proposed mcHMM | Validation | 0.26036 |
| proposed mcHMM + LDA | Validation | 0.23328 |
| Top Ranking Participant | Validation | 0.20287 |
| Top Ranking Participant | Final | 0.09956 |
| proposed mcHMM + LDA | Final | 0.18465 |

Table 4: Results for ChaLearn gesture data set

opment phase through the final phase, dropping by up to six percentage points, due to parameter tuning. We placed fourth place in the final results of round 1 of the Chalearn 2012 gesture challenge using the augmented method.

## 7. Conclusion and Future Work

In the course of this paper, we have investigated the use of motion patterns (representing sub-actions) exhibited during different complex human activities. Using a language-motivated approach we developed a dynamic Bayesian model which combined the temporal ordering power of dynamic Bayesian networks with the automatic clustering power of hierarchical Bayesian models such as the LDA word-topic clustering model. We also showed how to use the Gibbs samples for rapid Bayesian inference of video segment clip category. Being a generative model, we can detect abnormal activities based on low likelihood measures. This framework was validated by its comparable performance on tests performed on the daily activities data set, a naturalistic data set involving everyday activities in the home.

We also investigated the use of a multichannel HMM as a generative probabilistic model for single activities and it performed comparably to the state-of-the-art activity classification techniques which are typically discriminative in nature, on two extreme data sets—the simplistic KTH, and the very complex and realistic HMDB data sets. An additional benefit of this framework was its usefulness for gesture spotting based on learning from only one, or few training examples. We showed how the use of the generative dynamic Bayesian model naturally lent itself to the spotting task, during inference. The efficacy of this model was shown by the results obtained from participating in ChaLearn Gesture Challenge where an implementation of the model finished top-5 in the competition.

In the future, we will consider using the visual words learned from a set of training videos to automatically segment a test video. The use of auto-detected video segments could prove useful both in activity classification and gesture spotting. It will also be interesting to explore the use of different descriptors available in the literature, in order to find those best-suited for representing naturalistic videos.

## Acknowledgments

The authors wish to thank the associate editors and anonymous referees for all their advice about the structure, references, experimental illustration and interpretation of this manuscript. The work benefited significantly from our participation in the ChaLearn challenge as well as the accompanying workshops.

## References

*University of Central Florida, Computer Vision Lab*, 2010. URL `http://server.cs.ucf.edu/~vision/data/UCF50.rar`.

ChaLearn Gesture Dataset (CGD2011), ChaLearn, California, 2011. URL `http://gesture.chalearn.org/2011-one-shot-learning`.

J. Aggarwal and M. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43: 16:1–16:43, Apr 2011.

Y. Benabbas, A. Lablack, N. Ihaddadene, and C. Djeraba. Action Recognition Using Direction Models of Motion. In *Proceedings of the 2010 International Conference on Pattern Recognition*, pages 4295–4298, 2010.

H. Bilen, V. P. Namboodiri, and L. Van Gool. Action recognition: A region based approach. In *Proceedings of the 2011 IEEE Workshop on the Applications of Computer Vision*, pages 294 –300, 2011.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, Mar. 2003.

M. Bregonzio, S. Gong, and T. Xiang. Recognising action as clouds of space-time interest points. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1948–1955, 2009.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

K. G. Derpanis, M. Sizintsev, K. Cannons, and R. P. Wildes. Efficient action spotting based on a spacetime oriented structure representation. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1990–1997, 2010.

P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Proceedings of the 2005 IEEE Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

A. Gilbert, J. Illingworth, and R. Bowden. Action recognition using mined hierarchical compound features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):883–897, 2011.

S. Gong and T. Xiang. Recognition of group activities using dynamic probabilistic networks. In *Proceedings of the 2003 IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–749 vol.2, 2003.

G. Heinrich. Parameter estimation for text analysis,. Technical report, University of Leipzig, 2008.

T. Hospedales, S.-G. Gong, and T. Xiang. A Markov Clustering Topic Model for Mining Behaviour in Video. In *Proceedings of the 2009 International Conference on Computer Vision*, pages 1165–1172, 2009.

A. Kläser, M. Marszalek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *Proceedings of the 2008 British Machine Vision Conference*, 2008.

A. Kovashka and K. Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2046–2053, 2010.

H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of the 2011 International Conference on Computer Vision*, 2011.

I. Laptev. On Space-Time Interest Points. *International Journal of Computer Vision*, 64:107–123, September 2005.

I. Laptev and T. Lindeberg. Space-time interest points. In *Proceedings of the 2003 International Conference on Computer Vision*, pages 432–439, 2003.

I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning Realistic Human Actions From Movies. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

P. Matikainen, M. Hebert, and R. Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *Proceedings of the 2009 IEEE Workshop on Video-Oriented Object and Event Classification*, Sep 2009.

P. Matikainen, M. Hebert, and R. Sukthankar. Representing Pairwise Spatial and Temporal Relations for Action Recognition. In *Proceedings of the 2010 European Conference on Computer Vision*, September 2010.

R. Messing, C. Pal, and H. Kautz. Activity Recognition Using the Velocity Histories of Tracked Keypoints. In *Proceedings of the 2009 International Conference on Computer Vision*, 2009.

P. Natarajan and R. Nevatia. Coupled hidden semi markov models for activity recognition. In *Proceedings of the IEEE Workshop on Motion and Video Computing*, 2007.

N. T. Nguyen, D. Q. Phung, and S. Venkatesh. Learning and detecting activities from movement trajectories using the hierarchical hidden markov models. In *Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition*, pages 955–960, 2005.

E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Proceedings of the 2006 European Conference on Computer Vision*, pages 490–503, 2006.

N. Oliver, E. Horvitz, and A. Garg. Layered representations for human activity recognition. In *Proceedings of the 2002 IEEE International Conference on Multimodal Interfaces*, pages 3–8, 2002.

N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8): 831–843, 2000.

J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish. Continuous hidden Markov modeling for speaker-independent word spotting. In *Proceedings of the 1989 International Conference on Acoustics, Speech, and Signal Processing*, pages 627–630, 1989.

R. Rose and D. Paul. A Hidden Markov Model based keyword recognition system. In *Proceedings of the 1990 International Conference on Acoustics, Speech, and Signal Processing*, 1990.

C. Schüldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *Proceedings of the 2004 International Conference on Pattern Recognition*, pages 32–36, 2004.

P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Procedings of the ACM International Conference on Multimedia*, pages 357–360, 2007.

H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *Proceedings of the 2009 British Machine Vision Conference*, sep 2009.

H. Wang, A. Kläser, C. Schmid, and L. Cheng-Lin. Action Recognition by Dense Trajectories. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3169–3176, Jun 2011.

G. Willems, T. Tuytelaars, and L. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proceedings of the 2008 European Conference on Computer Vision*, pages 650–663, 2008.

J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of the 1992 IEEE Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.

L. Yeffet and L. Wolf. Local trinary patterns for human action recognition. In *Proceedings of the 2009 International Conference on Computer Vision*, 2009.

J. Yuan, Z. Liu, and Y. Wu. Discriminative subvolume search for efficient action detection. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Journal of Machine Learning Research 14 (2013) 2213-2238 Submitted 11/12; Revised 3/13; Published 8/13

# Segregating Event Streams and Noise with a Markov Renewal Process Model

**Dan Stowell** DAN.STOWELL@EECS.QMUL.AC.UK
**Mark D. Plumbley** MARK.PLUMBLEY@EECS.QMUL.AC.UK
*Centre for Digital Music*
*Queen Mary University of London*
*London, E1 4NS, United Kingdom*

**Editor:** Samy Bengio

## Abstract

We describe an inference task in which a set of timestamped event observations must be clustered into an unknown number of temporal sequences with independent and varying rates of observations. Various existing approaches to multi-object tracking assume a fixed number of sources and/or a fixed observation rate; we develop an approach to inferring structure in timestamped data produced by a mixture of an unknown and varying number of similar Markov renewal processes, plus independent clutter noise. The inference simultaneously distinguishes signal from noise as well as clustering signal observations into separate source streams. We illustrate the technique via synthetic experiments as well as an experiment to track a mixture of singing birds. Source code is available.

**Keywords:** multi-target tracking, clustering, point processes, flow network, sound

## 1. Introduction

Various approaches exist for the task of inferring the temporal evolution of multiple sources based on joint observations (Mahler, 2007; Van Gael et al., 2009). They are generally based on a model in which sources are continuously observable, in the sense that they are expected to emit/return observations at every time step (though there may be missed detections). Yet there are various types of source for which observations are inherently intermittent, and for which this intermittence exhibits temporal structure that can be characterised as a point process. Examples include sound event sequences such as bird calls or footsteps (Wang and Brown, 2006), internet access logs (Arlitt and Williamson, 1997), pulsars in astronomy (Keane et al., 2010) and neural firing patterns (Bobrowski et al., 2009). Intermittent observations are also often output from *sparse representation* techniques, which transform signals into a representation with activations distributed sparsely in time and state (Plumbley et al., 2010).

In this paper we describe a generic problem setting that may be applied to such data, along with an approach to estimation. We are given a set of timestamped data, and we assume each datum is produced by one of a set of similar but independent signal processes, or by a "clutter" noise process, with known parameters. We do not know the true partitioning of the data into sequences each generated by a single process, and wish to infer this. We do not know how many processes are active, and we do not assume that each process produces the same number of observations, or observations at the same time points.

©2013 Dan Stowell and Mark D. Plumbley.

This specific type of clustering problem has applications in various domains. For example, when sparse representation techniques are used for source separation in time series, they often yield a set of atomic activations which must be clustered according to their underlying source, and preferably to discard any spurious noise activations (Plumbley et al., 2010). Temporal dependence information may help to achieve this (cf. Mysore et al. 2010). Timestamped data such as internet access logs often contain no explicit user association, yet it may be desirable to group such data by user for for further analysis (Arlitt and Williamson, 1997). In computational audio scene analysis, it is often the case that sound sources emit sound only intermittently during their presence in the scene, yet it is desirable to track their temporal evolution.

### 1.1 Related Work

To our knowledge, this particular problem setting has not been directly addressed in the literature. Temporal data is most commonly treated using a model of sources which update continuously, or synchronously at an underlying temporal sampling rate. Pertinent formulations for our purposes include the infinite factorial hidden Markov model (infinite FHMM) of Van Gael et al. (2009), or the probability hypothesis density filter (PHD filter) (Mahler, 2007), both of which infer an unknown number of independent Markov sources. FHMMs assume that the underlying sources are not intermittent during their lifetime, and also that they persist throughout the whole observation period. Pragmatically, intermittent emissions may be handled by incorporating silence states, though to implement arbitrary-duration silence states may require additional workarounds such as multiple parallel/sequential silences. The PHD filter allows for stochastic missed detections but not for structured intermittency.

Among techniques which do not assume a synchronous update, graph clustering approaches such as normalised cuts have similarities to our approach (Shi and Malik, 2000). In particular, Lagrange et al. (2008) apply normalized cuts in order to cluster temporally-ordered data. However, the normalised cuts method is applied to undirected graphs, and Lagrange et al. (2008) use perceptually-motivated similarity criteria rather than directed Markov dependencies as considered herein. Further, the normalized cuts method does not include a representation of clutter noise, and so Lagrange et al. (2008) perform signal/noise cluster selection as a separate postprocessing step. In the present work we include an explicit noise model.

In automatic speech recognition, segmental models or fragment-decoding models are inferred using a combinatorial graph search through temporal observations (Glass, 2003; Barker et al., 2005), and thus have resonances with the method we will develop here. However they address only a single-source problem. (Barker et al. 2005 considers multi-talker background noise but only one foreground source.)

Our problem setting also exhibits similarities with that of structure discovery in Bayesian networks (Koivisto and Sood, 2004). However, in that context the dependency structure is inferred from correlations present in multiple observations from each vertex in the structure. In the present case we have only one observation per vertex, plus the partial ordering implied by temporality.

In the following we develop a model in which an unknown number of point-process sources are assumed to be active as well as Poisson clutter, and describe how to perform a maximum likelihood inference which clusters the signal into individual identified tracks plus clutter noise. We then demonstrate the performance of the approach in synthetic experiments, and in an experiment analysing birdsong audio.

## 2. Preliminaries

Throughout we will consider sets of observations in the form $\{(X,T)\}$ where $X$ is state and $T$ is time. A Markov renewal process (MRP) generates a sequence of such observations having the Markov property:

$$\begin{aligned} &P(\tau_{n+1} \leq t, X_{n+1} = j \mid (X_1,T_1),\ldots,(X_n = i,T_n)) \\ &\qquad = P(\tau_{n+1} \leq t, X_{n+1} = j \mid X_n = i) \qquad\qquad \forall n \geq 1, t \geq 0, i, j \in \mathcal{S} \end{aligned}$$

where $\tau_{n+1}$ is the time difference $T_{n+1} - T_n$. Note that $\tau$ is not explicitly given in observations $\{(X,T)\}$, but can be inferred if we know that a particular pair of observations are adjacent members within a sequence.

We will have cause to represent our data as a *network flow* problem (Bang-Jensen and Gutin, 2007, Chapter 3). A *network* is a graph supplemented such that each arc $A_{ij}$ has a *lower capacity* $l_{ij}$ and *upper capacity* $u_{ij}$, and a *cost* $a_{ij}$. A *flow* is a function $x : A \to \mathcal{R}_0$ that associates a value with each arc in the network. We will be concerned with integer flows $x : A \to \mathcal{Z}_0$. A flow is *feasible* if $l_{ij} \leq x_{ij} \leq u_{ij}$ for all $A_{ij}$ in the graph, and for all vertices (except for any source/sink vertices) the sum of the inward flow is equal to the sum of the outward flow. For any flow we can calculate a total cost as the sum of $a_{ij}x_{ij}$ over all $A_{ij}$. We define the *value* of a feasible flow to be the sum of $x_{ij}$ over all arcs leading from source vertices.

The standard terminology of flow networks associates capacities, flows and costs with arcs but not vertices. However, in the following we will have cause to associate such attributes with vertices as well as with arcs. This can be implemented transparently by the standard technique of *vertex expansion*, in which each vertex is replaced by an in-vertex and an out-vertex, plus a single arc between them which bears the associated attributes (Bang-Jensen and Gutin, 2007, Section 3.2.4).

## 3. Mixtures of Markov Renewal Processes with Clutter Noise

For the present task, we consider MRPs which are time-limited: each process comes into being at a particular point in time (governed by an independent Poisson process with intensity $\lambda_b(X)$), and after each observation it may "die" with an independent death probability $p_d(X)$. Otherwise it transitions to a new random state-and-time according to the transition distribution $f_x(X,\tau)$. The overall system to be considered is not one but a set of such time-limited MRPs, plus a separate Poisson process that generates clutter noise with intensity $\lambda_c(X)$. The MRPs are independent but share common parameters. We will refer to the overall system (including the noise process) as a *multiple Markov renewal process* system or *MMRP*, in order to clarify when we are referring to the whole system or to a single MRP.

We receive a set of $N$ observations in the form $\{(X,T)\}$ and we assume that they were generated by an MMRP for which the process parameters are known, but the number $K$ of MRPs is unknown as well as the allocation of each observation to its generating process. We assume that each observation is generated either by one MRP or by the noise process. Given these observations as well as model parameters $f_x(X,\tau)$, $\lambda_b$, $p_d$, $\lambda_c$, there are many ways to cluster the observations into $K \in [0,N]$ non-overlapping subsets to represent the assertion that each cluster represents all the emissions from a single MRP, with $H$ of the observations not included in any cluster and considered to be noise. The

overall likelihood under a chosen clustering is given by

$$\text{likelihood} = \prod_{k=1}^{K} p_{\text{MRP}}(k) \prod_{\eta=1}^{H} p'_{\text{NOISE}}(\eta)$$

where $p_{\text{MRP}}(k)$ represents the likelihood of the observation subsequence in cluster $k$ being generated by a single MRP, and $p'_{\text{NOISE}}(\eta)$ represents the likelihood of a single observation datum under the noise model. (A set of clusters is arbitrarily indexed by $k \in [1, K]$.)

In order to find the maximum likelihood solution, we may equivalently divide the likelihood expression through by a constant factor, to give an alternative expression to be maximised. We divide by the likelihood that *all* data were generated by the noise process, to give the likelihood ratio:

$$\text{L} = \prod_{k=1}^{K} \frac{p_{\text{MRP}}(k)}{p_{\text{NOISE}}(k)} \tag{1}$$

where for notational simplicity we use $p_{\text{NOISE}}(k)$ as the joint likelihood of all observations contained within cluster $k$ under the noise model. This likelihood ratio $L$ will shortly be seen to be a convenient expression to optimise—in particular because the likelihoods for the $H$ data points labelled as noise do not need to be considered in (1) since their likelihood ratios are 1 (they have the same likelihood in the numerator and denominator).

The component likelihood ratio for a single cluster $k$ is given by

$$\frac{p_{\text{MRP}}(k)}{p_{\text{NOISE}}(k)} = \frac{p_b(X_{k,1}) \cdot p_d(X_{k,n}) \cdot \prod_{i=2}^{n_k} f_{X_{k,i-1}}(X_{k,i}, T_{k,i} - T_{k,i-1})}{\prod_{i=1}^{n_k} p_c(X_{k,i})}$$

where $(X_{k,i}, T_{k,i})$ refers to the $i$th observation assigned to cluster $k$, this cluster having $n_k$ observations indexed in ascending time order. The term $p_b(\cdot)$ refers to the likelihood associated with a single observation under the Poisson process parametrised by $\lambda_b$, and similarly for $p_c(\cdot)$ for the clutter process parametrised by $\lambda_c$.

The overall likelihood ratio $L$ tells us the relative likelihood that the observation set was generated by the selected clustering of signals and noise, as opposed to the possibility that all observations were generated by clutter noise. Our goal is to find the clustering that yields the highest likelihood ratio, and therefore the set of MRP track identities that is most likely to originate from signal rather than noise.

### 3.1 Network Flow Representation

For any observation set of non-trivial size, there is a combinatorial explosion of possible clusterings available and enumerating them all is intractable. In this subsection we propose to transform the problem into an equivalent problem of network flow, which can be addressed using graph theoretic techniques.

To maximise the likelihood ratio, we can equivalently minimise its negative logarithm, which we will consider as a "cost" for any particular solution. We define additive component costs for

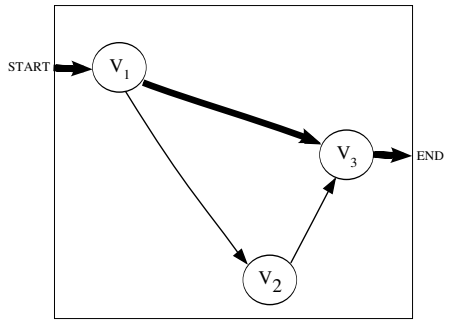



Figure 1: Simple illustration of a path within a network that might correspond to a single MRP sequence. Time increases along the horizontal axis. The bold arrows indicate a path from the first to the third datum (the second datum being left out of the corresponding cluster). The thin arrows indicate an alternative possible path.

birth, death, transition and clutter respectively as:

$$a_b(X) = -\log p_b(X), \tag{2}$$
$$a_d(X) = -\log p_d(X), \tag{3}$$
$$a_t(X, X', \tau) = -\log f_X(X', \tau), \tag{4}$$
$$a_c(X) = \log p_c(X), \tag{5}$$

which leads to the following expression for the overall cost under a particular cluster assignment:

$$-\log(\mathrm{L}) = \sum_{k=1}^{K} \left( a_b(X_{k,1}) + a_d(X_{k,n}) + \sum_{i=2}^{n_k} a_t(X_{ik,i-1}, X_{k,i}, T_{k,i} - T_{k,i-1}) + \sum_{i=1}^{n_k} a_c(X_{k,i}) \right). \tag{6}$$

The Markov structure of transitions, as well as this representation as additive costs, permit a natural representation as a problem defined on a directed graph. If we construct a directed graph with observations as vertices and possible transitions as arcs, then every possible path in the graph (from any vertex to any other reachable vertex) corresponds to one potential MRP cluster (Figure 1). A set of $K$ paths corresponds to a set of $K$ MRP clusters. To reflect the assumption that each observation is generated by no more than one MRP, we require that a vertex can be a member of no more than one path in such a set. Vertices not included in any of the paths correspond to noise observations.

Given our restriction that a vertex can be included in no more than one path, the problem of finding a mutually compatible set of MRP clusterings is equivalent to solving a particular kind of *network flow* problem (Bang-Jensen and Gutin, 2007, Chapter 3). In our case, the concept of a flow will be used to pick out a set of arcs in the graph corresponding to a possible clustering, by

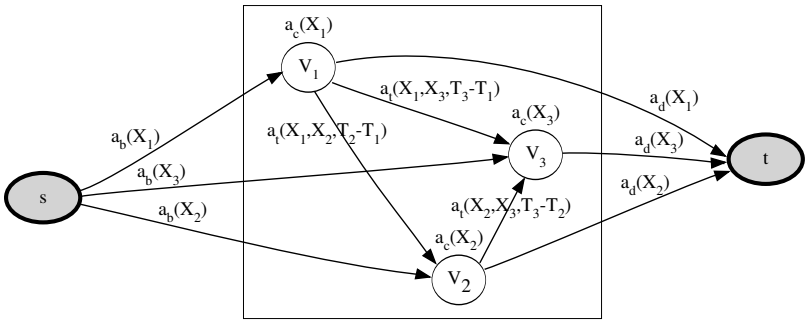



Figure 2: Constructing the weighted flow network for a set of three observations.

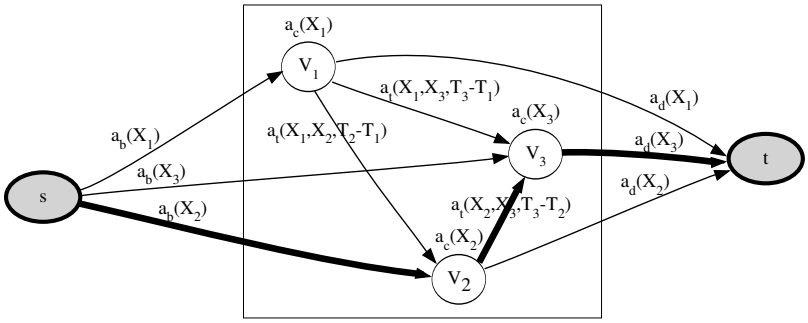



Figure 3: The network of Figure 2, with a single-path flow indicated (*s-2-3-t*).

associating each arc with a value 1 or 0 indicating whether the arc is included in the clustering. Therefore, in addition to the requirement that the flow is integer-valued, all arcs will be defined to have unit capacity: $l_{ij} = 0, u_{ij} = 1$ for all $A_{ij}$. To reflect our assumption that each observation can be included in only one cluster, we will also specify unit capacities for all vertices.

It remains to specify how we can associate the costs (2)–(5) with the network such that we can solve for the minimum-cost solution to (6). Transition costs will be associated with arcs, and clutter costs with vertices, but in order to include birth and death costs we must modify the network by adding a single "source" vertex with an outward arc to all other vertices, and a single "sink" vertex with an inward arc from all other vertices, and by requiring that no other vertices act as sources or sinks (i.e., in a feasible flow, their inward and outward flows must balance). We then associate birth costs with arcs from the source and death costs with arcs to the sink. This means that all feasible flows in our network will be composed of paths which consist of one single birth cost, plus a sequence of clutter and transition costs, and a single death cost. The source and sink have infinite capacity, allowing for solutions with unbounded $K$.

Putting these considerations together, constructing the directed graph proceeds as follows:

- A unit-capacity vertex $V_i$ is created corresponding to each observation $(X_i, T_i)$. The clutter noise cost $a_c(X_i)$ is associated with this vertex.

- A unit-capacity arc $A_{ij}$ is created corresponding to each possible transition between two observations such that $T_i < T_j$. The transition cost $a_t(X_i, X_j, T_j - T_i)$ is associated with this arc.

- A "source" vertex $s$ is added, with one arc $A_{si}$ leading from $s$ to each of the observation vertices. The birth cost $a_b(X_i)$ is associated with each arc $A_{si}$.

- A "sink" vertex $t$ is added, with one arc $A_{it}$ leading from each of the observation vertices to $t$. The death cost $a_d(X_i)$ is associated with each arc $A_{it}$.

The temporal ordering of observations means that the graph will contain no cycles.

An illustration of the network constructed for a set of three observations is given in Figure 2. It is clear that any path from the source $s$ to a sink $t$ (we call this an $(s,t)$-path) visits a sequence of vertices representing a temporal sequence of observations. In the case given in Figure 2, seven different $(s,t)$-paths are possible, and various combinations of these can form a feasible flow. For example the flow along the single path *s-2-3-t* highlighted in Figure 3 represents the possibility that the observations $X_2$ and $X_3$ were generated by a single MRP while $X_1$ is clutter: the costs associated with flow along that path (the *path flow*) are related to the birth of *2*, the transition from *2* to *3*, and the death of *3*, plus the clutter noise costs. The cost associated with any single-path flow corresponds to one of the $K$ top-level summands in Equation (6). Since in our case each (s,t)-path carries one unit of flow, the *value* of each feasible flow is the number of paths it contains, and corresponds to the number of MRP processes inferred in the data. The total *cost* of each feasible flow is the sum of the path costs contained, and corresponds to the sum calculated in Equation (6).

### 3.2 Inference

The minimum cost flow in a network constructed according to our scheme corresponds to the clustering with maximum likelihood ratio. So to perform inference we can use existing algorithms that solve minimum-cost flow problems. The *value* of the minimum-cost flow, which gives the number of MRP sources inferred, may be any integer between 0 and $N$.

*Full inference:* We use the Edmonds-Karp algorithm (Bang-Jensen and Gutin, 2007, Chapter 3), which iteratively searches for single paths in a *residual network* representation and does not get trapped in local optima. The Edmonds-Karp algorithm is often used to find maximum-value flow but can be used to optimise cost in our case of binary capacities.

The asymptotic time complexity of the Edmonds-Karp search relates to the number of vertices and arcs as $O(|V||A|^2)$. The number of vertices is closely related to the number of observations $N$; since we generate an arc for every possible transition between a pair of observations, $|A|$ may be on the order of $N^2$ in the worst case. Hence we add a constraint in constructing the arcs which is reasonable in many applications: we assert that transitions have an upper limit in the size of the time step, and so we do not create arcs for time separations above some threshold $\tau_{\max}$. The cardinality $|A|$ is then on the order of $NB$ where $B$ is the maximum number of observations within a time window of size $\tau_{\max}$ (and often $B << N$).

*Greedy inference:* If faster search is required at the cost of optimality, greedy search strategies are available. One such strategy is to repeatedly apply a minimum-cost path algorithm to the network, at each iteration taking the resulting path as an identified cluster and removing its vertices from the network before the next iteration. Since the graph is acyclic, finding a minimum-cost path can be performed very efficiently with order $O(|A|+|V|)$ at each iteration (Bang-Jensen and Gutin, 2007, Section 2.3.2); however there is no guarantee of optimality since the overall minimum-cost flow is not guaranteed to be composed of path flows of lowest individual cost. In our experiments

we will compare this greedy search empirically against the optimal search (using the same $\tau_{\max}$ threshold for both).

In the present work we primarily consider offline (batch) inference. However, online inference is possible within the same framework, in which new observations are received incrementally by updating the graph as observations arrive. The Edmonds-Karp search cannot be used on such a dynamic network, except by re-starting the search from scratch upon update. Alternative strategies such as those based on cycle-cancelling can be used to provide an updateable inference (Bang-Jensen and Gutin, 2007, Section 3.10.1). The speed of cycle-cancelling relative to Edmonds-Karp may depend on the nature of the data; we implemented both and found the cycle-cancelling relatively slow.

Thus far we have considered inference using a single set of MMRP model parameters, encoded as the costs in (6). It may be of value to evaluate the same data under different MMRP models, in situations where multiple types of MRP process (having different parameters) may be active. Multiple parametrisations cannot be represented together in a single flow network since they would assign conflicting costs to arcs. To accommodate incompatible costs is equivalent to the "multi-commodity" extension of the minimum-cost flow problem, which is NP-complete (Even et al., 1975). However, if the clutter noise model is held constant between two different MMRP inferences, then the two likelihood ratios calculated by (1) can be divided through to give a likelihood ratio between the two. This allows us to choose between possible MMRP models although not to combine them in a single clustering.

To summarise the MMRP inference described in this section: given a set of observations plus MRP process parameters and noise process parameters, one first represents the data as a flow network, with added source and sink nodes, and with costs representing component likelihoods (Section 3.1). One then applies a minimum-cost flow algorithm to the network, such as Edmonds-Karp (which we use for "full inference") or a suboptimal greedy search. Each $(s,t)$-path in the resulting minimum-cost flow represents a single cluster (a single MRP sequence) in the maximum-likelihood result, while the nodes which receive no flow represent data to be labelled as noise.

## 4. Experiments

We have described a multiple Markov renewal process (MMRP) inference technique which takes an MRP model, an iid clutter noise model and a set of timestamped data points, and finds a maximum-likelihood partition of the data into zero or more MRP sequences plus clutter noise. In the following, we will first illustrate its properties with a synthetic experiment designed to explore robustness (Section 4.2). We then apply MMRP inference in two experiments based on applications to audio tracking tasks: a synthetic experiment based on a well-known test of auditory "streaming" (Section 4.3), and an experiment to track multiple singing birds in an audio mixture (Section 4.4). However, we must first consider how to evaluate algorithm outputs.

### 4.1 Evaluation Measures

To judge the empirical performance of our inference procedure, we must determine whether it can correctly separate signal from noise, and whether it can correctly separate each individual MRP sequence into its own stream. MMRP inference can be considered as a clustering task and could be evaluated accordingly. However, the noise cluster is qualitatively different from the MRP clusters,

and the transitions within MRP sequences are the latent features of primary interest, so we will focus our evaluation measures on signal/noise separation and transitions.

In the following our statistics will be based on the standard F-measure (Witten and Frank, 2005, Chapter 5), which summarises precision and recall as follows:

$$\begin{aligned} F &= 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \\ &= \frac{2t_+}{(2t_+ + f_- + f_+)} \end{aligned} \tag{7}$$

where $t_+$ is the number of true positive detections, $f_+$ the number of false positive detections (noise data labelled as signal), and $f_-$ the number of false negative detections (signal data labelled as noise).

However, the task for which our MMRP inference is designed is not an ordinary classification task: the signal/noise label for each ground-truth datum can be treated as a class label to be inferred, but the individual signal streams to be recovered do not have labels. To quantify performance we use the F-measure in two ways. The first (which we denote $F_{\text{SN}}$) evaluates the signal/noise classification performance without considering the clustering. The second (which we denote $F_{\text{sigtrans}}$) evaluates the performance at recovering the *pairwise transitions* that are found in the ground-truth signals, that is, the arcs in the true dependency graph underlying the data. In order to make the two measures relatively independent, we measure $F_{\text{sigtrans}}$ only on event pairs that have been correctly classified as signal, since otherwise false-positive noise events could have a strong influence on both (see Figure 4). Thus, in the following we use $F_{\text{SN}}$ to measure signal/noise separation and $F_{\text{sigtrans}}$ to determine whether inference is correctly recovering separate streams.

### 4.2 Synthetic Experiment I: MMRP Generated Data

We designed a synthetic experiment to generate data under the MMRP model described in previous sections, with user-specified parameters including birth intensity, death probability, and clutter noise intensity. The test was conducted with state $X$ defined on a discrete alphabet and continuous time $T$, and the transition network among states and times was algorithmically generated as follows: for each state, a random subset of possible next states was selected, with the number of out-arcs dependent on the user-specified sparsity of the transition model. The weights of the out-arcs were sampled as a multinomial distribution (sampled from a symmetric Dirichlet distribution with $\alpha = 1$). Each out-arc was also associated with a density on the size of the time gap to the next event, taking a log-normal distribution with a mean randomly sampled from a log-normal parent distribution. To create an observation set, a set of birth events and clutter events were sampled independently from their Poisson distributions, and then each birth event was used as the starting point to sample a single $\{(X,T)\}$ sequence using the death probability and the transition network. The intensities for the birth process and the noise process were uniform across the alphabet of states, and so in the following we parametrise them by their intensity along the time axis only. Similarly, in the present experiment we held death probability as uniform across state. Observation sets and noise events were sampled within a time window of fixed duration. We used a signal-to-noise ratio (SNR) parameter to control the intensity of noise observations ($\lambda_c$) in relation to that of signal observations:

$$\lambda_c = \frac{\lambda_b}{p_d} \cdot \text{SNR}.$$

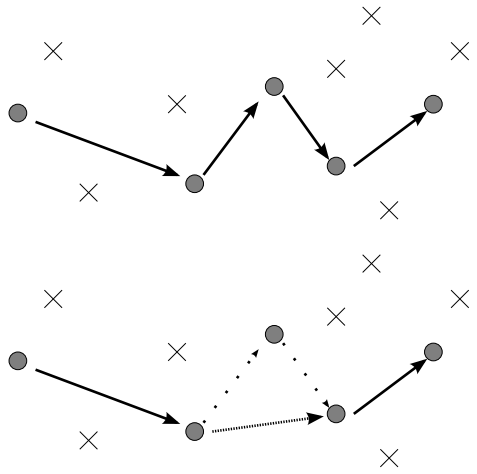


Figure 4: Illustration of errors reflected in $F_{\text{sigtrans}}$. The upper diagram shows a hypothetical ground-truth transition through a sequence of five observations (circles) accompanied by clutter noise (crosses). The lower diagram shows what would happen if inference missed one of those observations out of the chain, resulting in one false-positive (dashed arrow) for a transition that does not exist in the ground-truth. If the "skipped" observation is labelled as noise then the two false-negative arcs (dotted arrows) would not be considered in $F_{\text{sigtrans}}$ (its omission would already be represented in the $F_{\text{SN}}$ statistic): considering the false-positive and the two true-positives and applying (7), the $F_{\text{sigtrans}}$ value then is $\frac{2}{3}$. If the "skipped" observation is labelled as signal then the false-negative arcs are also considered and $F_{\text{sigtrans}}$ is $\frac{2}{5}$.

The factor of $p_d$ appears as well as the birth intensity ($\lambda_b$) because the SNR relates to the count of all signal observations (not just births), and for a fixed death probability we have a geometric distribution over the number of detections per birth with expected value $1/p_d$.

To evaluate performance of our inference applied to such data, we repeatedly generated observation sets as described, and ran both the greedy and full inference algorithms on the data. Unless otherwise stated, for all synthesis runs we used the following parameters: alphabet size 10, SNR 0 dB, birth intensity 0.2 per second, death probability 0.1, observation duration 40 seconds. (We also ran tests with alphabet size 100, obtaining very similar results, and so we have not included those.) In each case a transition network was generated with a sparsity of 50%, and the parent distribution for the transition time densities was a log-normal centred on 1 second with a standard deviation of 1; distributions for each transition arc were log-normal with mean sampled from that parent distribution and a standard deviation of 0.1.

The chosen setting for death probability implies an expected chain length of 10 emissions for a single MRP source. Together with the the birth intensity and SNR this implies that a typical generated observation set would consist of 160 observations, half being signal and half noise. Empirically, each of our observation sets had a mean polyphony (the number of simultaneously active sources) varying from around 0.1 to 4.5, with substantial variation in the polyphony during the course of each (generally 0–10).

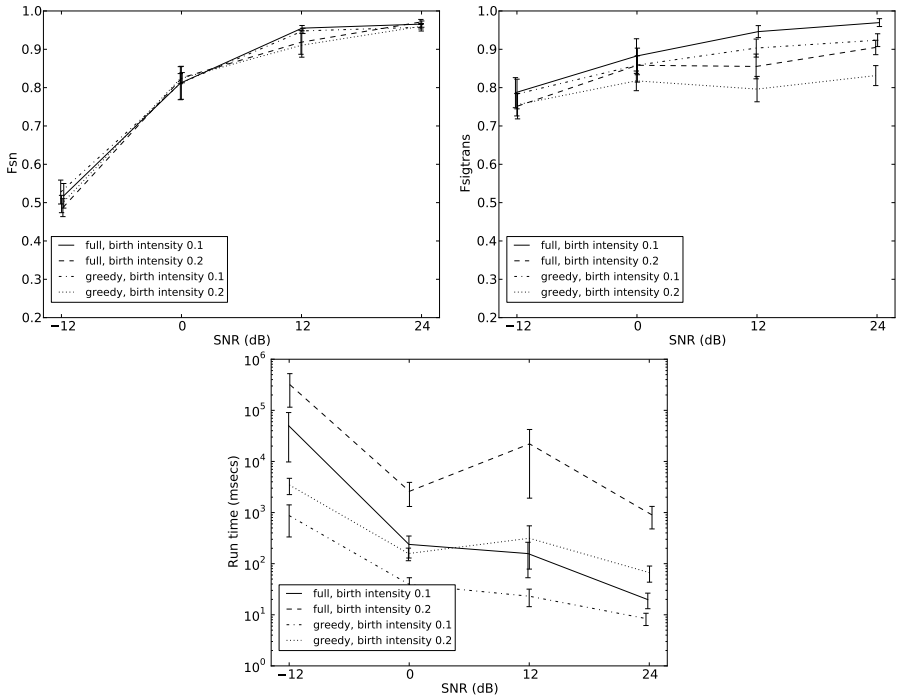



Figure 5: Performance of the full and greedy inference algorithms with varying SNR. Plots show the F-measure for signal/noise separation ($F_{\text{SN}}$, upper left) and signal transitions ($F_{\text{sigtrans}}$, upper right), as well as the measured runtime (lower). Means and confidence intervals are shown, taken over 20 independent runs. States are defined on an alphabet of size 10, and transition sparsity set at 10% or 50% (alternating runs). In this plot, we also compare birth intensities ($\lambda_b$) of 0.1 and 0.2.

Results were evaluated using the $F_{\text{SN}}$ and $F_{\text{sigtrans}}$ measures described in Section 4.1. For our first test, the algorithms were supplied with the true model parameters $p_b$, $f_X$, $p_d$, $p_c$ to calculate (6). Figure 5 shows how performance varies with SNR. In this synthetic experiment, the separation of signal and noise (measured by $F_{\text{SN}}$) is strong at high SNRs and falls off a little as the SNR approaches zero. With very adverse SNR (-12 dB) performance drops off noticeably. This is the case for both the full and greedy algorithms. The $F_{\text{sigtrans}}$ measure shows a milder decline with SNR, but also notable differences between the full and greedy inference, with a consistent benefit in accurate recovery of transitions if the full inference is used. We also show the measured runtime in Figure 5: the increased accuracy of full inference in recovering signal transitions comes at a cost of increased runtime, especially under adverse SNRs (because of the larger number of noise events generated).

In order to study the sensitivity of inference to misspecified or unknown parameters, we also ran the same test but with systematically misspecified parameters for inference. This is important not only because we seek a robust algorithm, but also because parameters such as the birth density and death probability together imply approximate expectations about the level of polyphony in the signal. Since one advantage of our approach is that it infers an arbitrary number of signal sequences

in the data, we are interested to determine whether the correctness of these parameters is crucial for successful inference.

Results are shown in Figure 6. We see that both algorithms (greedy and full) are robust to poor estimation of the birth density, death probability and SNR. The advantage of full over greedy inference is maintained at around five percentage points in $F_{\text{sigtrans}}$ through most of these varied conditions.

We also tested misspecification of transition probabilities. In order to create a controllable amount of misspecification we implemented a stochastic degradation of the transition density information: given a degradation parameter $d \in [0,1]$, for every state in the transition table with probability $d$ we resampled the set of out-arc weights; and then for each out-arc separately, with probability $d$ we resampled the mean of its log-normal density over time. This gave a stochastic corruption of the transition probability which could range from moderate to very strong. Results (Figure 6) show that misspecification of the transition probabilities exhibits a strong effect compared against the other variations: the algorithms are relatively robust up to around 10% degradation, but $F_{\text{sigtrans}}$ in particular falls dramatically when the transition probabilities become badly corrupted. This reflects the fact that the transition probabilities encode the key structural distinction between signal and noise, *and* the key information that one could use to disambiguate two co-occurring signal streams.

We also investigated how inference may degrade when conditions fail to match some of the assumptions of the model: in many applications there may be missed detections, or noise may not be truly independent but exhibit correlations with the signal. Figure 7 shows the performance of inference as these issues are progressively introduced into the data. Missed detections were simulated by omitting observations at random; noise correlations were simulated by selecting a controllable fraction of the noise observations, and modifying those noise observations to have the same state and very similar time position as a randomly-selected signal datum. The algorithms appear moderately robust to such problems: $F_{\text{SN}}$ progressively deteriorates as the proportion of issues increases, but $F_{\text{sigtrans}}$ exhibits notable strong declines down toward chance performance with strong degradation. However, the algorithms (both greedy and full) are robust to moderate violations of the assumptions.

However, we also noticed that correlated noise led to a significant increase in algorithm runtime. This is plotted in Figure 8, showing that correlated noise beyond 25% can lead to run-times which are orders of magnitude longer, even though the data under consideration has the same number of observations and the same ratio of signal and noise observations. This occurs in the full algorithm, and also in the greedy algorithm though with less severity. We propose that the reason for this is that when the flow network includes many search paths which are extremely similar—for example differing only in the choice of a particular signal datum or a competing noise datum, both at the same location and thus with the same likelihood—then this can create a combinatorial explosion of paths that must be explicitly searched. Standard network search algorithms use branch-and-bound-type optimisation to avoid explicit recursion into many of the candidate paths (Papadimitriou and Steiglitz, 2000). This optimisation ignores search paths which have no possibility of improving on a locally cached result, and so speeds up search while still finding the global optimum. The effect of this optimisation is weakened when many paths have very similar costs, and search time increases in practice even though the formal size of the problem is no different.

To summarise the observations made in this experiment, we find that MMRP inference is generally quite robust to variations in conditions and model parameters. The greedy algorithm achieves performance close to that of the full algorithm in most cases, although the full algorithm consistently

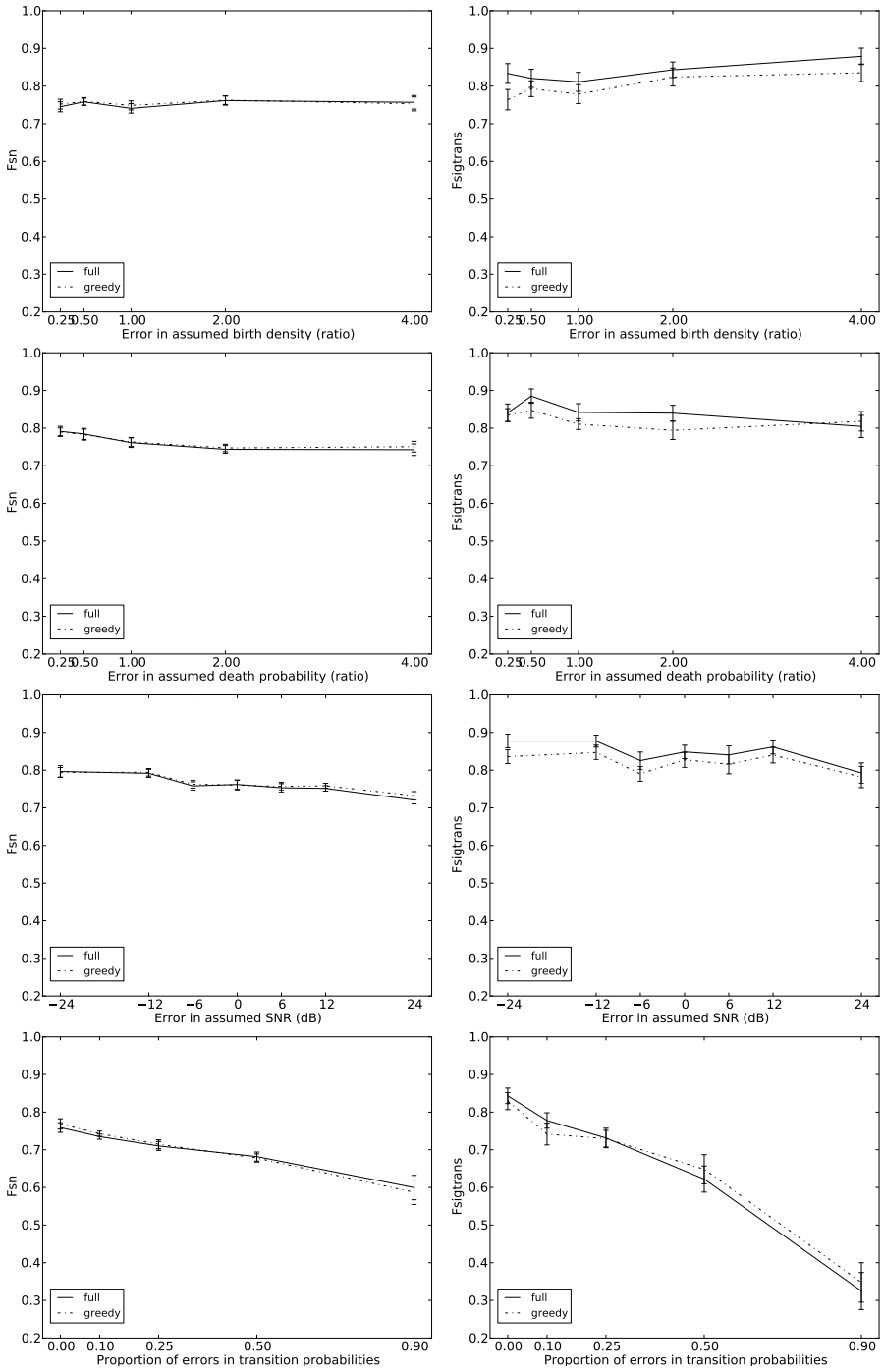



Figure 6: Sensitivity of inference to misspecified parameters. Plots are as in Figure 5 but showing how performance varies with mismatch between the true and specified parameters for the birth density, death probability, SNR, and transition density. SNR is fixed at 12 dB for all plots, except in the SNR plot for which we average over runs with true SNR $\in 0, 6, 12$ dB to confirm that SNR sensitivity does not vary strongly with SNR.

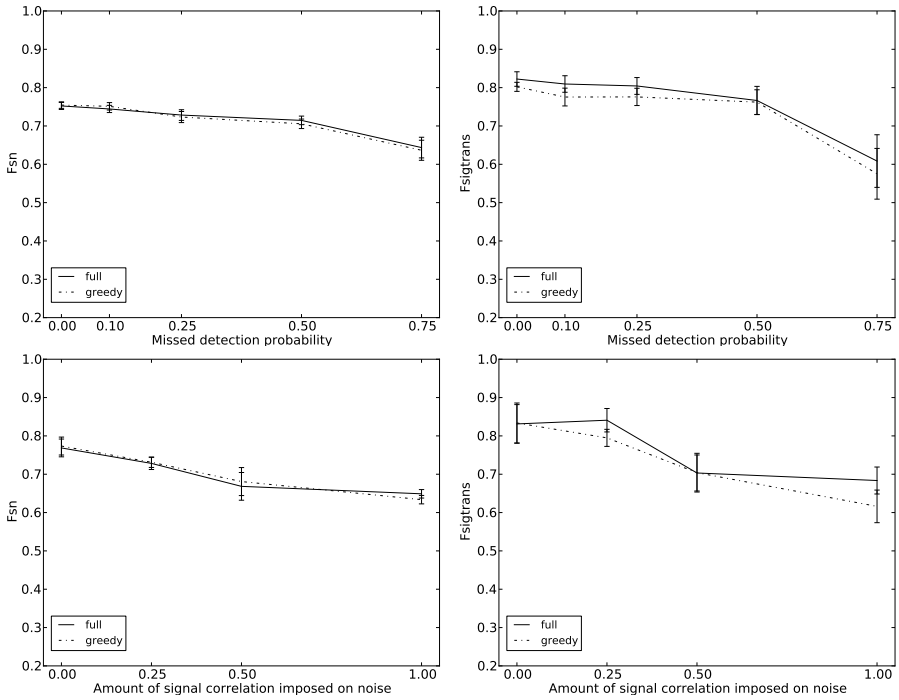



Figure 7: Sensitivity of inference to missed data and correlated noise. Plots are as in Figure 6 but showing how performance varies when some detections are missed, and when noise is not independent but correlated with signal.

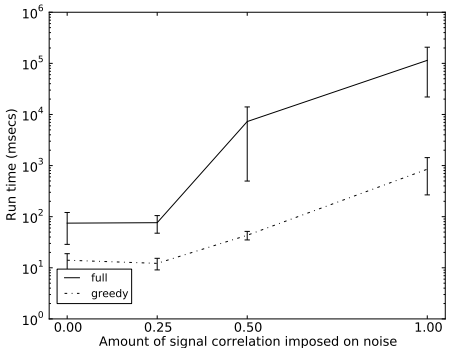



Figure 8: Algorithm run-time for the correlated-noise test of Figure 7.

achieves stronger $F_{\text{sigtrans}}$ in all but some strongly adverse conditions. This shows empirically that the greedy algorithm has a tendency to find local optima, and the results suggest these local optima reflect not so much issues in signal/noise discrimination but "crossed wires" in MRP sequences. The most critical parameter for successful MMRP inference appears to be the transition probability structure rather than assumptions about birth/death probabilities, which accords with our intuition that the Markov structure of the sequences is the source of the discriminative power. As well as the

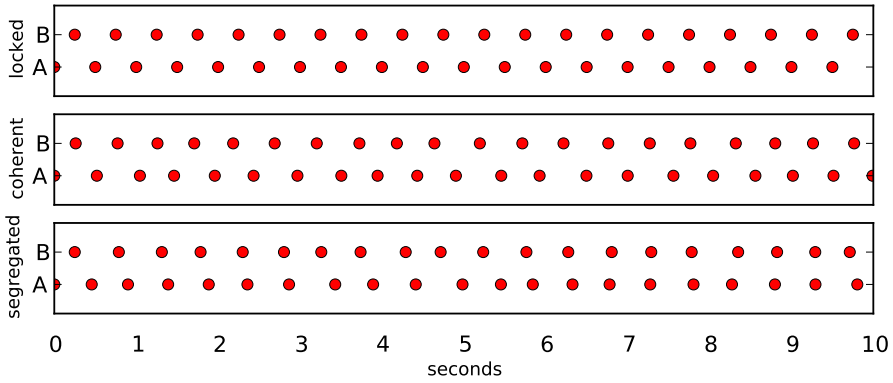



Figure 9: Examples of sequences generated by strict locked ABABAB repetition (top), and by similar generators but with time offsets affected by process noise reflecting either coherent (ABABAB, middle) or segregated (A_A_A_ and _B_B_B, bottom) dependency structure.

transition structure, another important consideration is signal/noise correlation, which in the present experiment can lead to impaired $F_{\text{sigtrans}}$ results as well as notably increased computation time.

### 4.3 Synthetic Experiment II: Auditory Streaming

To illustrate the relevance of our algorithm to the multi-source tracking required in tasks such as machine listening (or computational auditory scene analysis), we next consider a synthetic experiment inspired by the classic "audio streaming" experiments used to explore human auditory grouping of sound sequences (Winkler et al., 2012). In this context the MRP model might be taken to represent not necessarily a model of how event sequences were generated, but a compact model of expectations about event sequences that can be used for computational tasks such as auditory streaming.

A strictly alternating sequence of the form ABABAB…, where A and B are different tones (Figure 9, top row), can be interpreted either as a single alternating sequence (the "coherent" interpretation) or as a simultaneous but out-of-phase pair of constant sequences (the "segregated" interpretation). Various factors can lead an observer to prefer one interpretation or the other; here we focus on the case where drift in the timing of the events makes one or the other model more likely (Cusack and Roberts, 2000, Experiment 2). If the sequences drift such that the phase of the As and Bs remain in constant relationship (Figure 9, second row), this is consistent with a "coherent" alternating generator, though may by chance be generated by a "segregated" pair of generators. If the sequences drift such that the phase relationship is not maintained (third row), then this is inconsistent with the "coherent" model but consistent with the "segregated" model. We can generate data with these properties and observe how the MMRP inference behaves under the assumptions of each model.

For our synthetic experiment we defined two separate MRP transition models (one "coherent" and one "segregated") to emit values in a one-dimensional state space $\mathcal{X} \in \mathbb{R}$. Each model was

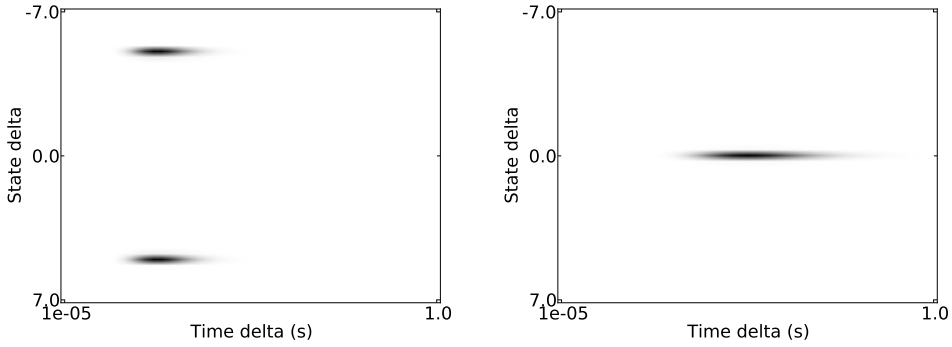



Figure 10: MRP transition probability densities for the two synthetic models: coherent (left) and segregated (right). The coherent model uses two Gaussians centred on 0.25 sec and $\pm 5$ in state, while the segregated model uses one Gaussian centred on 0.5 sec and 0 in state. For each Gaussian, the standard deviation is 0.1 in state and 0.2 in log-time-delta.

specified by a Gaussian mixture probability distribution defined on state-delta and log-time-delta:

$$P(\tau_{n+1} \leq t, X_{n+1} = j \mid X_n = i) = f(X_{n+1} - X_n, \log \tau_{n+1}) .$$

Figure 10 illustrates the transition models. Time differences here are modelled as log-Gaussian to reflect a simple yet perceptually plausible model for lower-bounded time intervals. The variance of the Gaussian components leads to process noise, and the two models tend to output different sequences in general. We also define a "locked" model for generation only, which generates a strict ABABAB sequence with no process noise. Its emissions could in principle be explained by either of the two other models.

These models served two roles in our experiment, to synthesise data and to analyse it. For synthesis, we generated four simultaneous sequences each with a random offset in state space, and we also added iid Poisson clutter noise in the same region of state space, whose intensity is held constant within each run to create a given SNR. In the case of the segregated model, each generator was a pair of such models, independent except for the initial phase and offset, generating As and Bs as was done in Figure 10. In this experiment we did not use probabilistic births or deaths during synthesis, instead generating a fixed polyphony lasting the whole of the excerpt. For MMRP inference we used fixed parameters derived from the SNR value and an arbitrary death probability of 0.033. The following relationships show how to derive the birth and clutter likelihood parameters from the SNR value expressed as a ratio:

$$p_b = \frac{\mathrm{SNR} \cdot p_d}{1 + \mathrm{SNR}} , \qquad p_c = \frac{1}{1 + \mathrm{SNR}} . \tag{8}$$

The factor of $p_d$ enters into the calculation of $p_b$ for the reasons described in Section 4.2.

The first column of Figure 11 shows the results of generating data under the locked, coherent and segregated models, with two generated sequences present in each case. The second column

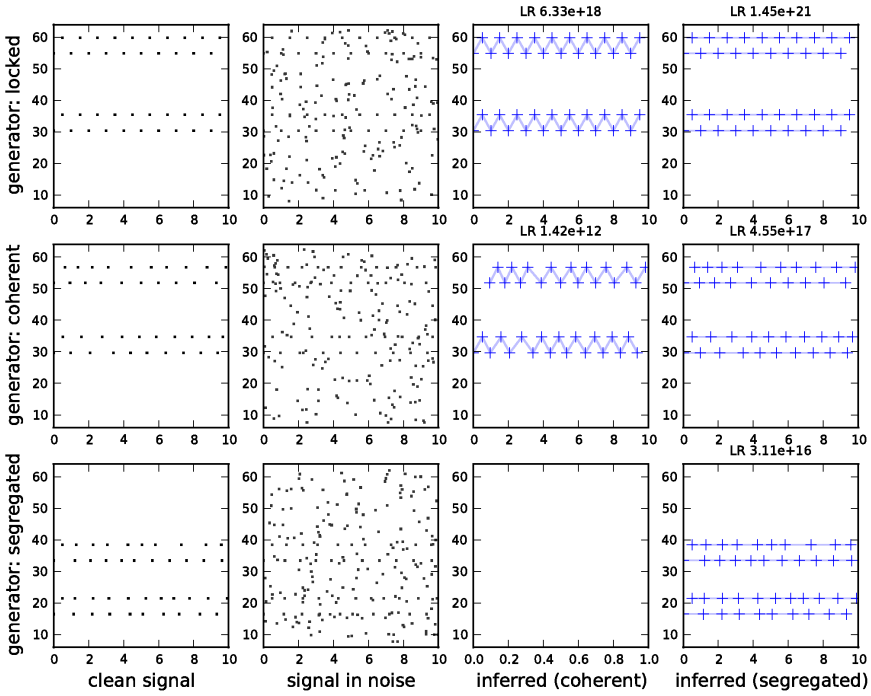



Figure 11: Results of generating observations under the locked, coherent or segregated model (in each row), and then analysing them using the coherent model or the segregated model (final two columns). Note that we have selected an example with clear pitch separation between streams, for visual clarity: in general, and in our tests with four streams, sequences often overlap in pitch and are not so obviously separable.

shows the sequences with added clutter noise at an SNR of -12 dB. The final two columns show the maximum-likelihood signal sequences inferred under the coherent and the segregated model. The MMRP inference typically extracts clear traces corresponding to the ground-truth signals, even in strongly adverse SNR. It is visually evident in the first column that the generated sequences in the middle row have some drift in their rate, but stay in order, while the As and Bs in the bottom row drift relative to each other and do not maintain order. This leads to unlikely emission sequences as judged by the coherent model, and so the coherent model finds the maximum-likelihood solution to be that with no sequences (the blank plot in the figure). Inference using the segregated model extracts traces in all three cases, since the phase-locked drift of the coherent model is not unlikely under the segregated model.

To evaluate MMRP inference in this case, we ran this process multiple times, varying the SNR level and whether the true SNR was known to the algorithm. When not known, the SNR estimate was arbitrarily held fixed at 0 dB. For each setting we conducted 20 runs and recorded the $F_{\mathrm{SN}}$ and $F_{\mathrm{sigtrans}}$ statistics. Figure 12 illustrates the results, showing broad consistency with the previous experiment. Recovery performance is very strong in all but the most adverse conditions, in most cases

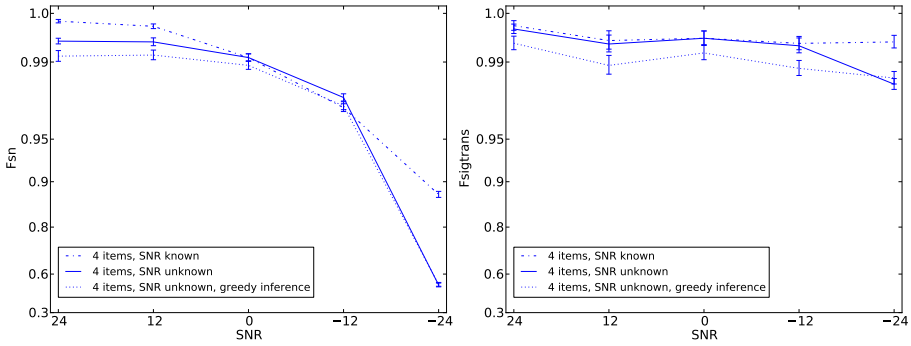



Figure 12: F-measure for signal/noise separation ($F_{\text{SN}}$) and transitions ($F_{\text{sigtrans}}$). The ground truth in each case is a combination of four ABABAB streams, generated via the coherent or segregated cases (20 runs of each type). Means and standard errors are shown; the vertical axis is reverse-log-scaled so that the results very near 1.0 can be distinguished.

being well above 0.95. For these particular scenarios, signal/noise discrimination is impaired under the strongest condition tested (SNR -24 dB), but under other conditions the recovery is good, and $F_{\text{sigtrans}}$ remains strong in all runs. As in the previous experiment, full inference shows a consistent advantage over the greedy inference, though this tails off at -24 dB SNR. In this test, knowledge of the true SNR gives a further boost in the performance of MMRP inference.

### 4.4 Birdsong Audio Experiment

Many natural sound sources produce signals with structured patterns of emissions and silence, for example birdsong or footsteps. As in the previous experiment inspired by auditory streaming, if we model these natural sound sources with an MRP then our inference procedure should be able to separate multiple simultaneous "streams" of emissions. In the following experiment we studied the ability of our inference to perform this separation in data derived from audio signals containing multiple instances of a species of bird common in many European countries, the Common Chiffchaff (Salomon and Hemim, 1992). Chiffchaff song consists of sequences of typical length 8–20 "syllables". Each syllable is a pitched note consisting of a downward chirp to a briefly-held tone in the region of 5–8 kHz. Syllables are separated by around 0.2–0.3 seconds. The exact note sequence has not to our knowledge been studied in detail; it appears to exhibit only short-range dependency, and is thus amenable to analysis under Markovian assumptions.

#### 4.4.1 Data Preparation

To aid reproducibility, we used recordings from the Xeno Canto database of publicly-available bird recordings.[1] We located 25 recordings of song of the Chiffchaff (species *Phylloscopus collybita*) recorded in Europe (excluding any recordings marked as having "deviant" song or uncertain species identity; also excluding *calls* which are different from *song* in sound and function). The recordings used are listed in Table 1. We converted the recordings to 44.1 kHz mono wave files, high-pass filtered them at 2 kHz, and normalised the amplitude of each file. File durations varied from 8.5

1. Available at `http://www.xeno-canto.org/europe`.

| ID | Country | ID | Country |
|---|---|---|---|
| XC103404 | pl | XC48263 | no |
| XC25760 | dn | XC48383 | de |
| XC26762 | se | XC54052 | it |
| XC28027 | de | XC55168 | fr |
| XC29706 | se | XC56298 | de |
| XC31881 | nl | XC56410 | ru |
| XC32011 | nl | XC57168 | fr |
| XC32094 | no | XC65140 | es |
| XC35097 | es | XC77394 | dk |
| XC35974 | cz | XC77442 | se |
| XC36603 | cz | XC97737 | uk |
| XC36902 | nl | XC99469 | pl |
| XC46524 | nl | | |

Table 1: Chiffchaff audio samples used in our data set, giving the Xeno Canto ID and the country code. Each recording can be accessed via a URL such as `http://www.xeno-canto.org/XC103404`, and the data set is also archived at `http://archive.org/details/chiffchaff25`.

seconds to many minutes, so to create a set of independent audio samples which could be mixed together to create mixtures with overlapping bouts of song, audio files were each trimmed automatically to their highest-amplitude 8.5-second segment. Source code for these preprocessing steps are published along with the full code.[2]

Each audio file was analysed separately to create training data; during testing, audio files were digitally mixed in groups of one to five files.

In order to convert an audio file into a sequence of events amenable to MMRP inference, we used spectro-temporal cross-correlation to detect individual syllables of song, as used by Osiejuk (2000). We designed a spectrotemporal template using a Gaussian mixture (GM) to represent the main characteristics of a single Chiffchaff syllable, a downward chirp to a briefly-held note (Figure 13). The GM was modelled on a Chiffchaff recording from Xeno Canto which was not included in our main data set (ID number XC48101). Then to analyse an audio file we converted the file into a spectrogram representation (512 samples per frame, 50% overlap between frames, Hann window), and converted the GM to a sampled grid template with the same time-frequency granularity as the spectrogram, before sliding the grid template along the time axis and along the frequency axis (between 3–8 kHz), evaluating the correlation between the template and spectrogram at each location. Correlation values were treated as detections if they were local peaks with value greater than a threshold correlation of 0.8.

Such cross-correlation detection applied to an audio file produces a set of observations, each having a time and frequency offset and a correlation strength (Figure 14). It typically contains one detection for every Chiffchaff syllable, with occasional doubled detections and spurious noise detections. When applied to mixtures of audio, this produces data appropriate for MMRP inference.

2. Available at `https://code.soundsoftware.ac.uk/projects/markovrenewal`.

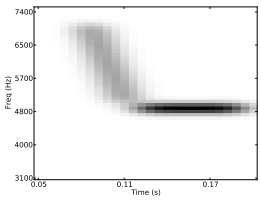



Figure 13: Template used for spectro-temporal cross-correlation detection. The downward and horizontal bars have equal total weight; the latter appears darker because shorter. The template is a manually-constructed Gaussian mixture model having 40 components. It is then used for signal pre-processing both during training and testing.

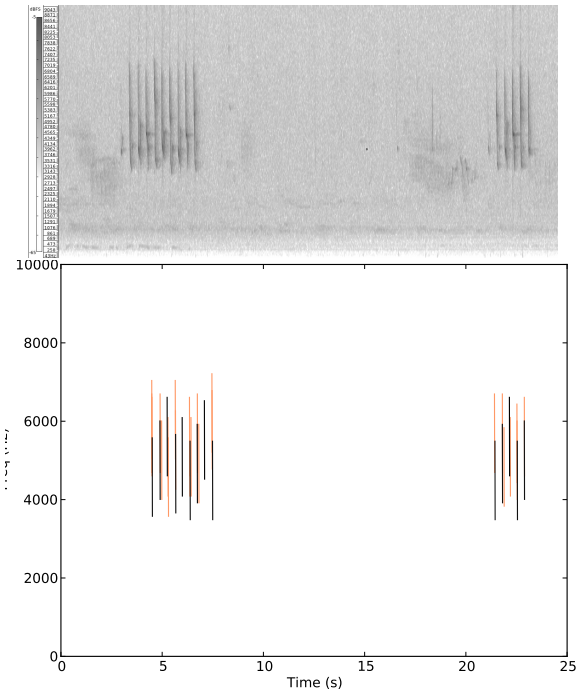



Figure 14: Example of cross-correlation detection: excerpt of spectrogram shown (top), and the corresponding detections (bottom). In the lower image, bold lines represent detections treated as "signal" in the filtering used for training, while the fainter lines represent detections used to train the noise model.

Note that the noise detections often have relatively strong signal correlations, as seen in Figure 14. From our first experiment (Section 4.2) we expect this to have an effect primarily on runtime, though it may also be an issue for performance. We will consider this in light of the results.

In order to derive a Gaussian mixture model (GMM) transition probability model from monophonic Chiffchaff training data, for each audio file in a training set we filtered the observations automatically to keep only the single strongest detection within any 0.2 second window. This time

limit corresponds to the lower limit on the rate of song syllables; such filtering is only appropriate for monophonic training sequences and was not applied to the audio mixtures used for testing. The filtered sequences were then used to train a 10-component GMM with full covariance, defined on the vector space having the following four dimensions:

- log(frequency) of syllable one
- log(frequency) of syllable two
- log(magnitude ratio between syllables)
- log(time separation between syllables)

This GMM then served as the transition probability distribution to be used in inference. We also trained a separate GMM to create a noise model, taking the set of observations that had been discarded in the above filtering step and training a 10-component GMM with full covariance to fit an iid distribution to the one-dimensional log(frequency) data for the noise observations.

#### 4.4.2 INFERENCE FROM AUDIO MIXTURES

In order to test whether the MMRP approach could recover syllable sequences from audio mixtures, we performed an experiment using five-fold cross-validation. For each fold we used 20 audio files for training, and then with the remaining five audio files we created audio mixtures of up to five signals, testing recovery in each case. For each mixture file, we applied spectro-temporal cross-correlation as described above, then performed both full and greedy inference using the empirically-derived signal and noise GMMs to provide densities/intensities for transition and clutter. We used fixed probabilities for $p_d$, inferred from the empirical average sequence length in the training data, and $p_b$, inferred using (8) with a default SNR estimate of 0 dB.

To provide a low-complexity baseline showing the recovery quality using only the marginal properties of the signal and noise, we also created a simple baseline system which treated both signal and noise as iid one-dimensional log(frequency) data, using maximum likelihood to label each observation as either signal or noise. The baseline system then clustered together observations that were identified as signal and were separated by less than 0.7 seconds (a duration chosen to reflect the 0.2–0.3 sec gap sizes in Chiffchaff song, with tolerance for occasional missed detections).

We tested each of these approaches using mixtures of one, two, three, four or five of the test recordings. As in the previous experiment, we measured the $F_{\mathrm{SN}}$ statistic to evaluate signal/noise separation, and the $F_{\mathrm{sigtrans}}$ statistic to evaluate the performance at recovering separate sequences.

Results are shown in Figure 15. Broad outcomes are similar to those of the previous experiments. Signal/noise discrimination is very similar between full and greedy inference, and remains steady as the polyphony increases. Again, though, the full inference shows a general advantage over greedy inference in the correct recovery of transitions. This pattern is consistent across all the polyphony levels tested, except the case of just one bird, in which there is no occasion for the greedy method to make mistakes by crossing one bird's track with another, so it achieves the same performance as full inference.

We also note that all the MMRP inference runs exhibit a significant and very strong improvement over the baseline, both for $F_{\mathrm{SN}}$ and $F_{\mathrm{sigtrans}}$. This shows that the transition information learnt from the training data is indeed pertinent in this application example.

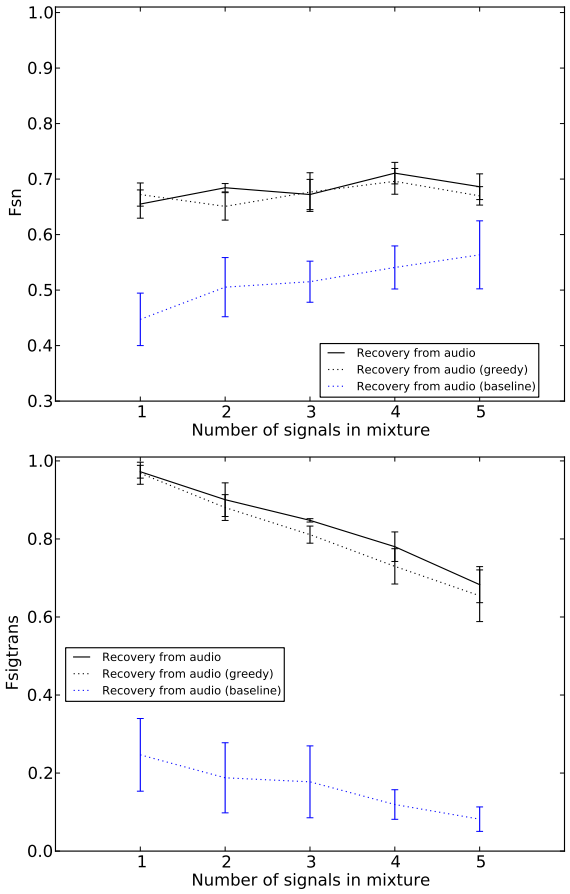



Figure 15: The $F_{\mathrm{SN}}$ and $F_{\mathrm{sigtrans}}$ evaluation measures for the Chiffchaff audio analyses. Means and standard errors are shown taken over the five folds of the cross-validation.

However, in this experiment the levels of $F_{\mathrm{SN}}$ obtained are much lower than in the purely synthetic experiments. A likely reason for this is the front-end we use to detect events in audio: the simple cross-correlation technique may extract slightly different events when applied to a mixture as opposed to the monophonic recordings. Another potential issue is whether the fitted GMMs generalise well from training to test data. In order to explore these factors we ran the same test with full inference, but with some variation on the front-end analysis:

**Ideal recovery:** To simulate ideal-case recovery, instead of using the audio mixture we simply pooled the signal and noise observations that had been derived from the test set's individual mono analysis, then performed MMRP inference as in the audio recovery case.

**Ideal recovery, synthetic noise:** To simulate ideal recovery but with more adverse noise conditions, we proceeded as in the ideal case, but also added extra clutter noise at 0 dB. To do this, we created a copy of every observation in the test set, but assigned it an independent random time position, thus creating noise with the same frequency distribution as the true signal but uncorrelated in time.

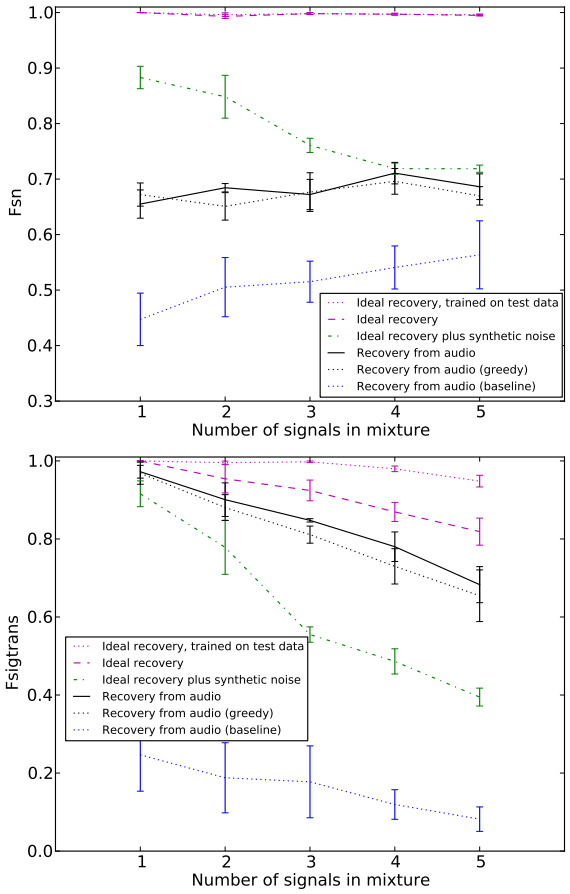



Figure 16: As Figure 15, with ideal-recovery results superimposed.

**Ideal recovery, tested on training set:** To measure an "upper limit" on performance and probe the generalisation capability of the algorithm, we proceeded as in the ideal case, but used GMMs trained on the actual test files to be analysed rather than on the separate training data.

Results, superimposed with the results from the standard detection approach, are shown in Figure 16. Very strong performance is achieved in the noiseless "ideal recovery" cases, achieving results similar to those in the synthetic experiments. The small size of the difference between training on the test data and on the training data (in particular for $F_{\mathrm{SN}}$) indicates that the algorithm can generalise across the data used in our experiment. However, the $F_{\mathrm{sigtrans}}$ measure shows a notable boost when trained on test data, which may reflect some degree of heterogeneity in transition distributions in the recordings. Resolving the similarity of sequences in birds across different geographical locations is of interest to bioacoustics researchers (see, e.g., Mahler and Gil 2009), but at present there are not the large annotated databases that would facilitate such analyses of similarity for a single species.

When synthetic noise is added to the ideal-recovery case, performance is reduced considerably. The $F_{\mathrm{SN}}$ measure approaches that of the more realistic case, while $F_{\mathrm{sigtrans}}$ is even more strongly impaired. This indicates that the detection front-end in the more realistic case is indeed a bottleneck

for performance, but its impact on $F_{\text{sigtrans}}$ is not as severe as on $F_{\text{SN}}$. Note that our synthetic noise is temporally decorrelated from the signal, whereas the noise present in recovery from audio mixtures shows quite strong correlations (Figure 14). Our results indicate that in this experiment the noise correlation is not a major impediment to recovery from audio, since the uncorrelated noise induces consistently worse performance in $F_{\text{sigtrans}}$, and a similar level of performance in $F_{\text{SN}}$ at high polyphony.

Taken together, these results show that the practical task of retrieving detections from audio mixtures has a significant effect on algorithm performance, but that MMRP inference still performs strongly in simultaneously inferring signal/noise discrimination and clustering signals into tracks. We are exploring more sophisticated bird syllable detection to improve on these results (Stowell et al., 2013). As in the synthetic experiments, in the present experiment the full MMRP inference shows a consistent $F_{\text{sigtrans}}$ benefit over the greedy inference, although this must be balanced against the additional runtime cost.

## 5. Conclusions

In this paper we have investigated the problem of segregating timestamped data originating in multiple point processes plus clutter noise. We developed an approach to inferring structure in data produced by a mixture of an unknown number of similar Markov renewal processes (MRPs) plus independent clutter noise. The inference simultaneously distinguishes signal from noise as well as clustering signal observations into separate source streams, by solving a network flow problem isomorphic to the MMRP mixture problem. Our method is general and has very few free parameters.

In experiments we have shown that inference can perform very well even under high noise conditions (as far as $-12$ dB SNR, depending on application). The full optimal MMRP inference incurs a higher complexity than a greedy approach, but generally achieves a more accurate recovery of the event-to-event transitions present in the data. In a synthetic experiment, we explored the robustness of inference, and found that good performance is possible despite misspecification of parameters such as the birth density and noise level. Inaccurate specification of the MRP transition probability structure can impair performance, as can correlated noise, though inference is still robust to mild corruptions of these types. Correlated noise can also incur high run-times because of its effect on the graph search.

To illustrate applications of the technique, we then conducted two experiments related to audio recognition tasks. In an experiment based on the "auditory streaming" paradigm, we showed that MMRP inference can recover polyphonic event streams from noisy observations, applying different MRP generative models to implement different expectations about the streams to be recovered. Then in an experiment on birdsong audio data we showed strong performance, albeit with a dependence on the quality of the underlying representation to recover events from audio data.

The inference in the present work is limited to models without hidden state and with only single-order Markov dependencies. These limitations arise from the combinatorial ambiguity in MMRP mixtures (unlike ordinary Markov models) over which is the immediate predecessor for each observation. Future work will aim to find techniques to broaden the class of models that can be treated in this way.

Reproducible research: Python source code for our implementation and our experiments is freely available online.[2]

## Acknowledgments

DS and MP are supported by EPSRC Leadership Fellowship EP/G007144/1.

This work is licensed under a Creative Commons Attribution 3.0 Unported License.
http://creativecommons.org/licenses/by/3.0/

## References

M. F. Arlitt and C. L. Williamson. Internet web servers: Workload characterization and performance implications. *IEEE/ACM Transactions on Networking*, 5(5):631–645, 1997. doi: 10.1109/90.649565.

J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer Verlag, London, 1st edition, 2007. URL http://www.cs.rhul.ac.uk/books/dbook/.

J. P. Barker, M. P. Cooke, and D. P. W. Ellis. Decoding speech in the presence of other sources. *Speech Communication*, 45(1):5–25, 2005. doi: 10.1016/j.specom.2004.05.002.

O. Bobrowski, R. Meir, and Y. C. Eldar. Bayesian filtering in spiking neural networks: Noise, adaptation, and multisensory integration. *Neural Computation*, 21(5):1277–1320, 2009. doi: 10.1162/neco.2008.01-08-692.

R. Cusack and B. Roberts. Effects of differences in timbre on sequential grouping. *Attention, Perception, & Psychophysics*, 62(5):1112–1120, 2000. doi: 10.3758/BF03212092.

S. Even, A. Itai, and A. Shamir. On the complexity of time table and multi-commodity flow problems. In *16th Annual Symposium on Foundations of Computer Science*, pages 184–193. IEEE, 1975. doi: 10.1109/SFCS.1975.21.

J. R. Glass. A probabilistic framework for segment-based speech recognition. *Computer Speech & Language*, 17(2):137–152, 2003. doi: 10.1016/S0885-2308(03)00006-8.

E. F. Keane, D. A. Ludovici, R. P. Eatough, M. Kramer, A. G. Lyne, M. A. McLaughlin, and B. W. Stappers. Further searches for rotating radio transients in the Parkes Multi-beam Pulsar Survey. *Monthly Notices of the Royal Astronomical Society*, 401(2):1057–1068, 2010. doi: 10.1111/j.1365-2966.2009.15693.x.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *The Journal of Machine Learning Research*, 5:549–573, 2004. URL http://jmlr.csail.mit.edu/papers/v5/koivisto04a.html.

M. Lagrange, L. G. Martins, J. Murdoch, and G. Tzanetakis. Normalized cuts for predominant melodic source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 16 (2):278–290, 2008.

B. Mahler and Diego Gil. *The Evolution of Song in the Phylloscopus Leaf Warblers (Aves: Sylviidae): A Tale of Sexual Selection, Habitat Adaptation, and Morphological Constraints*, volume 40, pages 35–66. 2009. doi: 10.1016/S0065-3454(09)40002-0.

R. P. S. Mahler. *Statistical Multisource-Multitarget Information Fusion*. Artech House, Boston/London, 2007.

G. Mysore, P. Smaragdis, and B. Raj. Non-negative hidden Markov modeling of audio with application to source separation. In *Proceedings of the International Conference on Latent Variable Analysis and Signal Separation (LVA / ICA)*, volume 6365/2010, pages 140–148, St. Malo, France, 2010. doi: 10.1007/978-3-642-15995-4_18.

T. S. Osiejuk. Recognition of individuals by song, using cross-correlation of sonograms of Ortolan buntings emberiza hortulana. *Biological Bulletin of Poznań*, 37(1 Suppl):39–50, 2000.

C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, New Jersey, new edition, 2000.

M. D. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, and M. E. Davies. Sparse representations in audio and music: From coding to source separation. *Proceedings of the IEEE*, 98(6):995–1005, 2010. doi: 10.1109/JPROC.2009.2030345.

M. Salomon and Y. Hemim. Song variation in the Chiffchaffs (*Phylloscopus collybita*) of the Western Pyrenees–the contact zone between the *collybita* and *brehmii* forms. *Ethology*, 92(4):265–282, 1992. doi: 10.1111/j.1439-0310.1992.tb00965.x.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

D. Stowell, S. Muševič, J. Bonada, and M. D. Plumbley. Improved multiple birdsong tracking with distribution derivative method and Markov renewal process clustering. In *Proceedings of the International Conference on Audio and Acoustic Signal Processing (ICASSP)*, 2013. preprint arXiv:1302.3642.

J. Van Gael, Y. W. Teh, and Z. Ghahramani. The infinite factorial hidden Markov model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 21, pages 1697–1704. 2009.

D. L. Wang and G. J. Brown, editors. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. IEEE Press, New York, 2006.

I. Winkler, S. Denham, R. Mill, T.M. Bőhm, and A. Bendixen. Multistability in auditory stream segregation: a predictive coding view. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 367(1591):1001–1012, 2012. doi: 10.1098/rstb.2011.0359.

I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, CA, USA, 2nd edition, 2005.

Journal of Machine Learning Research 14 (2013) 2239-2286 Submitted 9/12; Revised 4/13; Published 8/13

# Gaussian Kullback-Leibler Approximate Inference

**Edward Challis** E.CHALLIS@CS.UCL.AC.UK
**David Barber** D.BARBER@CS.UCL.AC.UK
*Department of Computer Science*
*University College London*
*London, WC1E 6BT, UK*

**Editor:** Manfred Opper

## Abstract

We investigate Gaussian Kullback-Leibler (G-KL) variational approximate inference techniques for Bayesian generalised linear models and various extensions. In particular we make the following novel contributions: sufficient conditions for which the G-KL objective is differentiable and convex are described; constrained parameterisations of Gaussian covariance that make G-KL methods fast and scalable are provided; the lower bound to the normalisation constant provided by G-KL methods is proven to dominate those provided by local lower bounding methods; complexity and model applicability issues of G-KL versus other Gaussian approximate inference methods are discussed. Numerical results comparing G-KL and other deterministic Gaussian approximate inference methods are presented for: robust Gaussian process regression models with either Student-*t* or Laplace likelihoods, large scale Bayesian binary logistic regression models, and Bayesian sparse linear models for sequential experimental design.

**Keywords:** generalised linear models, latent linear models, variational approximate inference, large scale inference, sparse learning, experimental design, active learning, Gaussian processes

## 1. Introduction

For a vector of parameters $\mathbf{w} \in \mathbb{R}^D$, in a large class of probabilistic models we require the inferential quantities

$$p(\mathbf{w}) = \frac{1}{Z}\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_{n=1}^{N} \phi_n(\mathbf{w}), \tag{1}$$

$$Z = \int \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_{n=1}^{N} \phi_n(\mathbf{w}) d\mathbf{w}, \tag{2}$$

where $p(\mathbf{w})$ is a multivariate real valued probability density function, $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a multivariate Gaussian density with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, and $\{\phi_n\}_{n=1}^N$ are positive, real valued, non-Gaussian potential functions.

The range of models that require us to compute these quantities is broad. In the Bayesian setting an important class is Bayesian generalised linear models (GLMs) for which examples include: sparse Bayesian linear models, where the Gaussian term is the likelihood and $\{\phi_n\}_{n=1}^N$ are factors of the sparse prior (Park and Casella, 2008); Gaussian process models, where the Gaussian term is a prior over latent function values and $\{\phi_n\}_{n=1}^N$ are factors of the non-Gaussian likelihood (Vanhatalo et al., 2009); and binary logistic regression, where the Gaussian term is a prior on the parameter

©2013 Edward Challis and David Barber.

vector and $\{\phi_n\}_{n=1}^N$ are logistic sigmoid likelihood functions (Jaakkola and Jordan, 1997). In the context of unsupervised learning examples include: independent components analysis, where the Gaussian term is the conditional density of the signals and $\{\phi_n\}_{n=1}^N$ are factors of the sparse density on the latent sources $\mathbf{w}$ (Girolami, 2001); and binary or categorical factor analysis models where the Gaussian term is the density on the latent variables and $\{\phi_n\}_{n=1}^N$ are factors of the binary or multinomial conditional distribution (Tipping, 1999; Marlin et al., 2011).

In Bayesian supervised learning, $Z$ is the marginal likelihood, otherwise termed the evidence, and the target density $p(\mathbf{w})$ is the posterior of the parameters conditioned on the data. Evaluating $Z$ is essential for the purposes of model comparison, hyperparameter estimation, active learning and experimental design. Indeed, any marginal function of the posterior such as a moment, or a predictive density estimate also implicitly requires $Z$.

In unsupervised learning, $Z$ is the model likelihood obtained by marginalising out the hidden variables $\mathbf{w}$ and $p(\mathbf{w})$ is the density of the hidden variables conditioned on the visible variables. $p(\mathbf{w})$ is required to optimise model parameters using either expectation maximisation or gradient ascent methods.

Computing $Z$, in either the Bayesian or unsupervised learning setting, is typically intractable due to the size of most problems of practical interest, which is usually much greater than one both in the dimension $D$ and the number of potential functions $N$. Methods that can efficiently approximate these quantities are thus required.

Due to the importance of this model class, a great deal of effort has been dedicated to finding accurate approximations to $p(\mathbf{w})$ and $Z$. Whilst there are many different possible approximation routes, including sampling, consistency methods such as expectation propagation and perturbation techniques such as the Laplace method, our focus here is on a technique that lower-bounds $Z$ and makes a Gaussian approximation to the target density $p(\mathbf{w})$.

We obtain a Gaussian approximation to $p(\mathbf{w})$ and a lower-bound on $\log Z$ by minimising the Kullback-Leibler divergence between the approximating Gaussian density and $p(\mathbf{w})$. Gaussian Kullback-Leibler approximate inference, which is how we refer to this procedure, is not new (Saul et al., 1996; Barber and Bishop, 1998; Seeger, 1999b; Kuss and Rasmussen, 2005; Opper and Archambeau, 2009). However, as we outline in the following subsection, we provide a number of theoretical and practical developments regarding its application.

### 1.1 Overview

In Section 2 we provide an introduction and overview of Gaussian Kullback-Leibler (G-KL) approximate inference methods for problems of the form of Equation (2) and describe a large class of models for which G-KL inference is feasible.

In Section 3 we address G-KL bound optimisation. We provide conditions on the potential functions $\{\phi_n\}_{n=1}^N$ for which the G-KL bound is smooth and concave. Thus we provide conditions for which optimisation using Newton's method will exhibit quadratic convergence rates and using quasi-Newton methods superlinear convergence rates.

In Section 4 we discuss the complexity of G-KL bound and gradient computations required to perform approximate inference. To make G-KL approximate inference scalable we present constrained parameterisations of covariance.

In Section 5 we compare G-KL approximate inference to other Gaussian approximate inference methods. We prove that the G-KL lower-bound is tighter than the bound offered by local

lower-bounding methods. We also discuss and compare computational scaling properties and model applicability issues.

In Section 6 we apply the G-KL procedure to three popular machine learning models. First, we consider the problem of Gaussian process regression with noise robust non-conjugate likelihoods. Second, we apply G-KL approximate inference to large Bayesian binary classification tasks. Third, we consider sequential experimental design in Bayesian sparse linear models. In these experiments we aim to assess the performance of the G-KL procedure in terms of speed, accuracy of inference and predictive performance. Results are compared to other deterministic Gaussian approximate inference procedures.

## 2. Gaussian KL Approximate Inference

The primary assumption of this work is that a target density of the form of Equation (2) with unbounded support in $\mathbb{R}^D$ is reasonably approximated by a Gaussian. Many approximate inference methods make this assumption, for example the Laplace approximation (see Barber, 2012 for a recent introduction), expectation propagation with an assumed Gaussian approximating density (Minka, 2001) and local variational bounding methods (Jaakkola and Jordan, 1997). This paper considers the method of fitting a Gaussian to $p(\mathbf{w})$ by minimising the Kullback-Leibler divergence between the two densities.

The Kullback-Leibler (KL) divergence for two probability density functions $q(\mathbf{w})$ and $p(\mathbf{w})$ is defined as

$$\mathrm{KL}(q(\mathbf{w})|\,p(\mathbf{w})) := \int_{\mathcal{W}} q(\mathbf{w}) \log \frac{q(\mathbf{w})}{p(\mathbf{w})} d\mathbf{w}, \tag{3}$$

where $\mathcal{W}$ is the support of $q(\mathbf{w})$. The KL divergence has the properties: $\mathrm{KL}(q(\mathbf{w})|\,p(\mathbf{w})) \geq 0$ for all $p(\mathbf{w})$ and $q(\mathbf{w})$, $\mathrm{KL}(q(\mathbf{w})|\,p(\mathbf{w})) = 0$ iff $q(\mathbf{w}) = p(\mathbf{w})$ almost everywhere, and $\mathrm{KL}(q(\mathbf{w})|\,p(\mathbf{w})) \neq \mathrm{KL}(p(\mathbf{w})|\,q(\mathbf{w}))$ for $q(\mathbf{w}) \neq p(\mathbf{w})$. The KL divergence, whilst not being a true metric, is thus a measure of the discrepancy between two probability distributions.

G-KL approximate inference proceeds by fitting the 'variational' Gaussian, $q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})$, to the target, $p(\mathbf{w})$, by minimising $\mathrm{KL}(q(\mathbf{w})|\,p(\mathbf{w}))$ with respect to the moments $\mathbf{m}$ and $\mathbf{S}$. Substituting Equation (1) into Equation (3), using the fact that the KL divergence is non-negative, we obtain the bound $\log Z \geq \mathcal{B}_{KL}(\mathbf{m},\mathbf{S})$ where

$$\mathcal{B}_{KL}(\mathbf{m},\mathbf{S}) := \underbrace{-\left\langle \log q(\mathbf{w}) \right\rangle_{q(\mathbf{w})}}_{\text{entropy}} + \underbrace{\left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\boldsymbol{\Sigma}) \right\rangle_{q(\mathbf{w})}}_{\text{Gaussian potential}} + \underbrace{\sum_{n=1}^{N} \left\langle \log \phi_n(\mathbf{w}) \right\rangle_{q(\mathbf{w})}}_{\text{site potentials}}, \tag{4}$$

and $\left\langle f(x) \right\rangle_{p(x)}$ denotes taking the expectation of $f(x)$ with respect to the density $p(x)$. Unless otherwise stated all expectations should be assumed to be taken with respect to $q(\mathbf{w})$ and so we omit this subscript in the following notation.

The first and second terms in Equation (4) are integrals that admit simple analytic forms, the last term in general does not. The entropy of the variational Gaussian distribution is equal to $\frac{1}{2}\log\det(2\pi e\mathbf{S})$. The second term is the Gaussian expectation of a negative quadratic in $\mathbf{w}$ which is itself a negative quadratic in $\mathbf{m},\mathbf{S}$. Whilst the site potentials are not, in full generality, easy to evaluate in the next section we describe a large class of models for which they can be computed efficiently and so for which the G-KL bound is tractable.

### 2.1 Tractable G-KL Approximations

To evaluate the G-KL bound, Equation (4), we are required to compute $\sum_{n=1}^{N} \langle \log \phi_n(\mathbf{w}) \rangle$. For generic potential functions $\{\phi_n\}_{n=1}^{N}$ computing the required integrals is not always a numerically accessible task. However, in many practical problems of interest each potential function $\phi_n$ takes the form

$$\phi_n(\mathbf{w}) = \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n), \tag{5}$$

for fixed vectors $\mathbf{h}_n$. We refer to such potentials as site projections. We note that the linear projection of a Gaussian random vector is also Gaussian distributed. That is to say if $y = \mathbf{w}^\mathsf{T}\mathbf{h}$ where $\mathbf{w} \sim \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})$ and $\mathbf{h}$ is fixed then $y \sim \mathcal{N}\left(y|\mathbf{h}^\mathsf{T}\mathbf{m}, \mathbf{h}^\mathsf{T}\mathbf{S}\mathbf{h}\right)$. We can use this result to express $\left\langle \log \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n) \right\rangle$ as a one-dimensional integral

$$\left\langle \log \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n) \right\rangle = \langle \log \phi_n(x) \rangle_{\mathcal{N}(x|m_n,s_n^2)} = \langle \log \phi_n(m_n + z s_n) \rangle_{\mathcal{N}(z|0,1)} \tag{6}$$

with $m_n := \mathbf{m}^\mathsf{T}\mathbf{h}_n$ and $s_n^2 := \mathbf{h}_n^\mathsf{T}\mathbf{S}\mathbf{h}_n$ (this result is presented in Barber and Bishop (1998) and Kuss and Rasmussen (2005) and Appendix A of this paper). The required integral can then be readily computed either analytically (for example $\phi(x) \propto e^{-|x|}$) or more generally using any one-dimensional numerical integration routine.

For models of the form of Equation (2), with each potential $\phi_n(\mathbf{w})$ a site projection, the G-KL bound can thus be expressed as

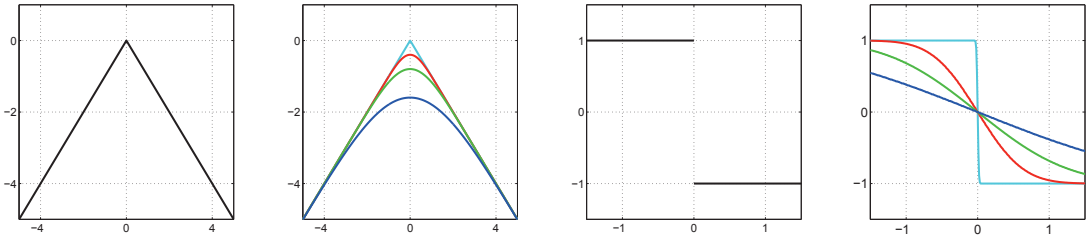
$$\mathcal{B}_{KL}(\mathbf{m},\mathbf{S}) = \underbrace{\frac{1}{2}\log\det(2\pi e\mathbf{S})}_{\text{entropy}} + \underbrace{\sum_{n=1}^{N} \langle \log \phi_n(m_n + z s_n) \rangle_{\mathcal{N}(z|0,1)}}_{\text{site projection potentials}} - \underbrace{\frac{1}{2}\left[\log\det(2\pi\boldsymbol{\Sigma}) + (\mathbf{m}-\boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{m}-\boldsymbol{\mu}) + \operatorname{trace}\left(\boldsymbol{\Sigma}^{-1}\mathbf{S}\right)\right]}_{\text{Gaussian potential}}. \tag{7}$$

#### 2.1.1 NON-GAUSSIAN MODELS

G-KL approximate inference is not limited to models where the target density has a Gaussian potential $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\boldsymbol{\Sigma})$. The bound can be evaluated in the more general case $p(\mathbf{w}) \propto \prod_{n=1}^{N} \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)$ with each $\phi_n$ non-Gaussian. One concrete example of this scenario is binary logistic regression with a Laplace prior on the parameter vector $\mathbf{w}$. In this context each $\phi_n$ potential function corresponds to either the logistic sigmoid factors specifying the likelihood or the $e^{-|w_d|}$ Laplace factors specifying the prior. When $p(\mathbf{w}) \propto \prod_{n=1}^{N} \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)$ the G-KL bound consists only of the first two terms in Equation (7).

## 3. G-KL Bound Optimisation

G-KL approximate inference proceeds to obtain the tightest lower-bound to $\log Z$ and the 'closest' Gaussian approximation to $p(\mathbf{w})$ by maximising $\mathcal{B}_{KL}(\mathbf{m},\mathbf{S})$ with respect to the moments $\mathbf{m}$ and $\mathbf{S}$ of the variational Gaussian density. Therefore, to realise the benefits of G-KL approximate inference we require stable and scalable algorithms to optimise the bound. To this end we now show that for a broad class of models the G-KL objective is both differentiable and concave.



(a) $-|x|$ (b) $-\langle|x|\rangle$ (c) $-\text{sgn}(x)$ (d) $-\langle\text{sgn}(x)\rangle$

Figure 1: Non-differentiable functions and their Gaussian expectations. Figures (a) and (c) plot the non-differentiable function $\psi(x) = -|x|$ and the non-continuous function $\psi(x) = -\text{sgn}(x)$. Figures (b) and (c) plot the expectations of those functions for Gaussian distributed $x$ as a function of the Gaussian mean $m$: $\langle\psi(x)\rangle_{\mathcal{N}(x|m,\sigma^2)}$. The expectations are smooth w.r.t. the Gaussian mean. As the variance of the Gaussian tends to zero the expectation converges to the underlying function value. Gaussian expectations taken w.r.t. $\mathcal{N}\left(x|m,\sigma^2\right)$ where $\sigma = 0.0125, 0.5, 1, 2$.

### 3.1 G-KL Bound Differentiability

Whilst the target density of our model may not be differentiable in $\mathbf{w}$ the G-KL bound with respect to the variational moments $\mathbf{m}, \mathbf{S}$ frequently is. See Figure 1 for a depiction of this phenomenon for two, simple, non-differentiable functions. The G-KL bound is in fact smooth for potential functions that are neither differentiable nor continuous (for example they have jump discontinuities). In Appendix C we show that the G-KL bound is smooth for potential functions that are piecewise smooth with a finite number of discontinuities, and where the logarithm of each piecewise segment is a quadratic. This class of functions includes the widely used Laplace density amongst others.

### 3.2 G-KL Bound Concavity

If each site potential $\{\phi_n\}_{n=1}^N$ is log-concave then the G-KL bound $\mathcal{B}_{KL}(\mathbf{m}, \mathbf{S})$ is jointly concave with respect to the variational Gaussian mean $\mathbf{m}$ and $\mathbf{C}$ the upper triangular Cholesky decomposition of covariance such that $\mathbf{S} = \mathbf{C}^\mathsf{T}\mathbf{C}$. We say that $f(x)$ is log-concave if $\log f(x)$ is concave in $x$.

Since the bound depends on the logarithm of $\prod_{n=1}^N \phi_n$ without loss of generality we may take $N = 1$. Ignoring constants with respect to $\mathbf{m}$ and $\mathbf{C}$, we can write the G-KL bound as

$$\mathcal{B}_{KL}(\mathbf{m}, \mathbf{C}) \overset{\text{c.}}{=} \sum_{d=1}^{D} \log C_{dd} - \frac{1}{2}\mathbf{m}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{m} + \boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{m} - \frac{1}{2}\text{trace}\left(\boldsymbol{\Sigma}^{-1}\mathbf{C}\mathbf{C}^\mathsf{T}\right) + \left\langle \log\phi(\mathbf{w}^\mathsf{T}\mathbf{h})\right\rangle. \quad (8)$$

Excluding $\left\langle \log\phi(\mathbf{w}^\mathsf{T}\mathbf{h})\right\rangle$ from the expression above all terms are concave functions exclusively in either $\mathbf{m}$ or $\mathbf{C}$. Since the sum of concave functions on distinct variables is jointly concave, the first four terms of Equation (8) represent a jointly concave contribution to the bound.

To complete the proof[1] we need to show that $\left\langle \log \phi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle$ is jointly concave in $\mathbf{m}$ and $\mathbf{C}$. Log-concavity of $\phi(x)$ is equivalent to the statement that for any $x_1, x_2 \in \mathbb{R}$ and any $\theta \in [0,1]$

$$\log \phi(\theta x_1 + (1-\theta)x_2) \geq \theta \log \phi(x_1) + (1-\theta)\log \phi(x_2). \tag{9}$$

Therefore, to show that $\mathcal{E}(\mathbf{m}, \mathbf{C}) := \left\langle \log \phi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle_{\mathcal{N}(\mathbf{w}|\mathbf{m}, \mathbf{C}^\mathsf{T}\mathbf{C})}$ is concave it suffices to show for any $\theta \in [0,1]$ that

$$\mathcal{E}(\theta\mathbf{m}_1 + (1-\theta)\mathbf{m}_2, \theta\mathbf{C}_1 + (1-\theta)\mathbf{C}_2) \geq \theta\mathcal{E}(\mathbf{m}_1, \mathbf{C}_1) + (1-\theta)\mathcal{E}(\mathbf{m}_2, \mathbf{C}_2).$$

This can be done by making the substitution $\mathbf{w} = \theta\mathbf{m}_1 + (1-\theta)\mathbf{m}_2 + (\theta\mathbf{C}_1 + (1-\theta)\mathbf{C}_2)^\mathsf{T}\mathbf{z}$, giving

$$\begin{aligned}\mathcal{E}(\theta\mathbf{m}_1 + (1-\theta)\mathbf{m}_2, \theta\mathbf{C}_1 + (1-\theta)\mathbf{C}_2) = \int \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \times \\ \log \phi\left(\theta\mathbf{h}^\mathsf{T}\left(\mathbf{m}_1 + \mathbf{C}_1^\mathsf{T}\mathbf{z}\right) + (1-\theta)\mathbf{h}^\mathsf{T}\left(\mathbf{m}_2 + \mathbf{C}_2^\mathsf{T}\mathbf{z}\right)\right) d\mathbf{z}.\end{aligned}$$

Using concavity of $\log \phi(x)$ with respect to $x$ and Equation (9) with $\mathbf{w}_1 = \mathbf{m}_1 + \mathbf{C}_1^\mathsf{T}\mathbf{z}$ and $\mathbf{w}_2 = \mathbf{m}_2 + \mathbf{C}_2^\mathsf{T}\mathbf{z}$ we have that

$$\begin{aligned}\mathcal{E}(\theta\mathbf{m}_1 + (1-\theta)\mathbf{m}_2, \theta\mathbf{C}_1 + (1-\theta)\mathbf{C}_2) &\geq \theta \int \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \log \phi\left(\mathbf{h}^\mathsf{T}\left(\mathbf{m}_1 + \mathbf{C}_1^\mathsf{T}\mathbf{z}\right)\right) d\mathbf{z} \\ &\quad + (1-\theta) \int \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \log \phi\left(\mathbf{h}^\mathsf{T}\left(\mathbf{m}_2 + \mathbf{C}_2^\mathsf{T}\mathbf{z}\right)\right) d\mathbf{z} \\ &= \theta\mathcal{E}(\mathbf{m}_1, \mathbf{C}_1) + (1-\theta)\mathcal{E}(\mathbf{m}_2, \mathbf{C}_2).\end{aligned}$$

Thus the G-KL bound is jointly concave in $\mathbf{m}, \mathbf{C}$ provided all site potentials $\{\phi_n\}_{n=1}^N$ are log-concave.

With consequence to the theoretical convergence rates of gradient based optimisation procedures, the bound is also strongly-concave. A function $f(\mathbf{x})$ is strongly-concave if there exists some $c < 0$ such that for all $\mathbf{x}$, $\nabla^2 f(\mathbf{x}) \preceq c\mathbf{I}$ (Boyd and Vandenberghe, 2004, Section 9.1.2).[2] For the G-KL bound the constant $c$ can be assessed by inspecting the covariance of the Gaussian potential, $\mathbf{\Sigma}$. If we arrange the set of all G-KL variational parameters as a vector formed by concatenating $\mathbf{m}$ and the non-zero elements of the column's of $\mathbf{C}$ then the Hessian of $\left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \mathbf{\Sigma}) \right\rangle$ is a block diagonal matrix. Each block of this Hessian is either $-\mathbf{\Sigma}^{-1}$ or its submatrix $\left[-\mathbf{\Sigma}^{-1}\right]_{i:D,i:D}$, where $i = 2, \ldots, D$. The set of eigenvalues of a block diagonal matrix is the union of the eigenvalues of each of the block matrices' eigenvalues. Furthermore, the eigenvalues of each submatrix are bounded by the upper and lower eigenvalues of $-\mathbf{\Sigma}^{-1}$. Therefore $\nabla^2 \mathcal{B}_{KL}(\mathbf{m}, \mathbf{S}) \succeq c\mathbf{I}$ where $c$ is $-1$ times the smallest eigenvalue of $\mathbf{\Sigma}^{-1}$. The sum of a strongly-concave function and a concave function is strongly-concave and thus the G-KL bound as a whole is strongly-concave.

### 3.3 Summary

In this section, and in Appendix C, we have provided conditions for which the G-KL bound is strongly concave, smooth, has closed sublevel sets and Lipschitz continuous Hessians. Under these

1. This proof was provided by Michalis K. Titsias and simplifies the original presentation made in (Challis and Barber, 2011).
2. We say for square matrices $\mathbf{A}$ and $\mathbf{B}$ that $\mathbf{A} \preceq \mathbf{B}$ iff $\mathbf{B} - \mathbf{A}$ is positive semidefinite.

conditions optimisation of the G-KL bound will have quadratic convergence rates using Newton's method and super-linear convergence rates using quasi-Newton methods (Nocedal and Wright, 2006; Boyd and Vandenberghe, 2004). For larger problems, where cubic scaling properties arising from the approximate Hessian calculations required by quasi-Newton methods are infeasible, we will use limited memory quasi-Newton methods, nonlinear conjugate gradients or Hessian free Newton methods to optimise the G-KL bound.

Concavity with respect to the G-KL mean is clear and intuitive—for any fixed G-KL covariance the G-KL bound as a function of the mean can be interpreted as a Gaussian blurring of $\log p(\mathbf{w})$—see Figure 1. As $\mathbf{S} = \nu^2\mathbf{I} \to \mathbf{0}$ then $\mathbf{m}^* \to \mathbf{w}^{MAP}$ where $\mathbf{m}^*$ is the optimal G-KL mean and $\mathbf{w}^{MAP}$ is the maximum a posteriori (MAP) parameter setting.

Another deterministic Gaussian approximate inference procedure for models of the form of Equation (2) are local variational bounding methods (discussed at further length in Section 5.1.1). For log-concave potentials local variational bounding methods, which optimise a different criterion with a different parameterisation to the G-KL bound, have also been shown to result in a convex optimisation problem (Seeger and Nickisch, 2011b). To the best of our knowledge, local variational bounding and G-KL approximate inference methods are the only known concave variational inference procedures for models of the form of Equation (2).

Whilst G-KL bound optimisation and MAP estimation share conditions under which they are concave problems, the G-KL objective is often differentiable when the MAP objective is not. Non-differentiable potentials are used throughout machine learning and statistics. Indeed, the practical utility of such non-differentiable potentials in statistical modelling has driven a lot of research into speeding up algorithms to find the mode of these densities—for example see Schmidt et al. (2007). Despite recent progress these algorithms tend to have slower convergence rates than quasi-Newton methods on smooth, strongly-convex objectives with Lipschitz continuous gradients and Hessians.

One of the significant practical advantages of G-KL approximate inference over MAP estimation and the Laplace approximation is that the target density is not required to be differentiable. With regards to the complexity of G-KL bound optimisation, whilst an additional cost is incurred over MAP estimation from specifying and optimising the variance of the approximation, a saving is made in the number of times the objective and its gradients need to be computed. Quantifying the net saving (or indeed cost) of G-KL optimisation over MAP estimation is an interesting question reserved for later work.

## 4. Complexity : G-KL Bound and Gradient Computations

In the previous section we provided conditions for which the G-KL bound is strongly concave and differentiable and so provided conditions for which G-KL bound optimisation using quasi-Newton methods will exhibit super-linear convergence rates. Whilst such convergence rates are highly desirable they do not in themselves guarantee that optimisation is scalable. An important practical consideration is the numerical complexity of the bound and gradient computations required by any gradient ascent optimisation procedure.

Discussing the complexity of G-KL bound and gradient evaluations in full generality is complex we therefore restrict ourselves to considering one particularly common case. We consider models where the covariance of the Gaussian potential in Equation (2) is spherical, $\mathbf{\Sigma} = \nu^2\mathbf{I}$, and each potential function is a site projection, $\phi_n(\mathbf{w}) = \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)$. For models that do not satisfy this assumption, in Appendix D we present a full breakdown of the complexity of bound and gradient

computations for each G-KL covariance parameterisation presented in Section 4.1.3 and a range of parameterisations for the Gaussian potential $\mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{\Sigma})$.

Note that problems where $\mathbf{\Sigma}$ is not a scaling of the identity can be reparameterised to an equivalent problem for which it is. For some problems this reparameterisation can provide significant reductions in complexity. The procedure, the domains for which it is suitable, and the possible computational savings it provides are discussed at further length in Appendix E.

For Cholesky factorisations of covariance, $\mathbf{S} = \mathbf{C}^\mathsf{T}\mathbf{C}$, of dimension $D$ the bound and gradient contributions from the $\log\det(\mathbf{S})$ and $\mathrm{trace}(\mathbf{S})$ terms in Equation (7) scale $O(D)$ and $O\left(D^2\right)$ respectively. Terms in Equation (7) that are a function exclusively of the G-KL mean, $\mathbf{m}$, scale at most $O(D)$ and are the cheapest to evaluate. The computational bottleneck arises from the projected variational variances $s_n^2 = \|\mathbf{C}^\mathsf{T}\mathbf{h}_n\|^2$ required to compute each $\left\langle \log\phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)\right\rangle$ term. Computing all such projected variances scales $O\left(ND^2\right)$.[3]

A further computational expense is incurred from computing the $N$ one dimensional integrals required to evaluate $\sum_{n=1}^N \left\langle \log\phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)\right\rangle$. These integrals are computed either numerically or analytically depending on the functional form of $\phi_n$. Regardless, this computation scales $O(N)$, possibly though with a significant prefactor. When numerical integration is required, we note that since $\left\langle \log\phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)\right\rangle$ can be expressed as $\langle \log\phi_n(m_n + z s_n)\rangle_{\mathcal{N}(z|0,1)}$ we can usually assert that the integrand's significant mass lies for $z \in [-5,5]$ and so that quadrature will yield sufficiently accurate results at modest computational expense. For all the experiments considered here we used fixed width rectangular quadrature and performing these integrals was not the principal bottleneck. For modelling scenarios where this is not the case we note that a two dimensional lookup table can be constructed, at a one off cost, to approximate $\langle \log\phi(m+zs)\rangle_{\mathcal{N}(z|0,1)}$ and its derivatives as a function of $m$ and $s$.

Thus for a broad class of models the G-KL bound and gradient computations scale $O\left(ND^2\right)$ for general parameterisations of the covariance $\mathbf{S} = \mathbf{C}^\mathsf{T}\mathbf{C}$. In many problems of interest the fixed vectors $\mathbf{h}_n$ are sparse. Letting $L$ denote the number of non-zero elements in each vector $\mathbf{h}_n$, computing $\left\{s_n^2\right\}_{n=1}^N$ scales now $O(NDL)$ where frequently $L \ll D$. Nevertheless, such scaling for the G-KL method can be prohibitive for large problems and so constrained parameterisations are required.

### 4.1 Constrained Parameterisations of G-KL Covariance

Unconstrained G-KL approximate inference requires storing and optimising $\frac{1}{2}D(D+1)$ parameters to specify the G-KL covariance's Cholesky factor $\mathbf{C}$. In many settings this can be prohibitive. To this end we now consider constrained parameterisations of covariance that reduce both the time and space complexity of G-KL procedures.

Gaussian densities can be parameterised with respect to the covariance or its inverse the precision matrix. A natural question to ask is which of these is best suited for G-KL bound optimisation. Unfortunately, the G-KL bound is neither concave nor convex with respect to the precision matrix. What is more, the complexity of computing the $\phi_n$ site potential contributions to the bound increases for the precision parameterised G-KL bound. Thus the G-KL bound seems more naturally parameterised in terms of covariance than precision.

3. We note that since a Gaussian potential, $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\mathbf{\Sigma})$, can be written as a product over $D$ site projection potentials computing $\left\langle \log\mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\mathbf{\Sigma})\right\rangle$ will in general scale $O\left(D^3\right)$—see Appendix B.3.2.

#### 4.1.1 OPTIMAL G-KL COVARIANCE STRUCTURE

As originally noted by Seeger (1999a), the optimal structure for the G-KL covariance can be assessed by calculating the derivative of $\mathcal{B}_{KL}(\mathbf{m},\mathbf{S})$ with respect to $\mathbf{S}$ and equating it to zero. Doing so, $\mathbf{S}$ satisfies

$$\mathbf{S}^{-1} = \boldsymbol{\Sigma}^{-1} + \mathbf{H}\boldsymbol{\Gamma}\mathbf{H}^{\mathsf{T}}, \tag{10}$$

where $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_n]$ and $\boldsymbol{\Gamma}$ is diagonal such that

$$\Gamma_{nn} = \left\langle \left(z^2 - 1\right) \frac{\log \phi_n(m_n + z s_n)}{2 s_n^2} \right\rangle_{\mathcal{N}(z|0,1)}. \tag{11}$$

$\boldsymbol{\Gamma}$ depends on $\mathbf{S}$ through the projected variance terms $s_n^2 = \mathbf{h}_n^{\mathsf{T}}\mathbf{S}\mathbf{h}_n$ and Equation (10) does not provide a closed form expression to solve for $\mathbf{S}$. Furthermore, iterating Equation (10) is not guaranteed to converge to a fixed point or uniformly increase the bound. Indeed this iterative procedure frequently diverges. We are free, however, to directly optimise the bound by treating the diagonal entries of $\boldsymbol{\Gamma}$ as variational parameters and thus change the number of parameters required to specify $\mathbf{S}$ from $\frac{1}{2}D(D+1)$ to $N$. This procedure, whilst possibly reducing the number of free parameters, requires us to compute $\log\det(\mathbf{S})$ where $\mathbf{S}$ has no convenient structure and so in general scales $O\left(D^3\right)$—infeasible when $D \gg 1$.

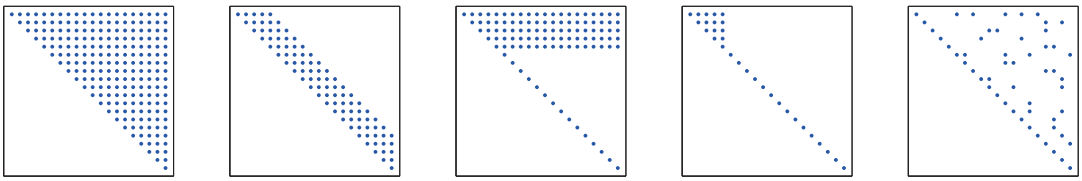
A further consequence of using this parameterisation of covariance is that the bound is non-concave. We know from Seeger and Nickisch (2011b) that parameterising $\mathbf{S}$ according to Equation (10) renders $\log\det(\mathbf{S})$ concave with respect to $(\Gamma_{nn})^{-1}$. However the site projection potentials are not concave with respect to $(\Gamma_{nn})^{-1}$ thus the bound is neither concave nor convex for this parameterisation resulting in convergence to a possibly local optimum. Non-convexity and $O\left(D^3\right)$ scaling motivates the search for better parameterisations of covariance. In Appendix B we provide equations for each term of the G-KL bound and its gradient for each of the covariance parameterisations considered below.

#### 4.1.2 FACTOR ANALYSIS

Parameterisations of the form $\mathbf{S} = \boldsymbol{\Theta}\boldsymbol{\Theta}^{\mathsf{T}} + \operatorname{diag}\left(\mathbf{d}^2\right)$ can capture the $K$ leading directions of variance for a $D \times K$ dimensional loading matrix $\boldsymbol{\Theta}$. Unfortunately this parameterisation renders the G-KL bound non-concave. Non-concavity is due to the entropic contribution $\log\det(\mathbf{S})$ which is not even unimodal. All other terms in the bound remain concave under this factorisation. Provided one is happy to accept convergence to possibly local optima, this is still a useful parameterisation. Computing the projected variances with $\mathbf{S}$ in this form scales $O(NDK)$ and evaluating $\log\det(\mathbf{S})$ and its derivative scales $O\left(K^2(K+D)\right)$.

#### 4.1.3 CONSTRAINED CONCAVE PARAMETERISATIONS

Below we present constrained parameterisations of covariance which reduce both the space and time complexity of G-KL bound optimisation whilst preserving concavity. To reiterate, the computational scaling figures for the bound and gradient computations listed below correspond to evaluating the projected G-KL variance terms, the bottleneck for models with an isotropic Gaussian potential $\boldsymbol{\Sigma} = \nu^2\mathbf{I}$. The scaling properties for other models are presented in Appendix D. The constrained parameterisations below have different qualities regarding the expressiveness of the variational Gaussian approximation. We note that a zero at the $(i,j)^{th}$ element of covariance specifies



(a) Full (b) Banded (c) Chevron (d) Subspace (e) Sparse

Figure 2: Sparsity structure for constrained concave Cholesky decompositions of covariance.

a marginal independence relation between parameters $w_i$ and $w_j$. Conversely, a zero at the $(i,j)^{th}$ element of precision corresponds to a independence relation between parameters $w_i$ and $w_j$ when conditioned on the other remaining parameters.

*Banded Cholesky.* The simplest option is to constrain the Cholesky matrix to be banded, that is $C_{ij} = 0$ for $j > i + B$ where $B$ is the bandwidth. Doing so reduces the cost of a single bound or gradient computation to $O(NDB)$. Such a parameterisation describes a sparse covariance matrix and assumes zero covariance between variables that are indexed out of bandwidth. The precision matrix for banded Cholesky factorisations of covariance will in general be non-sparse.

*Chevron Cholesky.* We constrain $\mathbf{C}$ such that $C_{ij} = \Theta_{ij}$ when $j \geq i$ and $i \leq K$, $C_{ii} = d_i$ for $i > K$ and 0 otherwise. We refer to this parameterisation as the chevron Cholesky since the sparsity structure has a broad inverted 'V' shape—see Figure 2. Generally, this constrained parameterisation results in a non-sparse covariance but sparse precision. This parameterisation is not invariant to index permutations and so not all covariates have the same representational power. For a Cholesky matrix of this form bound and gradient computations scale $O(NDK)$.

*Sparse Cholesky.* In general the bound and gradient can be evaluated more efficiently if we impose any fixed sparsity structure on the Cholesky matrix $\mathbf{C}$. In certain modelling scenarios we know a priori which variables are marginally dependent and independent and so may be able construct a sparse Cholesky matrix to reflect that domain knowledge. This is of use in cases where a low band width index ordering cannot be found. For a sparse Cholesky matrix with $DK$ non-zero elements bound and gradient computations scale $O(NDK)$.

*Subspace Cholesky.* Another reduced parameterisation of covariance can be obtained by considering arbitrary rotations in parameter space, $\mathbf{S} = \mathbf{E}^\mathsf{T}\mathbf{C}^\mathsf{T}\mathbf{C}\mathbf{E}$ where $\mathbf{E}$ is a rotation matrix which forms an orthonormal basis over $\mathbb{R}^D$. Substituting this form for the covariance into Equation (8) and for $\mathbf{\Sigma} = \nu^2\mathbf{I}$ we obtain, up to a constant,

$$\mathcal{B}_{KL}(\mathbf{m},\mathbf{C}) \stackrel{c.}{=} \sum_i \log C_{ii} - \frac{1}{2\nu^2}\left[\|\mathbf{C}\|^2 + \|\mathbf{m}\|^2\right] + \frac{1}{\nu^2}\boldsymbol{\mu}^\mathsf{T}\mathbf{m} + \sum_n \langle \log \phi(m_n + z s_n)\rangle_z$$

where $s_n = \|\mathbf{C}^\mathsf{T}\mathbf{E}^\mathsf{T}\mathbf{h}_n\|$. One may reduce the computational burden by decomposing $\mathbf{E}$ into two submatrices such that $\mathbf{E} = [\mathbf{E}_1, \mathbf{E}_2]$ where $\mathbf{E}_1$ is $D \times K$ and $\mathbf{E}_2$ is $D \times L$ for $L = (D-K)$. Constraining $\mathbf{C}$ such that $\mathbf{C} = \text{blkdiag}\,(\mathbf{C}_1, c\mathbf{I}_{L\times L})$, with $\mathbf{C}_1$ a $K \times K$ Cholesky matrix we have that

$$s_n^2 = \|\mathbf{C}_1^\mathsf{T}\mathbf{E}_1^\mathsf{T}\mathbf{h}_n\|^2 + c^2(\|\mathbf{h}_n\|^2 - \|\mathbf{E}_1^\mathsf{T}\mathbf{h}_n\|^2),$$

meaning that only the $K$ subspace vectors in $\mathbf{E}_1$ are needed to compute $\left\{s_n^2\right\}_{n=1}^N$. Since terms such as $\|\mathbf{h}_n\|$ need only be computed once the complexity of bound and gradient computations reduces to

scaling in $K$ not $D$. Further savings can be made if we use banded subspace Cholesky matrices: for $\mathbf{C}_1$ having bandwidth $B$ each bound evaluation and associated gradient computation scales $O(NBK)$.

The success of the subspace Cholesky factorisation depends on how well $\mathbf{E}_1$ captures the leading directions of variance. One simple approach to select $\mathbf{E}_1$ is to use the leading principal components of the 'data set' $\mathbf{H}$. Another option is iterate between optimising the bound with respect to $\{\mathbf{m}, \mathbf{C}_1, c\}$ and $\mathbf{E}_1$. We consider two approaches for optimisation with respect to $\mathbf{E}_1$. The first uses the form for the optimal G-KL covariance, Equation (11). By substituting in the projected mean and variance terms $m_n$ and $s_n^2$ into Equation (11) we can set $\mathbf{E}_1$ to be a rank $K$ approximation to this $\mathbf{S}$. The best rank $K$ approximation is given by evaluating the smallest $K$ eigenvectors of $\boldsymbol{\Sigma}^{-1} + \mathbf{H}\boldsymbol{\Gamma}\mathbf{H}^{\mathsf{T}}$. For very large sparse problems $D \gg 1$ we approximate this using the iterative Lanczos methods described by Seeger and Nickisch (2010). For smaller non-sparse problems more accurate approximations are available. The second approach is to optimise the G-KL bound directly with respect to $\mathbf{E}_1$ under the constraint that the columns of $\mathbf{E}_1$ are orthonormal. One route to achieving this is to use a projected gradient ascent method. Each of these methods and the associated subspace G-KL gradients are presented in greater detail in Appendix B.4.

## 5. Comparing Gaussian Approximate Inference Procedures

Due to their favourable computational and analytical properties multivariate Gaussian densities are used by many deterministic approximate inference routines. For models of the form of Equation (2) three popular, deterministic, Gaussian, approximate inference techniques are local variational bounding, Laplace approximations, and expectation propagation with an assumed Gaussian density. In this section we briefly review and compare these methods to the G-KL procedure.

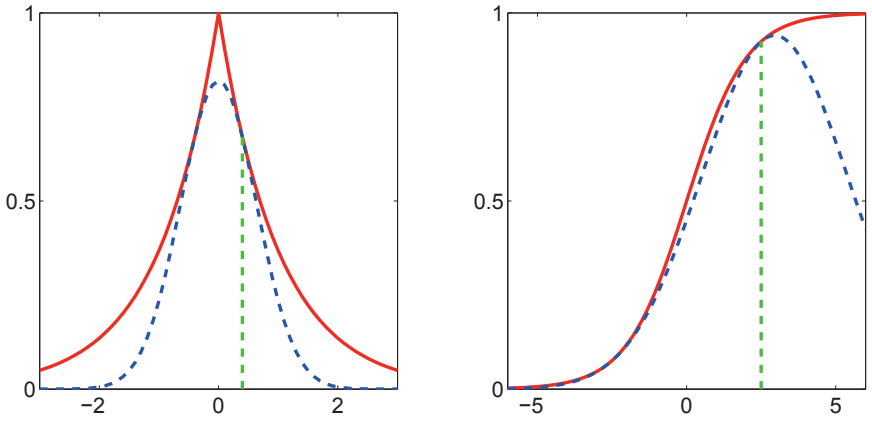
Of the three Gaussian approximate inference methods listed above only one, local variational bounding, provides a lower-bound to the normalisation constant $Z$. In Section 5.1 we give a brief overview of local bounding procedures and show that the G-KL lower-bound dominates the local lower-bound on $\log Z$.

In Section 5.2 we discuss the applicability of each Gaussian approximate inference method. Specifically we describe the computational scaling properties of each of the algorithms and the potential functions to which they can successfully be applied

### 5.1 Gaussian Lower-Bounds

An attractive property of G-KL approximate inference is that it provides a strict lower-bound on $\log Z$. Lower-bounding procedures are particularly useful for a number of theoretical and practical reasons. The primary theoretical advantage is that it provides concrete exact knowledge about $Z$ and thus also the target density $p(\mathbf{w})$. Lower-bounds may also be used in conjunction with upper bounds to form bounds on marginal quantities of interest (Gibbs and MacKay, 2000). Thus the tighter the lower-bound on $\log Z$ the more informative it is. Practically, optimising a lower-bound is often a more numerically stable task than the criteria provided by other deterministic approximate inference methods.

Another well studied route to obtaining a lower-bound for problems of the form of Equation (2) are so called local variational bounding procedures, see for example: Jaakkola and Jordan (1997), Gibbs and MacKay (2000), Girolami (2001), Palmer et al. (2006), and Nickisch and Seeger (2009). Whilst both G-KL and local procedures have been discussed in the literature for some time, little work has been done to elucidate the relation between them. In Section 5.1.1 we give an overview




(a) (b)

Figure 3: Exponentiated quadratic lower-bounds for two super-Gaussian potential functions: (a) Laplace potential and lower-bound with operating point at 0.5; (b) Logistic sigmoid potential and lower-bound with operating point at 2.5.

of local variational bounding procedures. In Section 5.1.2 we prove that G-KL provides a tighter lower-bound on $Z$ than local lower-bounding methods.

### 5.1.1 LOCAL VARIATIONAL BOUNDS

Local variational procedures lower-bound $Z$ by replacing each potential $\phi_n$ in Equation (2) with a function that lower-bounds it and that renders the integral as a whole analytically tractable. Tractability is obtained by using exponentiated quadratic lower-bounds for each non-Gaussian site potential $\{\phi_n\}_{n=1}^N$. Local variational bounding procedures that use exponentiated quadratic site bounds return a Gaussian approximation to the target density $p(\mathbf{w})$.

Site potentials $\phi_n$ are known to have tight exponentiated quadratic lower-bounds provided they are super-Gaussian (Palmer et al., 2006). A function $f(x)$ is said to be super-Gaussian if $\exists b \in \mathbb{R}$ s.t. for $g(x) := \log f(x) - bx$ is even, convex and decreasing as a function of $y = x^2$. A number of potential functions of significant practical utility are super-Gaussian, examples include: the logistic sigmoid $\phi(x) = (1+\exp(-x))^{-1}$, the Laplace density $\phi(x) \propto \exp(-|x|)$ and the Student's $t$ density—see Figure 3 for plots of these potential functions and their respective lower-bounds.

Each site projection potential function is lower-bounded by an exponentiated quadratic parameterised in $\mathbf{w}$ and a variational parameter $\xi_n$. Since exponentiated quadratics are closed under multiplication one may bound the product of site potentials by an exponentiated quadratic also

$$\prod_n \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n) \geq c(\boldsymbol{\xi}) e^{-\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{F}(\xi)\mathbf{w}+\mathbf{w}^\mathsf{T}\mathbf{f}(\boldsymbol{\xi})}, \tag{12}$$

where the matrix $\mathbf{F}(\boldsymbol{\xi})$, vector $\mathbf{f}(\boldsymbol{\xi})$ and scalar $c(\boldsymbol{\xi})$ depend on the specific functions $\{\phi_n\}_{n=1}^N$ and the vectors $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_n]$; and $\boldsymbol{\xi}$ is a vector of length $N$ containing the variational parameters $\xi_n$. For

any setting of $\mathbf{w}$ there exists a setting of $\boldsymbol{\xi}$ for which the bound is tight. Thus we can obtain a bound on $Z$ by substituting Equation (12) into Equation (2):

$$\begin{aligned} Z &= \int \mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\boldsymbol{\Sigma}) \prod_{n=1}^{N} \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n) d\mathbf{w} \\ &\geq \int \mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\boldsymbol{\Sigma})\, c(\boldsymbol{\xi}) e^{-\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{F}(\boldsymbol{\xi})\mathbf{w}+\mathbf{w}^\mathsf{T}\mathbf{f}(\boldsymbol{\xi})} d\mathbf{w} \\ &= c(\boldsymbol{\xi}) \frac{e^{-\frac{1}{2}\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}}}{\sqrt{\det(2\pi\boldsymbol{\Sigma})}} \int e^{-\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{A}\mathbf{w}+\mathbf{w}^\mathsf{T}\mathbf{b}} d\mathbf{w}, \end{aligned} \tag{13}$$

where

$$\mathbf{A} := \boldsymbol{\Sigma}^{-1} + \mathbf{F}(\boldsymbol{\xi}) \quad \text{and} \quad \mathbf{b} := \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \mathbf{f}(\boldsymbol{\xi}). \tag{14}$$

Whilst both $\mathbf{A}$ and $\mathbf{b}$ are functions of $\boldsymbol{\xi}$, we drop this dependency for a more compact notation. One can interpret Equation (13) as a Gaussian approximation to the target density where $p(\mathbf{w}) \approx \mathcal{N}\left(\mathbf{w}|\mathbf{A}^{-1}\mathbf{b},\mathbf{A}^{-1}\right)$. Completing the square in Equation (13) and integrating, we have $\log Z \geq B(\boldsymbol{\xi})$, where

$$\mathcal{B}(\boldsymbol{\xi}) = \log c(\boldsymbol{\xi}) - \frac{1}{2}\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \frac{1}{2}\mathbf{b}^\mathsf{T}\mathbf{A}^{-1}\mathbf{b} - \frac{1}{2}\log\det(2\pi\boldsymbol{\Sigma}) - \frac{1}{2}\log\det(2\pi\mathbf{A}).$$

To obtain the tightest bound on $\log Z$ one then maximises $\mathcal{B}(\boldsymbol{\xi})$ with respect to $\boldsymbol{\xi}$.

### 5.1.2 COMPARING G-KL AND LOCAL BOUNDS

An important question is which method, local or G-KL, gives a tighter lower-bound on $\log Z$. Each bound derives from a fundamentally different criterion and it is not immediately clear which if either is superior. The G-KL bound has been noted before, empirically in the case of binary classification (Nickisch and Rasmussen, 2008) and analytically for the special case of symmetric potentials (Seeger, 2009), to be tighter than the local bound. It is tempting to conclude that such observed superiority of the G-KL method is to be expected since the G-KL bound has potentially unrestricted covariance $\mathbf{S}$ and so a richer parameterisation. However, many problems have more site potentials $\phi_n$ than Gaussian moment parameters, that is $N > \frac{1}{2}D(D+3)$, and the local bound in such cases has a richer parameterisation than the G-KL.

We derive a relation between the local and G-KL bounds for $\{\phi_n\}_{n=1}^N$ generic super-Gaussian site potentials. We first substitute the local bound on $\prod_{n=1}^N \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)$, Equation (12), into Equation (4) to obtain a new bound

$$\mathcal{B}_{KL}(\mathbf{m},\mathbf{S}) \geq \tilde{\mathcal{B}}_{KL}(\mathbf{m},\mathbf{S},\boldsymbol{\xi}),$$

where

$$\begin{aligned} 2\tilde{\mathcal{B}}_{KL} = -2\langle \log q(\mathbf{w})\rangle - \log\det(2\pi\boldsymbol{\Sigma}) + 2\log c(\boldsymbol{\xi}) - \left\langle (\mathbf{w}-\boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\rangle \\ - \left\langle \mathbf{w}^\mathsf{T}\mathbf{F}(\boldsymbol{\xi})\mathbf{w}\right\rangle + 2\left\langle \mathbf{w}^\mathsf{T}\mathbf{f}(\boldsymbol{\xi})\right\rangle. \end{aligned}$$

Using Equation (14) this can be written as

$$\tilde{\mathcal{B}}_{KL} = -\langle \log q(\mathbf{w})\rangle - \frac{1}{2}\log\det(2\pi\boldsymbol{\Sigma}) + \log c(\boldsymbol{\xi}) - \frac{1}{2}\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} - \frac{1}{2}\left\langle \mathbf{w}^\mathsf{T}\mathbf{A}\mathbf{w}\right\rangle + \left\langle \mathbf{w}^\mathsf{T}\mathbf{b}\right\rangle.$$

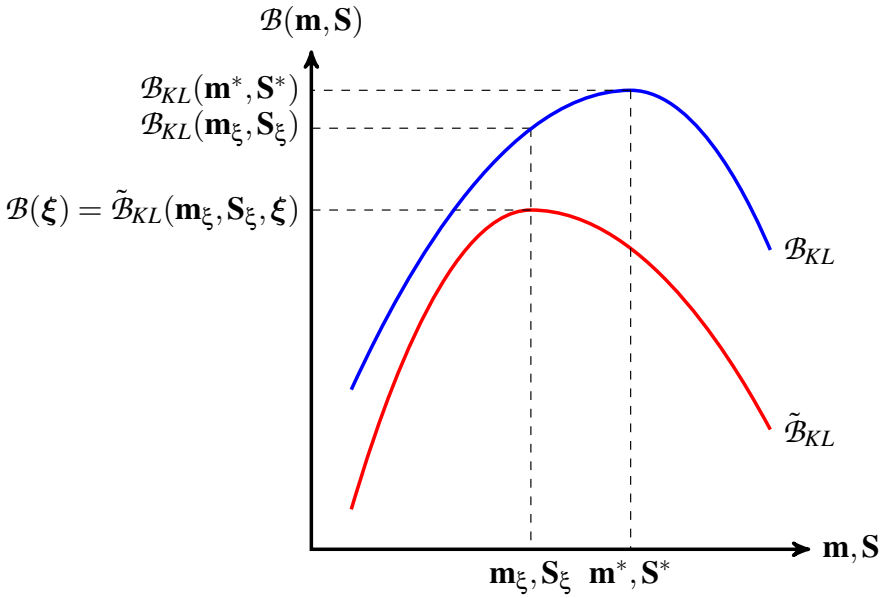



Figure 4: Schematic of the relation between the G-KL bound, $\mathcal{B}_{KL}$ (blue), and the weakened KL bound, $\tilde{\mathcal{B}}_{KL}$ (red), plotted as a function of the Gaussian moments $\mathbf{m}$ and $\mathbf{S}$ with $\boldsymbol{\xi}$ fixed. For any setting of the local site bound parameters $\boldsymbol{\xi}$ we have that $\mathcal{B}_{KL}(\mathbf{m},\mathbf{S}) \geq \tilde{\mathcal{B}}_{KL}(\mathbf{m},\mathbf{S},\boldsymbol{\xi})$. We show in the text that the local bound, $\mathcal{B}(\boldsymbol{\xi})$, is the maximum of the weakened KL bound, that is that $\mathcal{B}(\boldsymbol{\xi}) = \max_{\mathbf{m},\mathbf{S}} \tilde{\mathcal{B}}(\mathbf{m},\mathbf{S},\boldsymbol{\xi})$ with $\mathbf{m}_\xi,\mathbf{S}_\xi = \text{argmax}_{\mathbf{m},\mathbf{S}} \tilde{\mathcal{B}}(\mathbf{m},\mathbf{S},\boldsymbol{\xi})$ in the figure. The G-KL bound can be optimised beyond $\mathcal{B}_{KL}\left(\mathbf{m}_\xi,\mathbf{S}_\xi\right)$ to obtain different, optimal G-KL moments $\mathbf{m}^*$ and $\mathbf{S}^*$ that achieve a tighter lower-bound on $\log Z$.

By defining $\tilde{q}(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{A}^{-1}\mathbf{b},\mathbf{A}^{-1}\right)$ we obtain

$$\tilde{\mathcal{B}}_{KL} = -\text{KL}(\,q(\mathbf{w})|\,\tilde{q}(\mathbf{w})) - \frac{1}{2}\log\det(2\pi\boldsymbol{\Sigma}) + \log c(\boldsymbol{\xi}) - \frac{1}{2}\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \frac{1}{2}\mathbf{b}^\mathsf{T}\mathbf{A}^{-1}\mathbf{b} - \frac{1}{2}\log\det(2\pi\mathbf{A})\,.$$

Since $\mathbf{m},\mathbf{S}$ only appear via $q(\mathbf{w})$ in the KL term, the tightest bound is given when $\mathbf{m},\mathbf{S}$ are set such that $q(\mathbf{w}) = \tilde{q}(\mathbf{w})$. At this setting the KL term in $\tilde{\mathcal{B}}_{KL}$ is zero and $\mathbf{m}$ and $\mathbf{S}$ are given by

$$\mathbf{S}_\xi = \left(\boldsymbol{\Sigma}^{-1} + \mathbf{F}(\boldsymbol{\xi})\right)^{-1}, \qquad \mathbf{m}_\xi = \mathbf{S}_\xi\left(\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \mathbf{f}(\boldsymbol{\xi})\right),$$

and $\tilde{\mathcal{B}}_{KL}\left(\mathbf{m}_\xi,\mathbf{S}_\xi,\boldsymbol{\xi}\right) = \mathcal{B}\left(\boldsymbol{\xi}\right)$. To reiterate, $\mathbf{m}_\xi$ and $\mathbf{S}_\xi$ maximise $\tilde{\mathcal{B}}_{KL}(\mathbf{m},\mathbf{S},\boldsymbol{\xi})$ for any fixed setting of $\boldsymbol{\xi}$. Since $\mathcal{B}_{KL}(\mathbf{m},\mathbf{S}) \geq \tilde{\mathcal{B}}_{KL}(\mathbf{m},\mathbf{S},\boldsymbol{\xi})$ we have that,

$$\mathcal{B}_{KL}(\mathbf{m}_\xi,\mathbf{S}_\xi) \geq \tilde{\mathcal{B}}_{KL}(\mathbf{m}_\xi,\mathbf{S}_\xi,\boldsymbol{\xi}) = \mathcal{B}(\boldsymbol{\xi}).$$

The G-KL bound can be optimised beyond this setting and can achieve an even tighter lower-bound on $\log Z$,

$$\mathcal{B}_{KL}(\mathbf{m}^*, \mathbf{S}^*) = \max_{\mathbf{m},\mathbf{S}} \mathcal{B}_{KL}(\mathbf{m}, \mathbf{S}) \geq \mathcal{B}_{KL}(\mathbf{m}_\xi, \mathbf{S}_\xi).$$

Thus optimal G-KL bounds are provably tighter than both the local variational bound and the G-KL bound calculated using the optimal local bound moments $\mathbf{m}_\xi$ and $\mathbf{S}_\xi$. A graphical depiction of this result is presented in Figure 4.

The experimental results presented in Section 6 show that the improvement in bound values can be significant. Furthermore, constrained parameterisations of covariance, introduced in Section 4, which are required when $D \gg 1$, are also frequently observed to outperform local variational solutions despite the fact that they are not provably guaranteed to do so.

### 5.2 Complexity and Model Suitability Comparison

We briefly review the core computational bottlenecks and the conditions placed on the potential functions by the local variational bounding, the Laplace approximation and the Gaussian expectation propagation approximate inference methods. A more thorough comparison of these techniques in the context of binary Gaussian Process classification can be found in Nickisch and Rasmussen (2008). Subsequently, we go onto summarise and compare these properties versus the G-KL procedure.

#### 5.2.1 LAPLACE APPROXIMATIONS

Laplace methods, see Barber (2012) for an introduction, approximate the target density with a Gaussian whose mean is centered at the mode of $p(\mathbf{w})$ and whose covariance is the inverse Hessian at the mode of $\log p(\mathbf{w})$. The computational complexity of finding the mode is that of a continuous optimisation problem over $D$ real valued parameters on the joint likelihood objective $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_n \phi_n(\mathbf{w})$. Evaluating the Laplace estimate to $\log Z$ requires computing the determinant of the Hessian, and so scales $O\left(D^3\right)$ which, importantly, only needs to be computed once. To apply the Laplace approximation we require that the target density be twice continuously differentiable, that is we require that each potential function $\{\phi_n\}_{n=1}^N$ be twice continuously differentiable. Provided the Laplace approximation is valid it is generally the fastest of the methods listed here.

#### 5.2.2 LOCAL VARIATIONAL BOUNDING

Local variational bounding methods, as detailed in Section 5.1.1, have $N$ free variational parameters—one for each site potential $\phi_n$. Optimising the bound, using either generalised expectation maximisation or gradient based methods, requires solving $N$ linear symmetric $D \times D$ systems. Efficient exact implementations of this method maintain the covariance using its Cholesky factorisation and perform efficient rank one Cholesky updates (Seeger, 2007). Doing so each round of updates scales $O\left(ND^2\right)$. As detailed in Section 5.1.1, local variational bounding procedures are applicable provided tight exponentiated quadratic lower-bounds to the site projection potentials $\{\phi_n\}_{n=1}^N$ exist—that is each site potential is required to be super-Gaussian (Palmer et al., 2006).

Recently scalable approximate solvers for local variational bounding procedures have been developed—see Seeger and Nickisch (2011b) for a review. These methods make use of a number of algorithmic relaxations to reduce the computational burden of local bound optimisation. First, double loop algorithms are employed that reduce the number of times that $\log\det(\mathbf{A})$, see

Section 5.1.1, and its derivative needs to be computed. Second, these algorithms use approximate methods to evaluate the marginal variances that are required to drive local variational bound optimisation. Marginal variances are approximated either by constructing low rank factorisations of $\mathbf{A}$ using iterative Lanczos methods or by perturb and MAP sampling methods (Papandreou and Yuille, 2010; Seeger, 2010; Ko and Seeger, 2012). Both of these approximations can greatly increase the speed of inference and the size of problems to which local procedures can be applied. Unfortunately, these relaxations are not without consequence regarding the quality of approximate inference. For example, the $\log\det(\mathbf{A})$ term is no longer exactly computed and a lower-bound on $\log Z$ is no longer maintained—only an estimate of $\log Z$ is provided. Lanczos approximated marginal variances are often found to be strongly underestimated and bound values strongly overestimated. Whilst the scaling properties are in general problem and user dependent, roughly speaking, these relaxations reduce the computational complexity to scaling $O\left(KD^2\right)$ where $K$ is the rank of the approximate covariance factorisation.

### 5.2.3 GAUSSIAN EXPECTATION PROPAGATION

Gaussian expectation propagation methods seek to approximate the target density by sequentially matching moments between marginals of the variational Gaussian distribution and a density constructed from the Gaussian approximation and individual site potentials (Minka, 2001). Gaussian expectation propagation (G-EP), for problems of the form of Equation (2), is parameterised using $2N$ free variational parameters, updating each of which requires $N$ rank one $D \times D$ Cholesky updates and the solution of $N$ symmetric $D$-dimensional linear systems—thus scaling $O\left(ND^2\right)$ assuming $N > D$. Importantly, G-EP optimises neither a convex nor concave objective and is not guaranteed to converge. Whilst G-EP does not require the site projection potentials to be either smooth or super-Gaussian, convergence issues can occur if they are multimodal or not log-concave.

Provably convergent double loop extensions to G-EP have been developed—see Opper and Winther (2005) and references therein for details. Typically these methods are slower than vanilla G-EP implementations. However, recent algorithmic developments have yielded significant speed ups over vanilla G-EP whilst maintaining the convergence guarantees (Seeger and Nickisch, 2011a). Importantly, however, these procedures require the exact solution of rank $D$ symmetric linear systems and thus scale $O\left(D^3\right)$.

### 5.2.4 G-KL

G-KL approximate inference methods require that each site projection potential has unbounded support on $\mathbb{R}$. Unlike Laplace procedures G-KL is applicable for models with non-differentiable site potentials. Unlike local variational bounding procedures G-KL does not require the site potentials to be super-Gaussian. In contrast to G-EP, which is known to suffer from convergence issues for non log-concave sites, G-KL procedures optimise a strict lower-bound and convergence is guaranteed for gradient ascent optimisation.

When $\{\phi_n\}_{n=1}^N$ are log-concave G-KL bound optimisation is a concave problem and we are guaranteed to converge to the global optimum of the G-KL bound. Local bounding methods have also been shown to be concave problems in this setting (Nickisch and Seeger, 2009). However, as we have shown in Section 5.1, the optimal G-KL bound to $\log Z$ is provably tighter than the local variational bound.

Exact implementations of G-KL approximate inference require storing and optimising over $\frac{1}{2}D(D+3)$ parameters to specify the Gaussian mean and covariance. Often the number of G-KL parameters is greater than that for Laplace, G-EP or local variational bounding methods. However, the computations required by G-KL methods scale similarly to these other Gaussian approximation methods. Empirically, as we show in Section 6, G-KL approximate inference is seen to have comparable convergence speeds to local bounding methods and G-EP.

Importantly, G-KL procedures can be made scalable by using constrained parameterisations of covariance that do not require making a priori factorisation assumptions for the approximate posterior density. Scalable covariance decompositions for G-KL inference maintain a strict lower-bound on $\log Z$ whereas approximate local bound optimisers do not. G-EP, being a fixed point procedure, has been shown to be unstable when using low-rank covariance approximations and appears constrained to scale $O\left(ND^2\right)$ (Seeger and Nickisch, 2011a).

## 6. Applications

In this section we present results obtained from applying Gaussian KL approximate inference methods to three popular machine learning models. In Section 6.1 we compare deterministic Gaussian approximate inference methods in robust Gaussian process regression models. In Section 6.2 we compare the performance of the constrained parameterisations of G-KL covariance that we presented in Section 4.1.3 in large scale Bayesian logistic regression models. In Section 6.3 we compare Gaussian approximate inference methods to drive sequential experimental design procedures in Bayesian sparse linear models.

### 6.1 Robust Gaussian Process Regression

Gaussian Processes (GP) are a popular non-parametric approach to supervised learning problems, see Rasmussen and Williams (2006) for a thorough introduction, for which inference falls into the general form of Equation (2). Excluding limited special cases, computing $Z$ and evaluating the posterior density, necessary to make predictions and set hyperparameters, is analytically intractable.

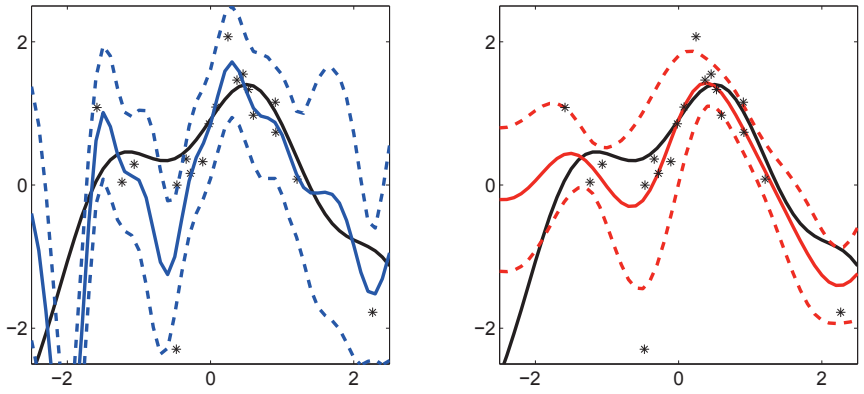
The supervised learning model for fully observed covariates $\mathbf{X} \in \mathbb{R}^{N\times D}$ and corresponding dependent variables $\mathbf{y} \in \mathbb{R}^N$ is specified by the GP prior on the latent function values $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and the likelihood $p(\mathbf{y}|\mathbf{w})$. The GP prior moments are constructed by the GP covariance and mean functions which take the covariates $\mathbf{X}$ and a vector of hyperparameters $\boldsymbol{\theta}$ as arguments. The posterior on the latent function values, $\mathbf{w}$, is given by

$$p(\mathbf{w}|\mathbf{y},\mathbf{X},\boldsymbol{\theta}) = \frac{1}{Z} p(\mathbf{y}|\mathbf{w}) \mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\boldsymbol{\Sigma}).$$

The likelihood factorises over data instances, $p(\mathbf{y}|\mathbf{w}) = \prod_{n=1}^{N} \phi(w_n)$, thus the GP posterior is of the form of Equation (1) with site projection potentials of the form of Equation (5).

#### 6.1.1 GP REGRESSION

For GP regression models the likelihood is most commonly Gaussian distributed, equivalent to assuming zero mean additive Gaussian noise. This assumption leads to analytically tractable, indeed Gaussian, forms for the posterior. However, Gaussian additive noise is a strong assumption to make, and is often not corroborated by real world data. Gaussian distributions have thin tales—the density




(a) Gaussian likelihood (b) Student's $t$ likelihood

Figure 5: Gaussian process regression with a squared exponential covariance function and (a) a Gaussian or (b) a Student's $t$ likelihood. Covariance hyperparameters are optimised for a training data set with outliers. Latent function posterior mean (solid) and $\pm 1$ standard deviation (dashed) values are plotted in blue (a) and red (b). The data generating function is plotted in black. The Student's $t$ model makes more conservative interpolated predictions whilst the Gaussian model appears to over-fit the data.

function rapidly tends to zero for values far from the mean—see Figure 6. Outliers in the training set then do not have to be too extreme to negatively affect test set predictive accuracy. This effect can be especially severe for GP models that have the flexibility to incorporate training set outliers to areas of high likelihood—essentially over-fitting the data.

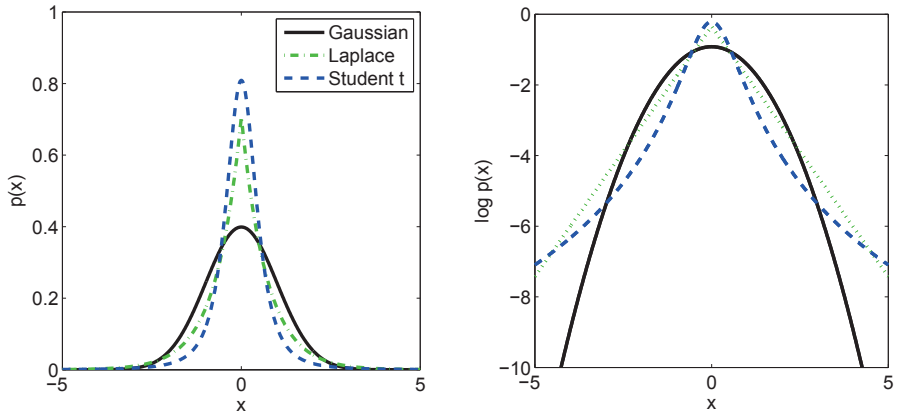
An example of GP regression applied to a data set with outliers is presented in figure 5(a). In this figure a GP prior with squared exponential covariance function coupled with a Gaussian likelihood over-fits the training data and the resulting predicted values differ significantly from the underlying data generating function.

One approach to prevent over-fitting is to use a likelihood that is robust to outliers. Heavy tailed likelihood densities are robust to outliers in that they do not penalise too heavily observations far from the latent function mean. Two distributions are often used in this context: the Laplace otherwise termed the double exponential, and the Student's $t$. The Laplace probability density function can be expressed as

$$p(y|\mu,\tau) = \frac{1}{2\tau} e^{-|y-\mu|/\tau},$$

where $\tau$ controls the variance of the random variable $y$ with $\mathrm{var}(y) = 2\tau^2$. The Student's $t$ probability density function can be written as

$$p(y|\mu,\nu,\sigma^2) = \frac{\Gamma\left(\frac{1}{2}(\nu+1)\right)}{\Gamma\left(\frac{1}{2}\nu\right)\sqrt{\pi\nu\sigma^2}} \left(1 + \frac{(y-\mu)^2}{\nu\sigma^2}\right)^{-\frac{\nu+1}{2}}$$




(a) (b)

Figure 6: Gaussian, Laplace and Student's $t$ densities with unit variance: (a) probability density functions and (b) log probability density functions. Laplace and Student's $t$ densities have stronger peaks and heavier tails than the Gaussian. Student's $t$ with d.o.f. $\nu = 2.5$ and scale $\sigma^2 = 0.2$, Laplace with $\tau = 1/\sqrt{2}$.

where $\nu \in \mathbb{R}^+$ is the degrees of freedom parameter, $\sigma \in \mathbb{R}^+$ the scale parameter, and $\text{var}(y) = \sigma^2\nu/(\nu-2)$ for $\nu > 2$. As the degrees of freedom parameter becomes increasingly large the Student's $t$ distribution converges to the Gaussian distribution. See Figure 6 for a comparison of the Student's $t$, Laplace and Gaussian density functions.

GP models with outlier robust likelihoods such as the Laplace or the Student's $t$ can yield significant improvements in test set accuracy versus Gaussian likelihood models (Vanhatalo et al., 2009; Jylanki et al., 2011; Opper and Archambeau, 2009). In figure 5(b) we model the same training data as in figure 5(a) but with a heavy tailed Student's $t$ likelihood, the resulting predictive values are more conservative and lie closer to the true data generating function than for the Gaussian likelihood model.

### 6.1.2 APPROXIMATE INFERENCE

Whilst Laplace and Student's $t$ likelihoods can successfully 'robustify' GP regression models to outliers they also render inference analytically intractable and approximate methods are required. In this section we compare G-KL approximate inference to other deterministic Gaussian approximate inference methods, namely: the Laplace approximation (Lap), local variational bounding (VB) and Gaussian expectation propagation (G-EP).

Each approximate inference method cannot be applied to each likelihood model. Since the Laplace likelihood is not differentiable everywhere Laplace approximate inference is not applicable. Since the Student's $t$ likelihood is not log-concave, indeed the posterior can be multi-modal, vanilla G-EP implementations are numerically unstable (Seeger et al., 2007). Recent work (Jylanki et al.,

| | | Gauss | Student's $t$ | | | Laplace | | |
|---|---|---|---|---|---|---|---|---|
| | | Exact | G-KL | VB | Lap | G-KL | VB | G-EP |
| **C. ST** | LML | $-15\pm2$ | $-75\pm2$ | $-240\pm21$ | $-7\pm1$ | $8\pm5$ | $2\pm2$ | $--\pm--$ |
| | MSE | $1.15\pm0.2$ | $1.6\pm0.2$ | $23.8\pm4$ | $2.2\pm0.4$ | $1.3\pm1.1$ | $1.2\pm1.0$ | $--\pm--$ |
| | TLP | $0.79\pm0.10$ | $0.73\pm0.05$ | $-0.65\pm0.06$ | $0.41\pm0.03$ | $0.97\pm0.06$ | $0.91\pm0.05$ | $--\pm--$ |
| **Friedman** | LML | $70\pm6$ | $-159\pm7$ | $-578\pm34$ | $-97\pm4$ | $-69\pm6$ | $-73\pm8$ | $--\pm--$ |
| | MSE | $10\pm3$ | $5\pm1$ | $17\pm2$ | $13\pm1$ | $5\pm1$ | $3\pm1$ | $--\pm--$ |
| | TLP | $-0.26\pm0.09$ | $0.12\pm0.09$ | $-0.54\pm0.06$ | $-0.65\pm0.06$ | $0.07\pm0.09$ | $0.25\pm0.11$ | $--\pm--$ |
| **Neal** | LML | $39\pm10$ | $-171\pm14$ | $-962\pm1$ | $-21\pm15$ | $-26\pm9$ | $-27\pm8$ | $-14\pm7$ |
| | MSE | $1.7\pm0.6$ | $2.9\pm1.1$ | $4.4\pm1.3$ | $0.9\pm0.5$ | $0.9\pm0.4$ | $0.9\pm0.4$ | $0.9\pm0.5$ |
| | TLP | $0.22\pm0.12$ | $0.88\pm0.03$ | $0.36\pm0.02$ | $0.67\pm0.08$ | $0.86\pm0.04$ | $1.13\pm0.02$ | $0.91\pm0.04$ |
| **Boston** | LML | $51\pm3$ | $-133\pm13$ | $-551\pm37$ | $-53\pm3$ | $-60\pm3$ | $-61\pm3$ | $-53\pm4$ |
| | MSE | $26\pm1$ | $25\pm2$ | $26\pm1$ | $23\pm2$ | $25\pm2$ | $26\pm1$ | $22\pm1$ |
| | TLP | $-0.74\pm0.07$ | $-0.44\pm0.03$ | $-0.58\pm0.03$ | $-0.44\pm0.03$ | $-0.52\pm0.06$ | $-0.51\pm0.02$ | $-0.46\pm0.03$ |

Table 1: Gaussian process regression results for different (approximate) inference procedures, likelihood models and data sets. First column section: Gaussian likelihood results with exact inference. Second column section: Student's $t$ likelihood results with G-KL, local variational bounding (VB) and Laplace (Lap) approximate inference. Third column section: Laplace likelihood results with G-KL, VB and Gaussian expectation propagation (G-EP) approximate inference. Each row presents the (approximate or lower-bound) log marginal likelihood (LML), test set mean squared error (MSE), or approximate test set log probability (TLP) values obtained by data set. Table values are the mean and standard error of the values obtained over the 10 random partitions of the data.

2011) has alleviated some of G-EP's convergence issues for Student's $t$ GP regression, however, these extensions are beyond the scope of this work.

Local variational bounding and G-KL procedures are applied to both likelihood models. For local variational bounding, both the Laplace and Student's $t$ densities are super-Gaussian and thus tight exponentiated quadratic lower-bounds exist—see Seeger and Nickisch (2010) for the precise forms that are employed in these experiments. Laplace, local variational bounding and G-EP results are obtained using the `GPML` toolbox (Rasmussen and Nickisch, 2010).[4] G-KL approximate inference is straightforward, for the G-KL approximate posterior $q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})$ the likelihood's contribution to the bound is

$$\langle \log p(\mathbf{y}|\mathbf{w}) \rangle_{q(\mathbf{w})} = \sum_n \left\langle \log \phi_n(m_n + z\sqrt{S_{nn}}) \right\rangle_{\mathcal{N}(z|0,1)}.$$

The equation above is equivalent to Equation (6) with $\mathbf{h}_n = \mathbf{e}_n$ the unit norm basis vector and $\phi_n$ the likelihood of the $n^{th}$ data point. The expectations for the Laplace likelihood site potentials have simple analytic forms—see Appendix B.2.1. The expectations for the Student's $t$ site potentials are evaluated numerically. All other terms in the G-KL bound have simple analytic forms and computations that scale $\leq O\left(D^3\right)$. G-KL results are obtained, as for all other results in this paper, using the `vgai` Matlab package—see Section 8. For the Laplace likelihood model, which is log-concave, Hessian free Newton methods were used to optimise the G-KL bound. For the Student's $t$ likelihood, which is not log-concave, LBFGS was used to optimise the G-KL bound.

4. The `GPML` toolbox can be downloaded from `www.gaussianprocess.org`.

### 6.1.3 EXPERIMENTAL SETUP

We consider GP regression with training data $\mathcal{D} = \{(y_n, \mathbf{x}_n)\}_{n=1}^N$ for covariates $\mathbf{x}_n \in \mathbb{R}^D$ and dependent variables $y_n \in \mathbb{R}$. We assume a zero mean Gaussian process prior on the latent function values, $\mathbf{w} = [w_1, ..., w_N]^\mathsf{T} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. The covariance, $\boldsymbol{\Sigma}$, is constructed as the sum of the squared exponential kernel and the independent white noise kernel,

$$\Sigma_{mn} = k(\mathbf{x}_m, \mathbf{x}_n, \boldsymbol{\theta}) = \sigma_{se}^2 e^{-\sum_d (x_{nd} - x_{md})^2 / l_d^2} + \gamma^2 \delta(n, m),$$

where $x_{nd}$ refers to the $d^{th}$ element of the $n^{th}$ covariate, $\sigma_{se}^2$ is the 'signal variance' hyperparameter, $l_d$ the squared exponential 'length scale' hyperparameter, and $\gamma$ the independent white noise hyperparameter (above $\delta(x, y)$ is the Kronecker delta such that $\delta(n, m) = 1$ if $n = m$ and 0 otherwise). Covariance hyperparameters are collected in the vector $\boldsymbol{\theta}$.

We follow the evidence maximisation or maximum likelihood two (ML-II) procedure to estimate the covariance hyperparameters, that is we set covariance hyperparameters to maximise $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$. Since $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ cannot be evaluated exactly we use the approximated values offered by each of the approximate inference methods. Covariance hyperparameters are optimised numerically using nonlinear conjugate gradients. The marginal likelihood, $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$, is not unimodal and we are liable to converge to a local optimum regardless of which inference method is used. All methods were initialised with the same hyperparameter setting. Hyperparameter derivatives for the G-KL bound are presented in Appendix F.2.

Likelihood hyperparameters were selected to maximise the log predicted probability scores on a held out validation data set. Simultaneous likelihood and covariance ML-II hyperparameter optimisation for the Student's $t$ and Laplace likelihoods yielded poor test set performance regardless of the approximate inference method used (as has been previously reported for Student's $t$ likelihoods in other experiments (Vanhatalo et al., 2009; Jylanki et al., 2011). For the Student's $t$ likelihood model the d.o.f. parameter was fixed with $\nu = 3$.

Results were obtained for the four approximate inference procedures on the four data sets using both the Laplace and the Student's $t$ likelihoods. Two UCI data sets were used:[5] Boston housing and Concrete Slump Test. And two synthetic data sets: Friedman[6] and Neal.[7] Each experiment was repeated over 10 randomly assigned training, validation and test set partitions. The size of each data set is as follows: Concrete Slump Test $D = 9$, $N_{trn} = 50$, $N_{val} = 25$, $N_{tst} = 28$; Boston $D = 13$, $N_{trn} = 100$, $N_{val} = 100$, $N_{tst} = 306$; Friedman $D = 10$, $N_{trn} = 100$, $N_{val} = 100$, $N_{tst} = 100$; Neal $D = 1$, $N_{trn} = 100$, $N_{val} = 100$, $N_{tst} = 100$. Each partition of the data was normalised using the mean and standard deviation statistics of the training data.

To asses the validity of the Student's $t$ and Laplace likelihoods we also implemented GP regression with a Gaussian likelihood and exact inference.

### 6.1.4 RESULTS

Results are presented in Table 1. Approximate log marginal likelihood (LML), test set mean squared error (MSE) and approximate test set log probability (TLP) mean and standard error values obtained over the 10 partitions of the data are provided. It is important to stress that the TLP values are approximate values for all methods, obtained by summing the approximate log probability of each

5. UCI data sets can be downloaded from `archive.ics.uci.edu/ml/datasets/`.
6. The Friedman data set is constructed as described in Kuss (2006) §5.6.1. and Friedman (1991).
7. The Neal data set is constructed as described in Neal (1997) §7.

test point using the surrogate score presented in Appendix F.1. For G-KL and VB procedures the TLP values are not lower-bounds.

The results confirm the utility of heavy tailed likelihoods for GP regression models. Test set predictive accuracy scores are higher with robust likelihoods and approximate inference methods than with a Gaussian likelihood and exact inference. This is displayed in the lower MSE error and higher TLP scores of the best performing robust likelihood results than for the Gaussian likelihood. Exact inference for the Gaussian likelihood model achieves the greatest LML in all problems except the Concrete Slump Test data. That exact inference with a Gaussian likelihood achieves the strongest LML and weak test set scores implies the ML-II procedure is over-fitting the training data with this likelihood model.

For the Student's $t$ likelihood the performance of each approximate inference method varied significantly. VB results were uniformly the weakest. We conjecture this is an artifact of the squared exponential local site bounds employed by the `gpml` toolbox poorly capturing the non log-concave potential functions mass. For Student's $t$ potentials improved VB performance has been reported by employing bounds that are composed of two terms on disjoint partitions of the domain (Seeger and Nickisch, 2011b), validating their efficacy in the context of Student's $t$ GP regression models is reserved for future work. For the test set metrics G-KL approximate inference achieves the strongest performance.

Broadly, the Laplace likelihood achieved the best results on all data sets. G-EP frequently did not converge for both the Friedman and Concrete Slump Test problems and so results are not presented. Unlike the Student's $t$ likelihood model, results are more consistent across approximate inference methods. G-KL achieves a narrow but consistently superior LML value to VB. Approximate test set predictive values are roughly the same for all inference methods with VB achieving a small advantage.

We reiterate that standard G-EP approximate inference, as implemented in the `GPML` toolbox, was used to obtain these results. The authors did not anticipate convergence issues for G-EP in the GP models considered—the Laplace likelihood model's log posterior is concave and the system has full rank. Power G-EP, as proposed in Minka (2004), has previously been shown to have robust convergence for under determined linear models with Laplace potentials (Seeger, 2008). Similarly, we expect that power G-EP would also exhibit robust convergence in GP models with Laplace likelihoods. Verifying this experimentally and assessing the performance of power G-EP approximate inference in noise robust GP regression models is left for future work.

The G-KL LML uniformly dominates the VB values. This is theoretically guaranteed for a model with fixed hyperparameters and log-concave site potentials, see Section 5.1.2 and Section 3.2. However, the G-KL bound is seen to dominate the local bound even when these conditions are not satisfied. The results show that both G-KL bound optimisation and G-KL hyperparameter optimisation is numerically stable. G-KL approximate inference appears more robust than G-EP and VB—G-KL hyperparameter optimisation always converged, often to a better local optima.

### 6.1.5 SUMMARY

The results confirm that the G-KL procedure as a sensible route for approximate inference in GP models with non-conjugate likelihoods. The G-KL procedure is generally applicable in this setting and easy to implement for new likelihood models. Indeed, all that is required to implement G-KL approximate inference for a GP regression model is the pointwise evaluation of the univariate

likelihood function $p(y_n|w_n)$. Furthermore, we have seen that G-KL optimisation is numerically robust, in all the experiments G-KL converged and achieved strong performance.

### 6.2 Bayesian Logistic Regression

In this section we examine the relative performance, in terms of speed and accuracy of inference, of each of the constrained G-KL covariance decompositions presented in Section 4.1.3. As a bench mark, we also compare G-KL approximate inference results to scalable approximate VB methods with marginal variances approximated using iterative Lanczos methods (Seeger and Nickisch, 2011b). Our aim is not make a comparison of deterministic approximate inference methods for Bayesian logistic regression models, see Nickisch and Rasmussen (2008) to that end, but to investigate the time accuracy trade-offs of each of the constrained G-KL covariance parameterisations.

Given a data set, $\mathcal{D} = \{(y_n, \mathbf{x}_n), n = 1, \ldots, N\}$ with class labels $y_n \in \{-1, 1\}$ and covariates $\mathbf{x}_n \in \mathbb{R}^D$, Bayesian logistic regression models the class conditional distribution using $p(y = 1|\mathbf{w}, \mathbf{x}) = \sigma\left(\mathbf{w}^\mathsf{T}\mathbf{x}\right)$, with $\sigma(x) := 1/(1+e^{-x})$ the logistic sigmoid function and $\mathbf{w} \in \mathbb{R}^D$ a vector of parameters. Under a Gaussian prior, $\mathcal{N}(\mathbf{w}|\mathbf{0}, \boldsymbol{\Sigma})$, the posterior is given by

$$p(\mathbf{w}|\mathcal{D}) = \frac{1}{Z}\mathcal{N}(\mathbf{w}|\mathbf{0}, \boldsymbol{\Sigma}) \prod_{n=1}^{N} \sigma\left(y_n \mathbf{w}^\mathsf{T}\mathbf{x}_n\right). \tag{15}$$

Where we have used the symmetry property of the logistic sigmoid such that $p(y = -1|\mathbf{w}, \mathbf{x}) = 1 - p(y = 1|\mathbf{w}, \mathbf{x}) = \sigma\left(-\mathbf{w}^\mathsf{T}\mathbf{x}\right)$. The expression above is of the form of Equation (2) with log-concave site projection potentials $\phi_n(x) = \sigma(x)$ and $\mathbf{h}_n = y_n\mathbf{x}_n$.

#### 6.2.1 EXPERIMENTAL SETUP

We synthetically generate the data sets. The data generating parameter vector $\mathbf{w}^{tr} \in \mathbb{R}^D$ is sampled from a factorising standard normal $\mathbf{w}^{tr} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The covariates, $\{\mathbf{x}_n\}_{n=1}^N$, are generated by first sampling an independent standard normal, then linearly transforming these vectors to impose correlation between some of the dimensions, and finally the data is renormalised so that each dimension has unit variance. The linear transformation matrix we use to impose correlation between covariates is a sparse square matrix generated as the sum of the identity matrix and a sparse matrix with one element from each row sampled from a standard normal. Class labels $y_n \in \{1, -1\}$ are sampled from the likelihood $p(y_n = 1|\mathbf{w}, \mathbf{x}_n) = \sigma(\mathbf{w}^\mathsf{T}\mathbf{x}_n)$. The inferential model's prior and likelihood distributions are set to match the data generating process.

Results are obtained for a range of data set dimensions: $D = 250, 500, 1000$ and $N = \frac{1}{2}D, D, 5D$. We also vary the size of the constrained covariance parameterisations, which is reported as $K$ in the result tables. For chevron Cholesky $K$ refers to the number of non-diagonal rows of $\mathbf{C}$. For subspace Cholesky $K$ is the dimensionality of the subspace. For banded Cholesky $K$ refers to the band width of the parameterisation. For the factor analysis (FA) parameterisation $K$ refers to the number of factor loading vectors. For local variational bounding (VB) approximate inference $K$ refers to the number of Lanczos vectors used to update the variational parameters. The parameter $K$ is varied as a function of the parameter vector dimensionality with $K = 0.05 \times D$ and $K = 0.1 \times D$.

Since the G-KL bound is strongly concave we performed G-KL bound optimisation using Hessian free Newton methods for all the Cholesky parameterised covariance experiments. G-KL bound optimisation was terminated when the largest absolute value of the gradient vector was less than $10^{-3}$. For subspace Cholesky we iterated between optimising the subspace parameters $\{\mathbf{m}, \mathbf{C}, c\}$

| | | | $N_{trn}=250$ | | $N_{trn}=500$ | | $N_{trn}=2500$ | |
|---|---|---|---|---|---|---|---|---|
| | | | $K=25$ | $K=50$ | $K=25$ | $K=50$ | $K=25$ | $K=50$ |
| Time (s) | G-KL | Chev | 0.49±0.02 | 0.69±0.08 | 1.25±0.04 | 1.36±0.04 | 16.50±0.89 | 17.31±0.82 |
| | | Band | 0.96±0.02 | 1.37±0.02 | 2.25±0.10 | 4.06±0.29 | 24.31±0.96 | 29.60±1.18 |
| | | Sub | 0.73±0.01 | 0.93±0.03 | 1.41±0.03 | 1.93±0.04 | 11.89±0.54 | 15.26±1.02 |
| | | FA | 2.05±0.26 | 2.29±0.21 | 2.92±0.17 | 3.47±0.17 | 20.06±1.51 | 22.69±2.70 |
| | VB | | 0.37±0.00 | 0.47±0.01 | 0.46±0.02 | 0.52±0.00 | 1.56±0.03 | 1.85±0.01 |
| $\tilde{\mathcal{B}}$ | G-KL | Chev | −1.19±0.01 | −1.15±0.01 | −0.93±0.01 | −0.91±0.01 | −0.42±0.00 | −0.41±0.00 |
| | | Band | −1.15±0.01 | −1.09±0.01 | −0.92±0.01 | −0.88±0.01 | −0.42±0.00 | −0.41±0.00 |
| | | Sub | −3.08±0.02 | −2.20±0.01 | −1.90±0.01 | −1.46±0.01 | −0.62±0.00 | −0.54±0.00 |
| | | FA | −1.19±0.01 | −1.17±0.01 | −0.93±0.01 | −0.91±0.01 | −0.41±0.00 | −0.40±0.00 |
| | VB | | −±− | −±− | −±− | −±− | −±− | −±− |
| $\|\mathbf{w}-\mathbf{w}_{tr}\|_2/D$ | G-KL | Chev | 0.88±0.00 | 0.87±0.00 | 0.84±0.00 | 0.84±0.00 | 0.64±0.00 | 0.64±0.00 |
| | | Band | 0.87±0.00 | 0.87±0.00 | 0.84±0.00 | 0.84±0.00 | 0.64±0.00 | 0.64±0.00 |
| | | Sub | 0.88±0.00 | 0.87±0.01 | 0.87±0.00 | 0.86±0.00 | 0.71±0.00 | 0.70±0.00 |
| | | FA | 0.88±0.00 | 0.87±0.01 | 0.84±0.00 | 0.84±0.00 | 0.64±0.00 | 0.64±0.00 |
| | VB | | 0.90±0.00 | 0.89±0.00 | 0.89±0.00 | 0.88±0.00 | 0.72±0.00 | 0.72±0.00 |
| $\log p(\mathbf{y}^*\|\mathbf{X}^*)/N_{tst}$ | G-KL | Chev | −0.58±0.01 | −0.58±0.01 | −0.50±0.01 | −0.49±0.01 | −0.18±0.00 | −0.18±0.00 |
| | | Band | −0.58±0.01 | −0.57±0.01 | −0.50±0.01 | −0.49±0.01 | −0.18±0.00 | −0.18±0.00 |
| | | Sub | −0.72±0.02 | −0.65±0.02 | −0.63±0.01 | −0.59±0.01 | −0.20±0.00 | −0.20±0.00 |
| | | FA | −0.58±0.01 | −0.58±0.01 | −0.51±0.01 | −0.50±0.01 | −0.18±0.00 | −0.18±0.00 |
| | VB | | −0.75±0.02 | −0.77±0.02 | −0.63±0.01 | −0.64±0.01 | −0.20±0.00 | −0.20±0.00 |

Table 2: Synthetic Bayesian logistic regression results for a model with unit variance Gaussian prior $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ with $\dim(\mathbf{w}) = 500$, likelihood $p(\mathbf{y}|\mathbf{w}, \mathbf{X}) = \prod_{n=1}^{N_{trn}} \sigma(y_n \mathbf{w}^\mathsf{T} \mathbf{x}_n)$, class labels $y_n \in \{+1, -1\}$ and $N_{tst} = 5000$ test points. G-KL results obtained using chevron Cholesky (Chev), banded Cholesky (Band), subspace Cholesky (Sub) and factor analysis (FA) constrained parameterisations of covariance. Approximate local variational bounding (VB) results are obtained using low-rank factorisations of covariance computed using iterative Lanczos methods. Parameter $K$ denotes the size of the constrained covariance parameterisation.

and updating the subspace basis vectors $\mathbf{E}$ each five times. The subspace vectors were updated using the fixed point iteration with the Lanczos approximation (see Appendix B.4.3 for details). For the FA parameterisation the G-KL bound is not concave so we use LBFGS to perform gradient ascent. All other `minFunc` options were set to default values.

VB approximate inference is achieved using the `glm-ie 1.4` package (Nickisch, 2012). VB inner loop optimisation used nonlinear conjugate gradients with at most 50 iterations. The maximum number of VB outer loop iterations was set to 10. All other VB `glm-ie` optimisation settings were set to default values. All results for these experiments were obtained using Matlab 2011a on a Intel E5450 3Ghz machine with 8 cores and 64GB of RAM.

### 6.2.2 RESULTS

Results for $D = 500$ are presented in Table 2. For reasons of space, results for $D = 250$ and $D = 1000$ are presented in Table 3 and Table 4 in Appendix G. The tables present average and standard error scores obtained from 10 synthetically generated data sets.

The average convergence time and standard errors of each of the methods is presented in the first row section of the result tables. In the smaller problems considered, the best G-KL times were

achieved by the chevron Cholesky covariance followed by the banded, the subspace and the FA parameterisations in that order.

The recorded banded Cholesky convergence times are seen to scale super-linearly with $K$. These results are a consequence of the implementation. Whilst chevron and banded parameterisations both scale $O(NDK)$ they access and compute different elements of the data and Cholesky matrices. The chevron gradients can be computed using standard matrix multiplications for which Matlab is highly optimised. The banded parameterisation needs to access matrix elements in a manner not standard to Matlab and so is much slower. This implementational artifact, despite a Matlab mex C implementation, could not be entirely eliminated.

VB and chevron G-KL achieved broadly similar convergence times for the $N \leq D$ and $D \leq 500$ experiments with VB faster in the larger $D$ experiments. VB is significantly faster than G-KL methods for the $N = 5 \times D$ experiments, this is a consequence of the double-loop structure of the VB implementation. Whilst the subspace G-KL method is significantly slower in the smaller problems when $D = 1000$ it is the fastest G-KL method, beating VB in problems where $N \leq D$.

In the result tables, the bound values are normalised by the size of the training set, with $\tilde{\mathcal{B}} = \mathcal{B}/N_{trn}$, to make comparisons across models easier. As the training set size increases the normalised bound value increases, presumably reflecting the fact that the posterior tends to a Gaussian in the limit of large data. Furthermore, the difference in bound values between the parameterisations become smaller as the size of the training set increases.

The G-KL banded covariance parameterisation achieves the strongest bound value with the chevron and factor analysis parameterisations a close second place. The subspace bound values are comparatively poor. This is not unexpected since the subspace parameterisation has a single parameter (denoted $c$ in Section 4.1.3) that specifies the variance in all directions orthogonal to the subspace vectors $\mathbf{E}$. It is known that the density $q$ that minimises $\mathrm{KL}(q|\,p)$ tends to seek out the modes of $p$ and avoid those regions of parameter space where $p$ is close to zero. Therefore the parameter $c$ will tend to the smallest value of the variance of $p$ in the directions orthogonal to the subspace vectors, the resulting G-KL bound value will therefore be greatly underestimated. The approximate VB method does not provide a lower bound when marginal variances are approximated using low-rank methods and therefore values are not reported in the result tables.

Since these results are obtained from data sets sampled from densities with known parameters we can directly asses the accuracy of the posterior parameter estimate against the ground truth. The posterior mean minimises the $\ell^2$ loss $\|\mathbf{w}^{tr} - \mathbf{w}\|_2$. Thus, in the third row section of the results table, we report the average error $\|\mathbf{w}^{tr} - \mathbf{m}\|_2$ where $\mathbf{m}$ is the mean of the Gaussian posterior approximation $q(\mathbf{w}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$. To make comparisons easier, the $\ell^2$ errors are normalised by the dimensionality of the respective models $D$. The results show that the the G-KL mean is broadly invariant to the G-KL covariance parameterisations used. VB results are noticeably poorer than the G-KL methods.

Approximate test set log predictive probabilities are presented in the fourth row section of the result tables. This metric is arguably the best suited to measure the global accuracy of the posterior approximations since it is an expectation over the entire support of the approximate posterior (MacKay and Oldfield, 1995). The values reported in the table are approximated using $\log p(\mathbf{y}^*|\mathbf{X}^*) \approx \sum_n \log \left\langle p(y_n^* \mathbf{w}^{\mathsf{T}} \mathbf{x}_n^*) \right\rangle_{q(\mathbf{w})}$. The values presented are normalised by the size of the test set where $N_{tst} = 10 \times D$ in all experiments. The results show that chevron, banded and FA parameterisations achieve the best, and broadly similar, performance. Test set predictive accuracy increases for all methods as a function of the training set size. Subspace G-KL and approximate VB achieve broadly similar and noticeably weaker performance than the other methods.

#### 6.2.3 SUMMARY

The results support the use of the constrained Cholesky covariance parameterisations to drive scalable G-KL approximate inference procedures. Whilst neither the banded nor the chevron Cholesky parameterisations are invariant to permutations of the index set they both achieved the strongest bound values and test set performance. Unfortunately, due to implementational issues, the banded Cholesky parameterisation gradients are slow to compute resulting in slow recorded convergence times. The non-concavity of the factor analysis parameterised covariance resulted in slower recorded convergence times than the concave models. Whilst the subspace G-KL parameterisation had poorer performance in the smaller problems it broadly matched or outperformed the approximate VB method in the largest problems.

### 6.3 Bayesian Sparse Linear Models

Many problems in machine learning and statistics can be addressed by linear models with sparsity inducing prior distributions. Examples include, feature selection in regression problems (Wipf, 2004), source separation (Girolami, 2001), denoising or deblurring problems (Fergus et al.), and signal reconstruction from a set of under-determined observations (Seeger and Nickisch, 2008). In all of these cases, the prior results in a posteriori parameter estimates that are biased towards sparse solutions. For feature selection problems this assumption can be useful if we believe that only a small subset of the features are necessary to model the data. Using an informative prior is essential in the case of under-determined linear models where there are more sources than signals, in which case hyper-planes in parameter space have equiprobable likelihoods and priors are needed to constrain the space of possible solutions.

Figure 7 depicts the posteriors resulting from an under-determined linear model for a selection of different priors. Since the Laplace prior is log-concave the posterior is unimodal and log-concave. For non log-concave priors the resulting posterior can be multimodal—for instance when $p(\mathbf{w})$ is the Student's $t$ distribution or the sparsity promoting distribution composed from a mixture of Gaussians.

In the case of signal reconstruction, deblurring and source separation sparse priors are used to encode some of the prior knowledge we have about the source signal we wish to recover. Natural images for instance are known to have sparse statistics over a range of linear filters (an example filter being the difference in intensities of neighbouring pixels) (Olshausen and Field, 1996). Sparse priors that encode this knowledge about the statistics of natural images then bias estimates towards settings that share this statistical similarity.

In this section we consider Bayesian sequential experimental design (SED) for the sparse linear model. At each stage of the SED process we approximate the posterior density of the model parameters and then use the approximate posterior to greedily select new, maximally informative measurements. The probabilistic model and experimental design procedure are described in Section 6.3.1 and Section 6.3.2. In Section 6.3.3 we compare approximate inference methods on a small scale artificial SED problem. In Section 6.3.4 we compare G-KL and approximate local variational bounding methods for SED on a $64 \times 64 = 4,096$ pixel natural image problem. Our approach follows that laid out in Seeger and Nickisch (2008), Seeger (2009) and Seeger and Nickisch (2011b).

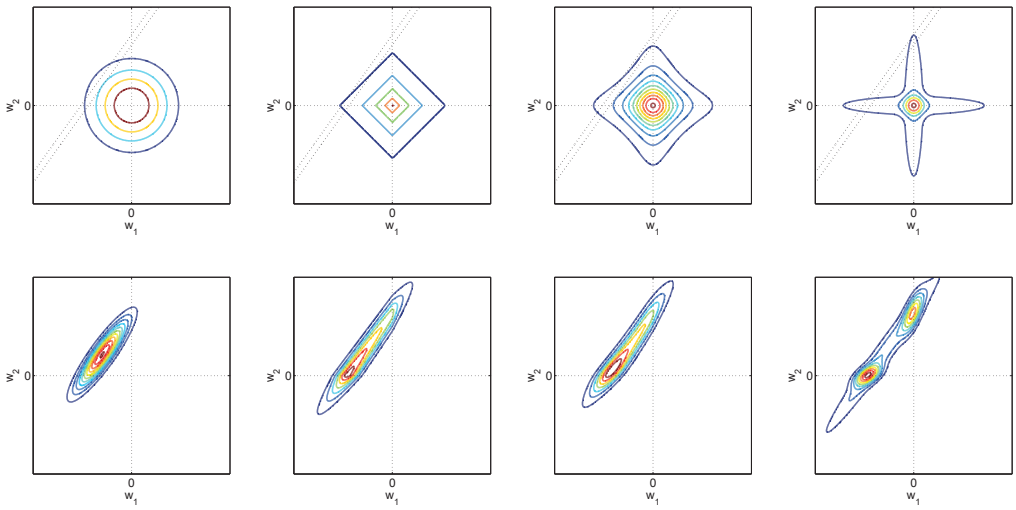



Figure 7: Isocontours of a linear model's prior, likelihood and resulting posterior densities. The top row plots contours of the two dimensional prior (solid) and the Gaussian likelihood (dashed). The second row displays the contours of the posterior induced by the prior and likelihood above it. Column one - a Gaussian prior, column two - a Laplace prior, column three - a Student's $t$ prior and column four a spike and slab prior constructed as a product over dimensions of a two component Gaussian mixture.

### 6.3.1 PROBABILISTIC MODEL

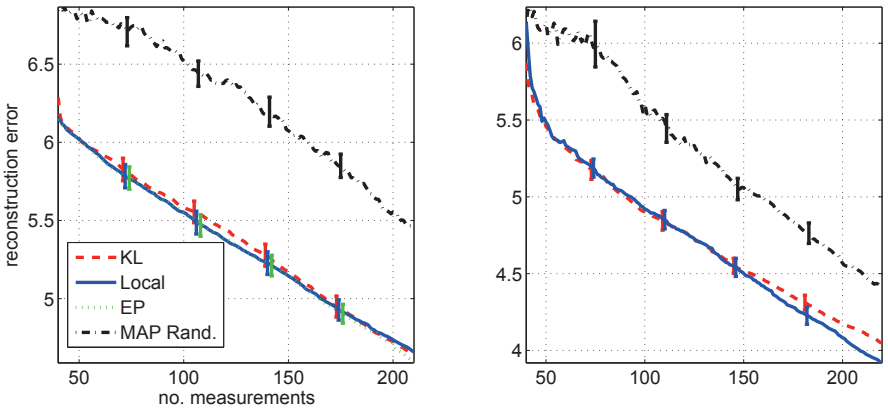
We observe noisy linear measurements $\mathbf{y} \in \mathbb{R}^N$ assumed to be drawn according to $\mathbf{y} = \mathbf{M}\mathbf{w} + \boldsymbol{\nu}$ where $\mathbf{M} \in \mathbb{R}^{N \times D}$ is the linear measurement matrix with $N \ll D$, $\boldsymbol{\nu} \sim \mathcal{N}\left(\mathbf{0}, \nu^2 \mathbf{I}\right)$ is additive Gaussian noise, and $\mathbf{w} \in \mathbb{R}^D$ is the signal that we wish to recover. A sparse prior, here we use either the Laplace or the Student's $t$, is placed on $\mathbf{s}$ the linear statistics of $\mathbf{w}$ such that $\mathbf{s} = \mathbf{B}\mathbf{w}$. The matrix $\mathbf{B} \in \mathbb{R}^{M \times D}$ is a collection of $M$ linear filters. By placing the prior directly on the statistics, $\mathbf{s}$, the posterior is proportional to the product of the Gaussian likelihood and the sparse prior potentials,

$$p(\mathbf{w}|\mathbf{M}, \mathbf{y}, \boldsymbol{\tau}, \nu^2) \propto \mathcal{N}\left(\mathbf{y}|\mathbf{M}\mathbf{w}, \nu^2 \mathbf{I}\right) p(\mathbf{s}), \quad \text{where} \quad \mathbf{s} = \mathbf{B}\mathbf{w}.$$

Since the priors are placed directly on the statistics $\mathbf{s}$ and not $\mathbf{w}$ they are not normalised densities with respect to $\mathbf{w}$, as a consequence $\mathcal{B}_{KL}(\mathbf{m}, \mathbf{S})$ is no longer a lower-bound to $\log Z$. However, since the normalisation constant of $p(\mathbf{s})$ is constant with respect to $\mathbf{w}$ ignoring this constant does not affect the G-KL approximation to the posterior density.

### 6.3.2 SEQUENTIAL EXPERIMENTAL DESIGN

SED for the sparse linear model described above is the problem of iteratively choosing which new measurement vectors, $\mathbf{M}^*$, to append to $\mathbf{M}$ so as to maximise subsequent estimation accuracy. Bayesian SED iterates between estimating the posterior density on $\mathbf{w}$, conditioned on current observations, and then using this density to select which new measurements to make. Following




(a) Laplace (b) Student's $t$

Figure 8: Sequential experimental design for the Bayesian sparse linear model with synthetic signals. Sparse signals, $\mathbf{w}$, are sampled from (a) a Laplace with $\tau = 0.2$ and (b) a Student's $t$ with $\nu = 3$, $\sigma^2 = 0.0267$.

Seeger and Nickisch (2011b) we use the information gain metric to decide which measurement vectors will be maximally informative. Information gain is defined as the difference in differential Shannon information of the posterior density before and after the inclusion of new measurements and their corresponding observations. For the linear model we consider, it is given by

$$I_{gain}(\mathbf{M}^*) = H\left[p(\mathbf{w}|\mathbf{M},\mathbf{y})\right] - H\left[p(\mathbf{w}|\mathbf{M},\mathbf{y},\mathbf{M}^*,\mathbf{y}^*)\right], \tag{16}$$

where $H\left[p(x)\right] := -\left\langle \log p(x) \right\rangle_{p(x)}$ is the Shannon differential entropy.

Since inference is not analytically tractable we cannot access either of the densities required by Equation (16). We can, however, obtain an approximation to the information gain by substituting in a Gaussian approximation to the posterior. Doing so with $p(\mathbf{w}|\mathbf{M},\mathbf{y}) \approx \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})$ we have $\left\langle \log p(\mathbf{w}|\mathbf{M},\mathbf{y}) \right\rangle \approx \frac{1}{2}\log\det(\mathbf{S}) + c$ with $c$ an additive constant. The second entropy is estimated by Gaussian conditioning on the joint approximate Gaussian density defined as $p(\mathbf{w},\mathbf{y}^*|\mathbf{y}) \propto \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})\,\mathcal{N}\left(\mathbf{y}^*|\mathbf{M}^*\mathbf{w},\nu^2\mathbf{I}\right)$.

The approximation to the information gain can then be written as

$$I_{gain}(\mathbf{M}^*) \approx \frac{1}{2}\log\det\left(\mathbf{M}^*\mathbf{S}\mathbf{M}^{*\mathsf{T}} + \nu^2\mathbf{I}\right) + c.$$

If we constrain the measurements to have unit norm $I_{gain}(\mathbf{M}^*)$ above will be maximised when the rows of $\mathbf{M}^*$ lie along the leading principal eigenvectors of the approximate posterior covariance $\mathbf{S}$. These eigenvectors are approximated in our experiments using iterative Lanczos methods.

### 6.3.3 SYNTHETIC SIGNALS

Initially we consider applying sequential experimental design to a sparse signal reconstruction problem using small scale synthetic signals. In this artificial set up we wish to recover some signal

$\mathbf{w}_{tr} \in \mathbb{R}^{512}$ from a set of noisy linear measurement $\mathbf{y} \in \mathbb{R}^m$ where $m \ll 512$. We initialised the experiments with $m_0 = 40$ random unit norm linear measurement vectors $\mathbf{M} \in \mathbb{R}^{m_0 \times 512}$.

In this setup we placed the sparse prior directly on $\mathbf{w}$ with $\mathbf{B} = \mathbf{I}$. Sparse signals, $\mathbf{w}_{tr}$, were sampled independently over dimensions from either the Laplace ($\mu = 0, \tau = 0.2$) or the Student's $t$ ($\nu = 3, \sigma^2 = 0.027$) densities. Noisy linear measurements were sampled from the source signals with $\mathbf{y} \sim \mathcal{N}\left(\mathbf{M}\mathbf{w}_{tr}, \nu^2\mathbf{I}\right)$ and $\nu^2 = 0.005$ throughout. Model priors and likelihoods were fixed to match the data generating densities.

For the Laplace generated signals we applied G-KL, local variational bounding (VB) and power G-EP ($\eta = 0.9$) approximate inference methods. G-EP and VB results were obtained using the publicly available `glm-ie` Matlab toolbox. Since the model is of sufficiently small dimensionality approximate covariance decompositions were not required. For the Student's $t$ generated signals only G-KL and VB approximate inference methods were applied since G-EP is unstable in this setting.

For the Laplace signals, when $D = 512$ and $N = 110$, inference takes 0.3 seconds for VB, 0.6 seconds for G-EP, and 1.6 seconds for G-KL.[8] For the Student's $t$ signals, again with $D = 512$ and $N = 110$, inference takes 0.3 seconds for VB and 6 seconds for G-KL. For Laplace signals, for which the G-KL bound is concave, gradient ascent was performed using a Hessian free Newton method with finite differences approximation for Hessian vector products—see (Nocedal and Wright, 2006, Chapter 7). For the Student's $t$ signals, for which the G-KL bound is not guaranteed to be concave or even unimodal, gradient ascent was performed using nonlinear scaled conjugate gradients. G-KL optimisation was terminated in both settings once the largest absolute value of the bound's gradient was less than 0.01. VB and G-EP were optimised for seven outer loop iterations after which no systematic improvement in the approximate $\log Z$ value was observed.

$\ell^2$ norm reconstruction error mean and standard error scores obtained over the 25 experiments conducted are presented in Figure 8. For the Laplace generated signals VB, G-EP and G-KL approximate inference procedures provide broadly the same reconstruction error performance. All sequentially designed procedures outperform MAP estimates with standard normal random measurements. The improved performance comes mainly in the first few iterations of the SED process with all methods achieving broadly similar iterative improvements in reconstruction error after that. For the Student's $t$ prior again VB and G-KL procedures obtain broadly the same performance with G-KL appearing to become slightly less effective towards the end of the experiment.

### 6.3.4 NATURAL IMAGES

We consider sequential experimental design for the problem of recovering natural images from a set of under-determined noisy linear measurements. This problem is modelled by placing priors on the statistics of natural images that are known to exhibit sparsity. These statistics can be captured by suitable linear projections of the image vector (formed by concatenating the pixel value columns of the image). For the results presented we employ two types of image filter known to exhibit sparse statistics in natural images: finite differences, the difference in intensity values of horizontally or vertically neighbouring pixels; and multi-scale orthonormal wavelet transforms, constructed using the Daubechies four wavelet—see Seeger and Nickisch (2011b) for further details. Both filters can be expressed as extremely sparse vectors, the set of which is collected in the matrix $\mathbf{B}$, giving $\mathbf{B} \in \mathbb{R}^{M \times D}$ where $M = 3 \times D$. Image filters were implemented using the `glm-ie` package. Laplace

8. Experiments were timed using Matlab R2009a on a 32 bit Intel Core 2 Quad 2.5 GHz processor.

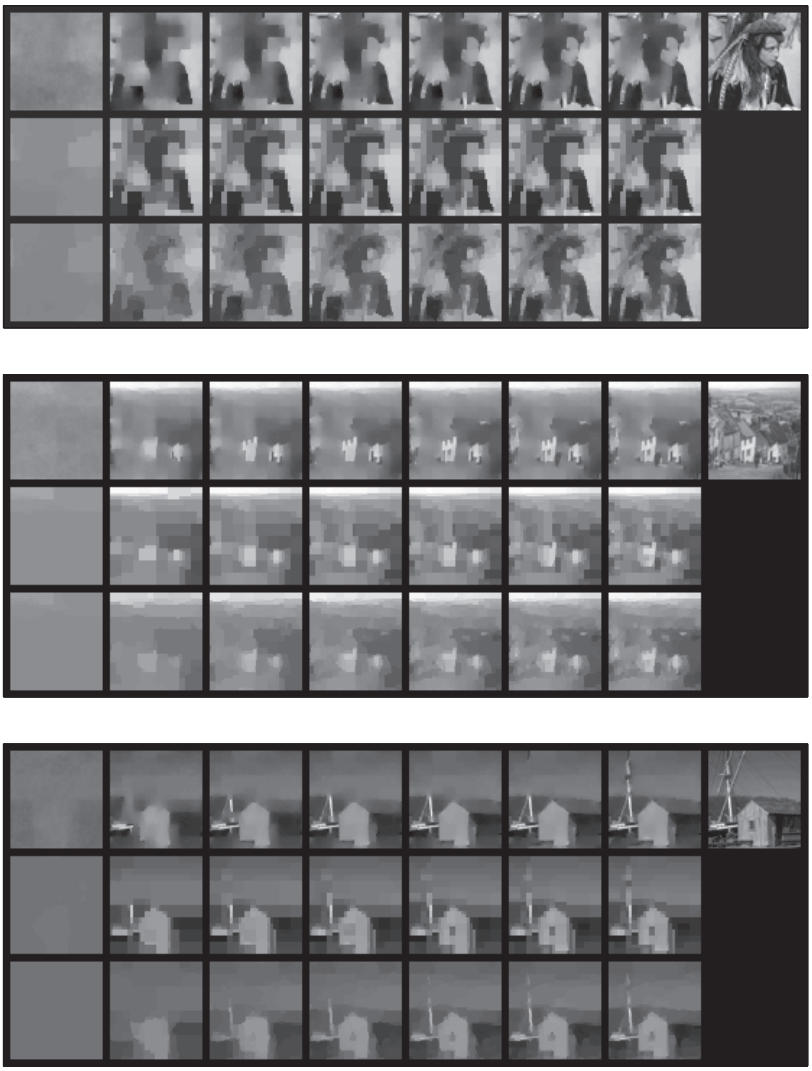


Figure 9: Reconstructed images from the Bayesian sequential experimental design (SED) experiments. We plot the estimated images obtained by each approximate inference procedure at different stages of the SED process. Each pane corresponds to a different underlying image. The true image is shown in the last image of the first row of each pane. Otherwise, the first row of each pane plots the G-KL mean, the second row the VB mean and the third row the MAP reconstruction with randomly selected measurement vectors. The $k^{th}$ column of each pane plots the estimated image using $100 + 300 \times (k-1)$ measurements.

priors placed on each of the linear filter responses had $\tau = 0.1$ for the finite difference filters and

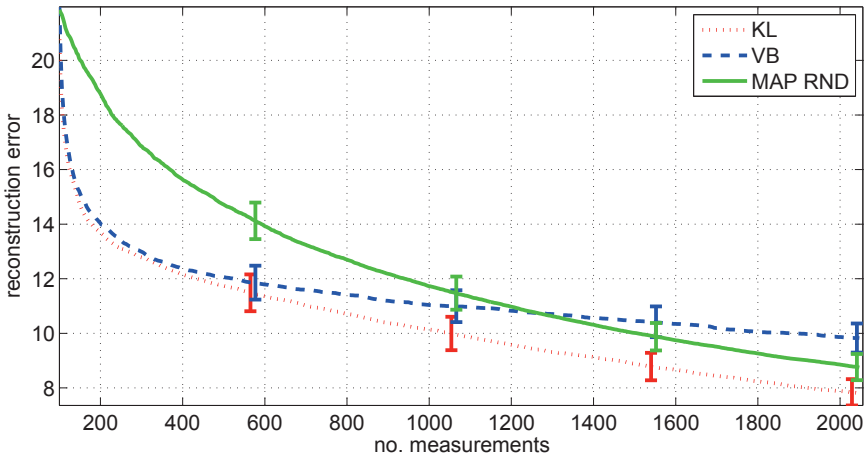



Figure 10: $\ell^2$ reconstruction errors for the natural image sequential experimental design task. Mean and standard error scores are presented averaged over 16 different $64 \times 64$ pixel images.

$\tau = 0.14$ for the wavelet filters. This experimental approach follows that laid out in Chapter 5 of Nickisch (2010).

We apply the SED procedure detailed above by iteratively approximating the posterior density $p(\mathbf{w}|\mathbf{y}, \mathbf{M}, \mathbf{B}, \boldsymbol{\tau}, \nu^2)$ where: $\mathbf{w} \in \mathbb{R}^D$ corresponds to the unknown image vector; $\mathbf{y} \in \mathbb{R}^N$ the noisy measurements where $N \ll D$; and $\mathbf{M} \in \mathbb{R}^{N \times D}$ is the linear measurement matrix constrained to have rows with unit norm. The measurement matrix is initialised with 100 standard normal randomly sampled vectors normalised to have unit norm. The sequential experimental design process approximates the posterior based on current measurements and the prior, these are then used to select new unit norm linear measurement vectors $\mathbf{M}^* \in \mathbb{R}^{3 \times D}$ to append to $\mathbf{M}$. New observations are then synthetically generated by drawing samples from the Gaussian $\mathbf{y}^* \sim \mathcal{N}\left(\mathbf{M}\mathbf{w}_{tr}, \nu^2\mathbf{I}\right)$. In the experiments conducted we use $64 \times 64 = 4096 = D$ pixel grey scale images. The images were down sampled from a collection frequently used by the vision community,[9] gray scale pixel intensities were linearly transformed to lie in $[-1, 1]$. The likelihood model was fixed with $\nu^2 = 0.005$.

In this larger setting we apply G-KL and VB approximate inference methods only and make use of approximate covariance decompositions. For G-KL approximate inference we use the chevron Cholesky decomposition with 80 non-diagonal rows. The chevron Cholesky parameterisation was chosen due to its strong performance in previous experiments with respect to both convergence time and accuracy of inference—see Section 6.2. VB inference is applied with low rank decompositions of covariance using 80 Lanczos vectors. For the first iteration of the SED procedure, $N_0 = 100$, G-KL converged in 30 seconds and VB in 5 seconds. At each iteration of the SED process each inference procedure was initialised with the posterior from the previous SED iteration. When $N = 2048$ updating the Gaussian approximate posterior took 60 seconds for G-KL and 25 seconds for VB. Convergence of VB inference is difficult to asses since the double loop algorithm with Lanczos approximated covariance is not guaranteed at each iteration to increase the approximated

9. Images were downloaded from `decsai.ugr.es/cvg/dbimagenes/index.php`.

marginal likelihood. We iterated the VB procedure for seven outer loop iterations at which point no systematic increases of approximate marginal likelihood values were observed. Fluctuations in VB approximate marginal likelihood value in subsequent iterations were roughly $\pm 10$. G-KL inference was terminated when the greatest absolute value of the bounds gradient was less than 0.1, at which point G-KL bound values increased by less than 0.5 per iteration. These results highlight a general distinction between the two methods, VB optimisation is an approximate EM algorithm whilst G-KL optimisation in this setting is implemented using an approximate second order gradient ascent procedure. EM is often reported to exhibit rapid convergence to low accuracy solutions but can be very slow at achieving high accuracy solutions (Salakhutdinov et al., 2003).

Reconstruction error results are plotted in Figure 10. We can see that SED offers greater reconstruction accuracy over random designs for a fixed budget of measurements. Up to roughly 400 designed measurement vectors both G-KL and VB procedures achieve similar reconstruction errors, after which the rate of VB iterative performance slows down eventually being overtaken by MAP reconstruction without design (MAP Rand). The reasons for this phenomenon are unclear. As more measurements are added the posterior density will become more spherical, for approximately spherical posteriors the benefit of design over simply adding random measurements is negligible. This could possibly explain the observation that G-KL and the MAP Rand procedures have similar gradients in Figure 10 towards the end of the experiment. Why the performance of VB approximate inference in particular degrades as more observations are added is not clear. One possible explanation is due to the Lanczos covariance approximation, as the posterior becomes increasingly spherical its spectrum will get flatter and the low-rank approximate factorisation may cause degraded Gaussian mean estimation.

Figure 9 displays the estimated deconvolved images at different stages of the SED process. Specifically we plot the G-KL and VB Gaussian mean estimates and the randomly designed MAP estimate. Interestingly, each method displays different visual traits with regards to the quality of the reconstructed image. G-KL estimates have patches with high fidelity and patches with low fidelity and a soft cloudy texture. VB and MAP Rand estimates appear more pixelated than the G-KL estimates with image accuracy more uniform across the image pane.

## 7. Discussion

We have presented several novel theoretical and practical developments concerning Gaussian Kullback-Leibler approximate inference procedures for models of the form of Equation (2). G-KL approximate inference is seeing a resurgence of interest by the research community—see, for example: Opper and Archambeau (2009), Ormerod and Wand (2012), Honkela et al. (2010) and Graves (2011). The work presented here provides further justification for its application as a Gaussian approximate inference procedure.

G-KL approximate inference's primary strength over other deterministic Gaussian approximate inference methods is the ease with which it can be applied to new models. All that is required to apply G-KL to a model of the form of Equation (2) is the pointwise evaluation of the univariate site projection potentials and that each of these potentials has unbounded support on $\mathbb{R}$. Unlike other deterministic Gaussian approximate inference methods G-KL does not require the site potentials to be differentiable, super-Gaussian or log-concave. Since the G-KL method optimises a strict lower-bound G-KL approximate inference is found to be numerically stable.

A long perceived disadvantage of G-KL approximate inference is the difficulty of optimising the bound with respect to the $O\left(D^2\right)$ parameters needed to specify the G-KL covariance matrix. We have shown, however, that whilst $O\left(D^2\right)$ parameters are required in full generality, the computations needed for bound optimisation compare favourably with other deterministic Gaussian approximate inference procedures. Importantly, we have shown that optimising the G-KL bound is a concave problem for models with log-concave potential functions $\{\phi_n\}_{n=1}^N$.

For larger problems we provided concave constrained parameterisations of covariance that allow G-KL methods to be applied to larger problems without imposing a priori factorisation assumptions on the approximate posterior density. The results presented in Section 6 show that such constrained covariance parameterisations are at least as good as other widely used deterministic methods at capturing posterior covariance. G-KL approximate inference using constrained concave covariance parameterisations have optimisation convergence times comparable to fast approximate variational local bound methods whilst maintaining a strict lower-bound on $\log Z$.

## 8. Publicly Available Code

A Matlab implementation of the G-KL approximate inference methods described in this paper is publicly available via the mloss.org website at `mloss.org/software/view/308/`. The `vgai` package implements G-KL approximate inference for models of the form of Equation (2) where each potential function is a site projection $\phi_n(\mathbf{w}) = \phi(\mathbf{w}^\mathsf{T}\mathbf{h}_n)$. The toolbox includes implementations of Gaussian, Laplace, Cauchy, Student's $t$, logistic sigmoid and logistic probit potential functions amongst others. Generic site projection potentials are supported if an implementation of $\psi := \log\phi : \mathbb{R} \to \mathbb{R}$ is provided. The package implements the unconstrained Cholesky, constrained Cholesky and factor analysis parameterisations of covariance discussed in Section 4.1. G-KL bound optimisation is achieved in the `vgai` package using Mark Schmidt's `minFunc` optimisation package.[10]

## Acknowledgments

The authors would like to thank the reviewers for their valuable comments which greatly helped improve this manuscript. The authors would also like to thank Michalis K. Titsias for providing a cleaner presentation of the G-KL bound concavity result and Peter Sollich for many helpful comments and suggestions.

## Appendix A. Univariate Expectation

For clarity of exposition we present a reworking of the result, as originally presented by Barber and Bishop (1998), that $\int \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})\,\psi(\mathbf{w}^\mathsf{T}\mathbf{h})d\mathbf{w} = \int \mathcal{N}\left(y|\mathbf{w}^\mathsf{T}\mathbf{h},\mathbf{h}^\mathsf{T}\mathbf{S}\mathbf{h}\right)\psi(y)dy$, where: $\psi : \mathbb{R} \to \mathbb{R}$ is some nonlinear function, $\mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})$ is a multivariate Gaussian density with mean $\mathbf{m} \in \mathbb{R}^D$ and covariance $\mathbf{S} \in \mathbb{R}^{D\times D}$, and $\mathcal{N}\left(y|\mathbf{m}^\mathsf{T}\mathbf{h},\mathbf{h}^\mathsf{T}\mathbf{S}\mathbf{h}\right)$ is a univariate Gaussian with mean $\mathbf{m}^\mathsf{T}\mathbf{h}$ and variance $\mathbf{h}^\mathsf{T}\mathbf{S}\mathbf{h}$.

10. The `minFunc` package can be downloaded from `www.di.ens.fr/~mschmidt/Software/minFunc.html`.

We start by showing that the $D$-dimensional expectation $\left\langle \psi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle_{\mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})}$ can be expressed as a univariate integral by making the substitution $\psi(\mathbf{w}^\mathsf{T}\mathbf{h}) = \int \delta(y - \mathbf{w}^\mathsf{T}\mathbf{h})\psi(y)dy$

$$\begin{aligned}
\left\langle \psi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle_{\mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})} &= \int \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})\, \psi(\mathbf{w}^\mathsf{T}\mathbf{h}) d\mathbf{w} \\
&= \int \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S}) \int \delta(y - \mathbf{w}^\mathsf{T}\mathbf{h})\psi(y) dy d\mathbf{w} \\
&= \int \underbrace{\int \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})\, \delta(y - \mathbf{w}^\mathsf{T}\mathbf{h}) d\mathbf{w}}_{:= p(y)} \psi(y) dy.
\end{aligned}$$

We now seek to show that $p(y) \equiv \mathcal{N}\left(y|\mathbf{m}^\mathsf{T}\mathbf{h}, \mathbf{h}^\mathsf{T}\mathbf{S}\mathbf{h}\right)$. First we make the substitution $\mathbf{w} = \mathbf{C}^\mathsf{T}\mathbf{v} + \mathbf{m}$, where $\mathbf{C}$ is the Cholesky decomposition of $\mathbf{S}$ such that $\mathbf{S} = \mathbf{C}^\mathsf{T}\mathbf{C}$, to get

$$p(y) := \int \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})\, \delta(y - \mathbf{w}^\mathsf{T}\mathbf{h}) d\mathbf{w} = \int \mathcal{N}(\mathbf{v}|\mathbf{0},\mathbf{I})\, \delta(y - \mathbf{v}^\mathsf{T}\mathbf{C}\mathbf{h} - \mathbf{m}^\mathsf{T}\mathbf{h}) d\mathbf{v}.$$

If we now define a basis in the vector space $\mathbf{v}$ with unit normal basis vectors $\{\mathbf{e}_d\}_{d=1}^D$ such that $\mathbf{e}_1$ is parallel to $\mathbf{C}\mathbf{h}$ so that $\mathbf{e}_1^\mathsf{T}\mathbf{C}\mathbf{h} = \|\mathbf{C}\mathbf{h}\|_2$ and so $\mathbf{e}_d^\mathsf{T}\mathbf{C}\mathbf{h} = 0$ when $d \neq 1$. Since $\mathcal{N}(\mathbf{v}|\mathbf{0},\mathbf{I})$ is isotropic the density is invariant to orthonormal transformations $\mathcal{N}(\mathbf{v}|\mathbf{0},\mathbf{I}) = \prod_{d=1}^D \mathcal{N}\left(\mathbf{e}_d^\mathsf{T}\mathbf{v}|0,1\right)$ and so

$$\begin{aligned}
p(y) &= \int \prod_{d=1}^D \mathcal{N}(v_d|0,1)\, \delta(y - \sum_{d=1}^D v_d \mathbf{e}_d^\mathsf{T}\mathbf{C}\mathbf{h} - \mathbf{m}^\mathsf{T}\mathbf{h}) d\mathbf{v} \\
&= \int \mathcal{N}(v_1|0,1)\, \delta(y - v_1 \mathbf{e}_1^\mathsf{T}\mathbf{C}\mathbf{h} - \mathbf{m}^\mathsf{T}\mathbf{h}) dv_1 \\
&= \mathcal{N}\left(y|\mathbf{m}^\mathsf{T}\mathbf{h}, \|\mathbf{C}\mathbf{h}\|_2^2\right) = \mathcal{N}\left(y|\mathbf{m}^\mathsf{T}\mathbf{h}, \mathbf{h}^\mathsf{T}\mathbf{S}\mathbf{h}\right).
\end{aligned}$$

## Appendix B. G-KL Bound Gradients

We present the G-KL bound and its gradient for Gaussian and generic site projection potentials with full Cholesky and factor analysis parameterisations of G-KL covariance. Gradients for the chevron, banded and sparse Cholesky covariance parameterisations are implemented simply by placing that Cholesky parameterisation's sparsity mask on the full Cholesky gradient matrix. Subspace Cholesky G-KL gradients and associated optimisation procedures are discussed in Section B.4.

### B.1 Entropy

For the Cholesky decomposition of covariance, $\mathbf{S} = \mathbf{C}^\mathsf{T}\mathbf{C}$, the entropy term of the G-KL bound and its gradient with respect to $\mathbf{C}$ are given by

$$\begin{aligned}
-\left\langle \log q(\mathbf{w}) \right\rangle_{q(\mathbf{w})} &= \frac{D}{2}\log(2\pi) + \frac{D}{2} + \sum_{d=1}^D \log(C_{dd}), \\
\frac{\partial}{\partial C_{ij}} -\left\langle \log q(\mathbf{w}) \right\rangle_{q(\mathbf{w})} &= \delta_{ij}\frac{1}{C_{ij}},
\end{aligned}$$

where $\delta_{ij}$ is the Kronecker delta.

For the factor analysis (FA) parameterisation of G-KL covariance, $\mathbf{S} = \text{diag}\left(\mathbf{d}^2\right) + \boldsymbol{\Theta}\boldsymbol{\Theta}^\mathsf{T}$ where $\mathbf{d} \in \mathbb{R}^D$ and $\boldsymbol{\Theta} \in \mathbb{R}^{D\times K}$, the entropy is given by,

$$-\left\langle \log q(\mathbf{w}) \right\rangle = \frac{D}{2}\log(2\pi) + \frac{D}{2} + \sum_d \log(d_d) + \frac{1}{2}\log\det\left(\mathbf{I}_{K\times K} + \boldsymbol{\Theta}^\mathsf{T}\text{diag}\left(\frac{1}{\mathbf{d}^2}\right)\boldsymbol{\Theta}\right),$$

admitting the gradients:

$$\frac{\partial}{\partial \mathbf{d}}\left\langle \log q(\mathbf{w}) \right\rangle_{q(\mathbf{w})} = 2\mathbf{d} \odot \text{diag}\left(\mathbf{S}^{-1}\right), \quad \text{and} \quad \frac{\partial}{\partial \boldsymbol{\Theta}}\left\langle \log q(\mathbf{w}) \right\rangle_{q(\mathbf{w})} = 2\mathbf{S}^{-1}\boldsymbol{\Theta}.$$

Where $\odot$ refers to taking the element wise product and $\text{diag}\left(\right)$ refers to either constructing a square diagonal matrix from a column vector or forming a column vector from the diagonal elements of a square matrix. Evaluating $\mathbf{S}^{-1}$ scales $O\left(K^2 D\right)$ using the Woodbury matrix inversion identity:

$$\mathbf{S}^{-1} = \text{diag}\left(\frac{1}{\mathbf{d}^2}\right) - \text{diag}\left(\frac{1}{\mathbf{d}^2}\right)\boldsymbol{\Theta}\left(\mathbf{I}_{K\times K} + \boldsymbol{\Theta}^\mathsf{T}\text{diag}\left(\frac{1}{\mathbf{d}^2}\right)\boldsymbol{\Theta}\right)^{-1}\boldsymbol{\Theta}^\mathsf{T}\text{diag}\left(\frac{1}{\mathbf{d}^2}\right).$$

### B.2 Site Projection Potentials

Each site projection potential's contribution to the G-KL bound can be expressed as

$$I_n = \left\langle \log \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n) \right\rangle = \left\langle \log \phi(y) \right\rangle_{\mathcal{N}(y|m_n, s_n^2)} = \left\langle \log \phi(m_n + z s_n) \right\rangle_{\mathcal{N}(z|0,1)},$$

where $m_n = \mathbf{h}_n^\mathsf{T}\mathbf{m}$ and $s_n^2 = \mathbf{h}_n^\mathsf{T}\mathbf{S}\mathbf{h}_n$. In order that general potentials of this form can be easily implemented for different functions $\phi_n$ we present the gradients according to their chain rule decomposition,

$$\frac{\partial I_n}{\partial \mathbf{m}} = \frac{\partial I_n}{\partial m_n}\frac{\partial m_n}{\partial \mathbf{m}} \quad \text{and} \quad \frac{\partial I_n}{\partial \mathbf{C}} = \frac{\partial I_n}{\partial s_n^2}\frac{\partial s_n^2}{\partial \mathbf{C}}. \tag{17}$$

Expressing $I_n$ and its derivatives as expectations with respect to the standard normal density renders the implementation of numerical integration routines simpler whilst avoiding expressions involving the derivative of the potential function itself. The expectations and their derivatives are given by:

$$\begin{aligned}
I_n &= \int \mathcal{N}(z|0,1)\log\phi_n(m_n + z s_n)dz, \\
\frac{\partial I_n}{\partial m_n} &= \int z\mathcal{N}(z|0,1)\frac{\log\phi_n(m_n + z s_n)}{s_n}dz, \\
\frac{\partial I_n}{\partial s_n^2} &= \int \left(z^2 - 1\right)\mathcal{N}(z|0,1)\frac{\log\phi_n(m_n + z s_n)}{2s_n^2}dz.
\end{aligned}$$

The gradients of $m_n = \mathbf{h}_n^\mathsf{T}\mathbf{m}$ and $s_n^2 = \mathbf{h}_n^\mathsf{T}\mathbf{S}\mathbf{h}_n$ are

$$\frac{\partial m_n}{\partial \mathbf{m}} = \mathbf{h}_n, \quad \text{and} \quad \frac{\partial s_n^2}{\partial \mathbf{C}} = 2\text{triu}\left(\mathbf{C}\mathbf{h}_n\mathbf{h}_n^\mathsf{T}\right),$$

where $\text{triu}\left(\cdot\right)$ is a sparsity mask such that elements below the diagonal are fixed to zero. For FA parameterisations we have

$$\frac{\partial s_n^2}{\partial \mathbf{d}} = 2\mathbf{h}_n^2 \odot \mathbf{d}, \quad \text{and} \quad \frac{\partial s_n^2}{\partial \boldsymbol{\Theta}} = 2\mathbf{h}_n\mathbf{h}_n^\mathsf{T}\boldsymbol{\Theta}.$$

### B.2.1 LAPLACE POTENTIALS

The Gaussian expectation of the logarithm of a Laplace potential has a simple analytic expression. Laplace potentials, as considered here, take the product of site projections form. Accordingly, we need only present the derivatives with respect to $m_n$ and $s_n^2$ so that they can be used in conjunction with Equation (17). We consider the case of a zero mean Laplace density, $p(\mathbf{w}^\mathsf{T}\mathbf{h}_n|\tau) = e^{-|\mathbf{w}^\mathsf{T}\mathbf{h}_n|/\tau}/2\tau$, giving

$$\langle \log p(m_n + z s_n) \rangle_z = -\log(2\tau) - \frac{1}{\tau} \langle |m_n + z s_n| \rangle_z . \tag{18}$$

Laplace potentials with non zero mean, $p(x) = e^{-|x-\eta|/\tau}/2\tau$, can be calculated by making the simple transformation $m_n' = m_n - \eta$. Evaluating the last term of Equation (18) above involves computing the expectation of a rectified univariate Gaussian random variable,

$$\langle |m_n + z s_n| \rangle_z = \left(\frac{2}{\pi}\right)^{\frac{1}{2}} s_n e^{-\frac{1}{2}a_n^2} + m_n \left[1 - 2\Phi(-a_n)\right]$$

where $\Phi(x) := \int_{-\infty}^{x} \mathcal{N}(t|0,1)\, dt$ and $a_n := m_n/s_n$. The corresponding derivatives of which are:

$$\frac{\partial \langle |m_n + z s_n| \rangle}{\partial m_n} = 1 - 2\Phi(-a_n),$$
$$\frac{\partial \langle |m_n + z s_n| \rangle}{\partial s_n^2} = \frac{a_n^2 + 1}{\sqrt{2\pi s_n^2}} e^{-\frac{1}{2}a_n^2} - \frac{a_n^2}{s_n} \mathcal{N}(a_n|0,1).$$

## B.3 Gaussian Potentials

For a Gaussian potential $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ the log expectation is given by

$$\left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \right\rangle_{q(\mathbf{w})} = -\frac{1}{2}\left[\log\det(2\pi\boldsymbol{\Sigma}) + (\mathbf{m}-\boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{m}-\boldsymbol{\mu}) + \operatorname{trace}\left(\boldsymbol{\Sigma}^{-1}\mathbf{S}\right)\right].$$

Derivatives with respect to the mean and covariance are:

$$\frac{\partial}{\partial \mathbf{m}} \left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \right\rangle = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{m}), \quad \text{and} \quad \frac{\partial}{\partial \mathbf{C}} \left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \right\rangle = -\operatorname{triu}\left(\mathbf{C}\boldsymbol{\Sigma}^{-1}\right).$$

For the FA covariance structure we have,

$$\frac{\partial}{\partial \mathbf{d}} \left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \right\rangle = -\operatorname{diag}\left(\boldsymbol{\Sigma}^{-1}\right) \odot \mathbf{d}, \quad \text{and} \quad \frac{\partial}{\partial \boldsymbol{\Theta}} \left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \right\rangle = -\boldsymbol{\Sigma}^{-1}\boldsymbol{\Theta}.$$

### B.3.1 GAUSSIAN LIKELIHOODS

Linear models with additive Gaussian noise have a likelihood potential that can be expressed as $\mathcal{N}\left(\mathbf{y}|\mathbf{H}^\mathsf{T}\mathbf{w}, \boldsymbol{\Sigma}\right)$ where $\mathbf{H} \in \mathbb{R}^{D\times N}$ and $\mathbf{y} \in \mathbb{R}^N$. In this setting typically we assume isotropic noise $\boldsymbol{\Sigma} = \nu^2\mathbf{I}$ and so present gradients for this case only. The expectation of the log of this term has the following algebraic from

$$\left\langle \log \mathcal{N}\left(\mathbf{y}|\mathbf{H}^\mathsf{T}\mathbf{w}, \nu^2\mathbf{I}\right) \right\rangle = -\frac{1}{2}\left[N\log(2\pi\nu^2) + \frac{1}{\nu^2}\left\langle \left(\mathbf{y} - \mathbf{H}^\mathsf{T}\mathbf{w}\right)^\mathsf{T}\left(\mathbf{y} - \mathbf{H}^\mathsf{T}\mathbf{w}\right)\right\rangle\right], \tag{19}$$

where the expectation of the quadratic can be expressed as

$$\left\langle \left(\mathbf{y}-\mathbf{H}^{\mathsf{T}}\mathbf{w}\right)^{\mathsf{T}}\left(\mathbf{y}-\mathbf{H}^{\mathsf{T}}\mathbf{w}\right)\right\rangle = \mathbf{y}^{\mathsf{T}}\mathbf{y} - 2\mathbf{y}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}}\mathbf{m} + \sum_{ij}\left[\mathbf{C}\mathbf{H}\right]_{ij}^{2} + \sum_{i}\left[\mathbf{H}^{\mathsf{T}}\mathbf{m}\right]_{i}^{2}.$$

Equation (19) admits the gradients:

$$\frac{\partial}{\partial \mathbf{m}}\left\langle \log \mathcal{N}\left(\mathbf{y}|\mathbf{H}^{\mathsf{T}}\mathbf{w},\nu^{2}\mathbf{I}\right)\right\rangle = \frac{1}{\nu^{2}}\left(\mathbf{y}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}} - \mathbf{H}\mathbf{H}^{\mathsf{T}}\mathbf{m}\right),$$
$$\frac{\partial}{\partial \mathbf{C}}\left\langle \log \mathcal{N}\left(\mathbf{y}|\mathbf{H}^{\mathsf{T}}\mathbf{w},\nu^{2}\mathbf{I}\right)\right\rangle = -\frac{1}{\nu^{2}}\mathrm{triu}\left(\mathbf{C}\mathbf{H}\mathbf{H}^{\mathsf{T}}\right).$$

For the FA parameterised covariance we have

$$\left\langle \left(\mathbf{y}-\mathbf{H}^{\mathsf{T}}\mathbf{w}\right)^{\mathsf{T}}\left(\mathbf{y}-\mathbf{H}^{\mathsf{T}}\mathbf{w}\right)\right\rangle = \mathbf{y}^{\mathsf{T}}\mathbf{y} - 2\mathbf{y}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}}\mathbf{m} + \sum_{i}\left[\mathbf{H}^{\mathsf{T}}\mathbf{m}\right]_{i}^{2} + \sum_{ij}\left[\mathbf{\Theta}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}}\right]_{ij}^{2} + \sum_{j}\left(\sum_{i}H_{ji}^{2}\right)d_{j}^{2}$$

with corresponding gradients:

$$\frac{\partial}{\partial d_{j}}\left\langle \log \mathcal{N}\left(\mathbf{y}|\mathbf{H}^{\mathsf{T}}\mathbf{w},\nu^{2}\mathbf{I}\right)\right\rangle = -\frac{1}{\nu^{2}}\left(\sum_{i}H_{ji}^{2}\right)d_{j},$$
$$\frac{\partial}{\partial \mathbf{\Theta}}\left\langle \log \mathcal{N}\left(\mathbf{y}|\mathbf{M}\mathbf{w},\nu^{2}\mathbf{I}\right)\right\rangle = -\frac{1}{\nu^{2}}\mathbf{H}\mathbf{H}^{\mathsf{T}}\mathbf{\Theta}.$$

#### B.3.2 GAUSSIAN POTENTIALS AS SITE PROJECTIONS

The Gaussian potential $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\mathbf{\Sigma})$ can be equivalently expressed as a product of $D$ site projection potentials. To see this we use the Cholesky factorisation of the precision matrix $\mathbf{\Sigma}^{-1} = \mathbf{P}^{\mathsf{T}}\mathbf{P}$. Making this substitution, we see that

$$\mathcal{N}(\mathbf{w}|\boldsymbol{\mu},\mathbf{\Sigma}) \propto e^{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^{\mathsf{T}}\mathbf{P}^{\mathsf{T}}\mathbf{P}(\mathbf{w}-\boldsymbol{\mu})} = e^{-\frac{1}{2}\|\mathbf{P}(\mathbf{w}-\boldsymbol{\mu})\|_{2}^{2}} = \prod_{d=1}^{D} e^{-\frac{1}{2}\left(\mathbf{p}_{d}^{\mathsf{T}}(\mathbf{w}-\boldsymbol{\mu})\right)^{2}}, \tag{20}$$

where the vector $\mathbf{p}_d$ is the $d^{th}$ row vector of $\mathbf{P}$, that is $\mathbf{p}_d^{\mathsf{T}} := \mathbf{P}_{d,:}$. Thus Equation 20 is a product of $D$ site projections with potential function $\phi_d(x) \propto e^{-\frac{1}{2}x^2}$.

### B.4 Subspace Covariance Decomposition

We consider optimising the G-KL bound with respect to a covariance matrix parameterised on a subspace of the parameters $\mathbf{w} \in \mathbb{R}^D$. Letting $\mathbf{E} = [\mathbf{E}_1, \mathbf{E}_2]$ be a matrix of orthonormal vectors that span $\mathbb{R}^D$ then we may parameterise the covariance as

$$\mathbf{S}' = \mathbf{E}^{\mathsf{T}}\mathbf{S}\mathbf{E} = [\mathbf{E}_1, \mathbf{E}_2]^{\mathsf{T}}\mathbf{S}\,[\mathbf{E}_1, \mathbf{E}_2],$$

which is equivalent to making an orthonormal transformation in the space of parameters $\mathbf{w}$ using $\mathbf{E}$. If we restrict $\mathbf{S}$ to be block diagonal, $\mathbf{S} = \mathrm{diag}\,(\mathbf{S}_1, \mathbf{S}_2)$, we can write $\mathbf{S}'$ as the sum

$$\mathbf{S}' = \mathbf{E}_1^{\mathsf{T}}\mathbf{S}_1\mathbf{E}_1 + \mathbf{E}_2^{\mathsf{T}}\mathbf{S}_2\mathbf{E}_2.$$

Since $\mathbf{E}$ is orthonormal it does not effect the value or gradient of the entropy's contribution to the bound since $\log\det(\mathbf{S}) = \log\det(\mathbf{S}')$. Provided the Gaussian potential has spherical covariance, $\boldsymbol{\Sigma} = \nu^2\mathbf{I}$, then $\mathbf{E}$ does not effect its contribution the G-KL bound since

$$\text{trace}\left(\boldsymbol{\Sigma}^{-1}\mathbf{S}'\right) = \frac{1}{\nu^2}\text{trace}\left(\mathbf{E}^\mathsf{T}\mathbf{S}\mathbf{E}\right) = \frac{1}{\nu^2}\text{trace}\left(\mathbf{S}\right).$$

Thus we are left to evaluate the projected variance terms $\left\{s_n^2\right\}_{n=1}^N$ required to evaluate the product of site potentials contribution. For $\mathbf{S}$ block diagonal with the second block component spherical, $\mathbf{S}_2 = c^2\mathbf{I}$, the orthonormal basis vectors $\mathbf{E}_2$ do not need to be computed or maintained since

$$s_n^2 = \mathbf{h}_n^\mathsf{T}\mathbf{S}'\mathbf{h}_n = \mathbf{h}_n^\mathsf{T}\mathbf{E}_1^\mathsf{T}\mathbf{S}_1\mathbf{E}_1\mathbf{h}_n + c^2\mathbf{h}_n^\mathsf{T}\mathbf{E}_2^\mathsf{T}\mathbf{E}_2\mathbf{h}_n = \mathbf{h}_n^\mathsf{T}\mathbf{E}_1^\mathsf{T}\mathbf{S}_1\mathbf{E}_1\mathbf{h}_n + c^2\left(\|\mathbf{h}_n\|_2^2 - \|\mathbf{E}_1\mathbf{h}_n\|^2\right).$$

We seek to optimise the G-KL bound w.r.t. to the subspace parameterised variational Gaussian by iterating between optimising the bound with respect to the parameters $\{\mathbf{m}, \mathbf{C}_1, c\}$ and updating the subspace basis vectors $\mathbf{E}_1$. In Section B.4.1 we present the gradients required to optimise the G-KL bound with respect to $\{\mathbf{m}, \mathbf{C}_1, c\}$. In Sections B.4.2 and B.4.3 we consider different routes to optimising the subspace basis $\mathbf{E}_1$.

### B.4.1 SUBSPACE CHOLESKY G-KL BOUND GRADIENTS

In this subsection we present the subspace Cholesky G-KL bound gradients. The subspace covariance matrix is given by $\mathbf{S} = \mathbf{E}_1^\mathsf{T}\mathbf{C}_1^\mathsf{T}\mathbf{C}_1\mathbf{E}_1 + c^2\mathbf{E}_2^\mathsf{T}\mathbf{E}_2$, where $\mathbf{C}_1 \in \mathbb{R}^{K\times K}$ is a Cholesky matrix, $c \in \mathbb{R}^+$ and $D = K + L$. Since $\mathbf{E}_2$ does not occur in the expressions presented below, in what follows we omit subscripts and denote $\mathbf{E}_1$ and $\mathbf{C}_1$ as $\mathbf{E}$ and $\mathbf{C}$. We reiterate that the Gaussian potential has spherical covariance $\boldsymbol{\Sigma} = \nu^2\mathbf{I}$. The G-KL bound for the subspace Cholesky covariance parameterisation is given by

$$\begin{aligned}\mathcal{B}_{KL}(\mathbf{m}, \mathbf{C}, c, \mathbf{E}) = \frac{D}{2}\log(2\pi) + \frac{D}{2} + \sum_{k=1}^K \log(C_{kk}) + L\log(c) \\ - \frac{D}{2}\log\left(2\pi\nu^2\right) - \frac{1}{\nu^2}\left[\|\mathbf{m} - \boldsymbol{\mu}\|_2^2 + \text{trace}\left(\mathbf{C}^\mathsf{T}\mathbf{C}\right) + Lc^2\right] \\ + \sum_{n=1}^N \left\langle \log\phi_n(m_n + z s_n)\right\rangle_{\mathcal{N}(z|0,1)}.\end{aligned}$$

The gradient of the G-KL entropy's contribution to the bound is

$$\frac{\partial}{\partial C_{ij}} - \langle\log q(\mathbf{w})\rangle = \delta_{ij}\frac{1}{C_{ij}}, \quad \text{and} \quad \frac{\partial}{\partial c} - \langle\log q(\mathbf{w})\rangle = \frac{L}{c}.$$

The Gaussian potential's contribution to the G-KL bound admits the gradients:

$$\frac{\partial}{\partial \mathbf{C}}\left\langle\log\mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}, \nu^2\mathbf{I}\right)\right\rangle = -\frac{1}{\nu^2}\mathbf{C}, \quad \text{and} \quad \frac{\partial}{\partial c}\left\langle\log\mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}, \nu^2\mathbf{I}\right)\right\rangle = -\frac{Lc}{\nu^2}.$$

The site projection potential's contribution to the G-KL bound is computed as in Section B.2 but with the partial derivatives of $s_n^2$ with respect to $\mathbf{C}$ and $c$:

$$\frac{\partial s_n^2}{\partial \mathbf{C}} = 2\text{triu}\left(\mathbf{C}\tilde{\mathbf{h}}_n\tilde{\mathbf{h}}_n^\mathsf{T}\right), \quad \frac{\partial s_n^2}{\partial c} = 2c\left(\|\mathbf{h}_n\|_2^2 - \|\tilde{\mathbf{h}}_n\|_2^2\right),$$

where $\tilde{\mathbf{h}}_n := \mathbf{E}\mathbf{h}_n$.

### B.4.2 SUBSPACE OPTIMISATION : PROJECTED GRADIENT ASCENT

One route to finding good subspace vectors $\mathbf{E}_1$ is to directly optimise the bound with respect to them. Again we omit subscripts since $\mathbf{E}_2$ makes no contribution to the expressions below. Optimisation is complicated by the fact that we require $\mathbf{E}$ to be orthonormal, that is we require that $\mathbf{E}^\mathsf{T}\mathbf{E} = \mathbf{I}_{K\times K}$. The set of all such orthonormal vectors forms a smooth manifold in $\mathbb{R}^{D\times K}$. A crude but simple approach to optimising the bound with respect to $\mathbf{E}$ is projected gradient ascent—after each gradient step we orthonormalise the updated basis:

$$\mathbf{E}^{new} := \text{orth}\left[\mathbf{E} + \alpha \frac{\partial}{\partial \mathbf{E}} \mathcal{B}_{KL}(\mathbf{m}, \mathbf{C}, \mathbf{E}, c)\right]$$

where $\text{orth}[\cdot]$ denotes an orthonormalisation operator, implemented for instance using a Gram-Schmidt procedure or the singular value decomposition, and $\alpha$ is a parameter controlling the gradient step size.

As described above, when $\mathbf{\Sigma} = \nu^2\mathbf{I}$, the only term in the G-KL bound that depends on $\mathbf{E}$ are the site projection potential functions $\left\langle \log \phi_n(\mathbf{w}^\mathsf{T}\mathbf{h}_n)\right\rangle$. The derivative of the bound then with respect to $\mathbf{E}$ is given by

$$\frac{\partial}{\partial \mathbf{E}} \mathcal{B}_{KL}(\mathbf{m}, \mathbf{C}, \mathbf{E}, c) = \sum_n \frac{\partial}{\partial s_n^2} \left\langle \log \phi(m_n + z s_n)\right\rangle \frac{\partial s_n^2}{\partial \mathbf{E}},$$

where the partial derivative with respect to $s_n^2$ is given in Section B.2 and

$$\frac{\partial s_n^2}{\partial \mathbf{E}} = \frac{\partial}{\partial \mathbf{E}} \mathbf{h}_n^\mathsf{T}\mathbf{E}^\mathsf{T}\mathbf{C}^\mathsf{T}\mathbf{C}\mathbf{E}\mathbf{h}_n = 2\mathbf{C}^\mathsf{T}\mathbf{C}\mathbf{E}\mathbf{h}_n\mathbf{h}_n^\mathsf{T}.$$

### B.4.3 SUBSPACE OPTIMISATION: FIXED POINT ITERATION

Another route to optimising the subspace vectors $\mathbf{E}$ is to use the form for the optimal G-KL covariance matrix presented in Equation (10). Using this method, once we have optimised the bound w.r.t. $\{\mathbf{m}, \mathbf{C}_1, c\}$ we update the subspace vectors $\mathbf{E}$ to be the leading $K$ eigenvectors of $\mathbf{S}$ as defined in Equation (21). Whilst this procedure is not guaranteed to increase the bound in experiments it has yielded strong performance—see for example Section 6.2 and Challis and Barber (2011).

For problems where the Gaussian potential has isotropic variance, $\mathbf{\Sigma} = \nu^2\mathbf{I}$, the form for the optimal G-KL inverse covariance, Equation (10), simplifies to

$$\mathbf{S}^{-1} = \frac{1}{\nu^2}\mathbf{I} + \mathbf{H}\mathbf{\Gamma}\mathbf{H}^\mathsf{T}, \tag{21}$$

where $\mathbf{\Gamma}$ is defined in Equation (11) of Section 11. We now consider two routes to updating the subspace vectors $\mathbf{E}$. First, we consider an approximate eigen decomposition method suitable for smaller non-sparse problems. Second, we consider an iterative Lanczos method better suited to larger sparse problems.

One route to possibly recovering the $K$ leading eigenvectors of $\mathbf{S}$ is to evaluate the $K$ smallest eigenvectors of $\frac{1}{\nu^2}\mathbf{I} + \mathbf{H}\mathbf{\Gamma}\mathbf{H}^\mathsf{T}$. We note that $\mathbf{H}\mathbf{\Gamma}\mathbf{H}^\mathsf{T} \approx \mathbf{H}\mathbf{\Gamma}'\mathbf{H}^\mathsf{T}$ where $\Gamma'_{nn} = \Gamma_{nn}$ if $\Gamma_{nn} > \delta$ and zero otherwise - we set $\delta$ small enough such that there are $K$ non zero diagonal elements $\mathbf{\Gamma}'$. If we now calculate the eigen decomposition to $\mathbf{H}\mathbf{\Gamma}'\mathbf{H}^\mathsf{T} = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^\mathsf{T}$ we see that

$$\left[\frac{1}{\nu^2}\mathbf{I} + \mathbf{H}\mathbf{\Gamma}'\mathbf{H}^\mathsf{T}\right]^{-1} = \mathbf{E}\text{diag}\left(\frac{\nu^2}{1 + \lambda'_{nn}\nu^2}\right)\mathbf{E}^\mathsf{T}.$$

For $L \ll D$ we can evaluate the $L$ eigenvectors of $\mathbf{H}\boldsymbol{\Gamma}'\mathbf{H}^\mathsf{T}$ cheaply since the eigenvalues of $\mathbf{X}\mathbf{X}^\mathsf{T}$ coincide with the eigenvalues of $\mathbf{X}^\mathsf{T}\mathbf{X}$.[11] Therefore approximating the $K$ dimensional subspace eigen decomposition reduces to the complexity of decomposing a $K \times K$ matrix. If $\delta$ is small enough this method can often outperform approximate iterative decompositions provided the data is non-sparse and of moderate dimensionality.

Iterative Lanczos methods can approximately recover the eigenvectors corresponding to the largest and smallest eigenvalues of a matrix. General details about Lanczos methods can be found in Golub and Van Loan (1996), for the special case of covariance matrices of the form Equation (21) details are provided in Seeger (2010). Iterative Lanczos methods are fast provided the number of eigenvectors we wish to recover is not too large and matrix vector products can be computed efficiently—for example when the matrix has some special structure or is sparse.

## Appendix C. Newton Convergence Rate Conditions

Sufficient conditions under which optimising $\mathcal{B}_{KL}(\mathbf{m}, \mathbf{C})$ using Newton's method will exhibit quadratic convergence rates are that $\mathcal{B}_{KL}(\mathbf{m}, \mathbf{C})$ is twice continuously differentiable, strongly concave, has closed sublevel sets and has Lipschitz continuous Hessians on the sublevel sets (Boyd and Vandenberghe, 2004, section 9.5.3). In Section 3.2 we showed that if all $\phi_n$ are log-concave then the bound is strongly concave in $\mathbf{m}, \mathbf{C}$. In this section we provide conditions for which the other requirements hold.

We consider G-KL inference problems of the form of Equation (2) where $\{\phi_n\}_{n=1}^N$ are site projection potentials that are piecewise exponentiated quadratics, log-concave and have unbounded support on $\mathbb{R}$. Specifically, we show that the required properties hold for potential functions that can be written

$$\phi(x) := \sum_{i=0}^{I} \mathbb{I}\left[x \in (l_i, l_{i+1})\right] \exp(a_i x^2 + b_i x + c_i)$$

where $-\infty = l_0 < l_1, ..., l_{I+1} = \infty$ and $\mathbb{I}[\cdot]$ is an indicator function equal to one when its argument is true and zero otherwise. Note that $\phi(x)$ need not be continuous and can have jump discontinuities at the partition points $l_k$. For such functions we have that $\log \phi(x) = \sum_{i=0}^{I} \mathbb{I}\left[x \in (l_i, l_{i+1})\right] a_i x^2 + b_i x + c_i$.

### C.1 Continuously Differentiable

The expectation of such potentials can then be expressed as a sum of integrals each over a disjoint domain

$$\left\langle \log \phi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle = \sum_{i=0}^{I} \int_{l_i}^{l_{i+1}} \mathcal{N}\left(z|m, s^2\right) a_i z^2 + b_i z + c_i dz, \tag{22}$$

where $m = \mathbf{m}^\mathsf{T}\mathbf{h}$ and $s^2 = \|\mathbf{C}\mathbf{h}\|_2^2$. Each integral on the right hand side of Equation (22) has a known analytic form which depends on terms of up to order 2 in $m, s$, standard normal density functions and standard normal cumulative distribution functions—see Marlin et al. (2011) and Herbrich (2005) for their explicit forms and derivatives w.r.t. $m, s$. As an example, and to make this more concrete, we give the truncated expectation of just the quadratic term $a_i z^2$ below

$$\int_{l_i}^{l_{i+1}} a_i z^2 \mathcal{N}\left(z|m, s^2\right) dz = a_i \left[ s^2 \left( \tilde{l}_i \mathcal{N}(\tilde{l}_i) - \tilde{l}_{i+1} \mathcal{N}(\tilde{l}_{i+1}) \right) + \left(s^2 + m^2\right) \left( \Phi(\tilde{l}_{i+1}) - \Phi(\tilde{l}_i) \right) \right],$$

11. To see this consider the eigen equation for $\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{E} = \mathbf{E}\boldsymbol{\Lambda}$ thus $\mathbf{X}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{E} = \mathbf{X}\mathbf{E}\boldsymbol{\Lambda}$.

where $\tilde{l}_i := (l_i - m)/s$, $\mathcal{N}(x)$ is the standard normal density function and $\Phi(x)$ the standard normal cumulative distribution function. The truncated Gaussian expectation of the linear, $b_i z$, and the constant, $c_i$, terms have similar simpler analytic expressions.

We note that the standard normal density function and the standard normal cumulative density function are both smooth. Thus the expectation in Equation (22) is the sum of smooth functions w.r.t. the parameters $m, s$. Therefore Equation (22) as a function of $\mathbf{m}, \mathbf{C}$ is the composition of a function that is smooth in $m, s$ and the functions $m = \mathbf{m}^\mathsf{T}\mathbf{h}$ and $s^2 = \|\mathbf{C}\mathbf{h}\|_2^2$ that are smooth in $\mathbf{m}, \mathbf{C}$. By the chain rule, we see that $\left\langle \log \phi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle$ is smooth with respect to $\mathbf{m}, \mathbf{C}$.

By Lebesgue's dominated convergence theorem, we expect the differentiability of $\left\langle \log \phi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle$ to hold for a much broader class of potentials $\phi$ than the piecewise exponentiated quadratic class of functions considered here.

### C.2 G-KL Sublevel Sets are Closed

The G-KL sublevel sets, $\mathcal{S}$, are defined

$$\mathcal{S} := \left\{ \mathbf{m} \in \mathbb{R}^D, \mathbf{C} \in \mathbb{R}^{D \times D}_{chol} | \mathcal{B}(\mathbf{m}, \mathbf{C}) \geq \mathcal{B}(\mathbf{m}_0, \mathbf{C}_0) \right\},$$

where $\mathbf{m}_0, \mathbf{C}_0$ are the moments that the G-KL bound optimisation procedure is initialised with and $\mathbb{R}^{D \times D}_{Chol}$ is the set of $D \times D$ upper triangular Cholesky matrices with strictly positive diagonals. Importantly $\mathcal{S}$ is closed since the G-KL bound is a closed function—which is a sufficient condition (Boyd and Vandenberghe, 2004, p.471). A function $f : \mathbb{R}^D \to \mathbb{R}$ with $\mathrm{dom}(f)$ open is closed iff $f$ converges to $-\infty$ along every sequence converging to a boundary point of $\mathrm{dom}(f)$ (Boyd and Vandenberghe, 2004, p.640). The G-KL bound is closed since it is the sum of the entropic term (which up to a constant is equal to $\sum_d \log C_{dd}$), a negative quadratic in $\mathbf{m}, \mathbf{C}$, and $\left\langle \log \phi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle$ (proven to be jointly concave in $\mathbf{m}, \mathbf{C}$). Thus for any sequence of moments $\{\mathbf{m}_k, \mathbf{C}_k\}$ that converges to the boundary of the G-KL domain we have $\mathcal{B}_{KL}(\mathbf{m}_k, \mathbf{C}_k)$ converging to $-\infty$.

### C.3 G-KL Lipschitz Continuous Hessians

We say the Hessian of $f$ is Lipschitz continuous on $\mathcal{S}$ if there exists a constant $L \geq 0$ such that $\forall \mathbf{x}, \mathbf{y} \in \mathcal{S}$

$$\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\|_2 \leq L \|\mathbf{x} - \mathbf{y}\|_2.$$

An equivalent condition is that the Hessian has bounded and continuous derivatives on $\mathcal{S}$. Since the bound is continuously differentiable, since the sublevel sets are closed and since the entropy's contribution to the bound ensures that $s^2$ is bounded below by a positive constant this property holds.

## Appendix D. Complexity of Bound and Gradient Computations

To perform G-KL approximate inference we optimise the G-KL bound, Equation (7), by gradient ascent. In this section we consider the computational scaling properties of single evaluations of the bound and its gradient. We consider each term that depends on the variational parameters $\mathbf{m}$ and $\mathbf{S}$ separately, namely: $\log\det(\mathbf{S})$ from the entropy's contribution, $\mathrm{trace}\left(\boldsymbol{\Sigma}^{-1}\mathbf{S}\right)$ and $\mathbf{m}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{m}$ from the Gaussian potential's contribution, and $\left\{m_n, s_n^2\right\}_{n=1}^N$ from the product of site projection potential's contribution.

The G-KL covariance parameterisations we consider are: full Cholesky, diagonal Cholesky, banded Cholesky with bandwidth $B$, chevron Cholesky with $K$ non-diagonal rows, subspace Cholesky

with $K$ dimensional subspace, sparse Cholesky with $DK$ non-zeros, and factor analysis (FA) with $K$ factor loading vectors. We report only the leading scaling terms and assume, for the sake of clarity, that $N \geq D \geq K, B$ where $N$ is the number of site factors and $D$ is the dimensionality of the parameter vector $\mathbf{w}$. In the last column we report the complexity figures required to compute the projected Gaussian moments $\{m_n, s_n^2\}_{n=1}^N$ where $m_n = \mathbf{m}^\mathsf{T}\mathbf{h}_n$, $s_n^2 = \|\mathbf{C}\mathbf{h}_n\|_2^2$, and $\mathtt{nnz} : \mathbb{R}^D \to \mathbb{N}$ is a function that counts the number of non-zero elements in a vector.

| | $\log\det(\mathbf{S})$ | trace $(\boldsymbol{\Sigma}^{-1}\mathbf{S})$ | | | $\mathbf{m}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{m}$ | | | $\{m_n, s_n^2\}_{n=1}^N$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $\boldsymbol{\Sigma}$ - iso | $\boldsymbol{\Sigma}$ - diag | $\boldsymbol{\Sigma}$ - full | $\boldsymbol{\Sigma}$ - iso | $\boldsymbol{\Sigma}$ - diag | $\boldsymbol{\Sigma}$ - full | $\mathtt{nnz}(\mathbf{h}) = D$ | $\mathtt{nnz}(\mathbf{h}) = L$ |
| $\mathbf{C}_{full}$ | $O(D)$ | $O(D^2)$ | $O(D^2)$ | $O(D^3)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(ND^2)$ | $O(NDL)$ |
| $\mathbf{C}_{diag}$ | $O(D)$ | $O(D)$ | $O(D)$ | $O(D)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(ND)$ | $O(NL)$ |
| $\mathbf{C}_{band}$ | $O(D)$ | $O(DB)$ | $O(DB)$ | $O(D^2B)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(NDB)$ | $O(NLB)$ |
| $\mathbf{C}_{chev}$ | $O(D)$ | $O(DK)$ | $O(DK)$ | $O(D^2K)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(NDK)$ | $O(NLK)$ |
| $\mathbf{C}_{sub}$ | $O(K)$ | $O(DK)$ | $O(DK)$ | $O(K^3)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(NK^2)$ | $O(NK^2)$ |
| $\mathbf{C}_{spar}$ | $O(D)$ | $O(DK)$ | $O(DK)$ | $O(D^2K)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(NDK)$ | $O(NLK)$ |
| $\mathbf{S}_{FA}$ | $O(D^2K)$ | $O(DK)$ | $O(DK)$ | $O(KD^2)$ | $O(D)$ | $O(D)$ | $O(D^2)$ | $O(NDK)$ | $O(NLK)$ |

## Appendix E. Transformation of Basis

When the model's Gaussian potential, $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, has full covariance optimising the G-KL bound can sometimes be made less expensive by linearly transforming the basis of the parameter vectors $\mathbf{m}$ and $\mathbf{C}$. To do this, essentially we hard code the information contributed to the posterior from the Gaussian potential into our G-KL parameters. That is we parameterise $\mathbf{m}$ and $\mathbf{C}$ as

$$\mathbf{C} = \tilde{\mathbf{C}}\mathbf{P} \quad \text{and} \quad \mathbf{m} = \mathbf{P}^\mathsf{T}\tilde{\mathbf{m}} + \boldsymbol{\mu} \tag{23}$$

where $\mathbf{P}$ is the Cholesky decomposition of the prior covariance such that $\mathbf{P}^\mathsf{T}\mathbf{P} = \boldsymbol{\Sigma}$. For the G-KL moments parameterised this way we each term of the G-KL bound can be evaluated using:

$$\begin{aligned}
-\langle \log q(\mathbf{w}) \rangle &= \log\det\left(\tilde{\mathbf{C}}\right) + \log\det(\mathbf{P}) + \frac{D}{2}\log(2\pi) + \frac{D}{2}, \\
2\left\langle \log \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \right\rangle &= -D\log(2\pi) - D - 2\log\det(\mathbf{P}) - \tilde{\mathbf{m}}^\mathsf{T}\tilde{\mathbf{m}} - \mathrm{trace}\left(\tilde{\mathbf{C}}^\mathsf{T}\tilde{\mathbf{C}}\right), \\
\left\langle \psi(\mathbf{w}^\mathsf{T}\mathbf{h}) \right\rangle &= \int \mathcal{N}(z|0,1)\,\psi(m + zs)dz,
\end{aligned}$$

where $m := \tilde{\mathbf{m}}^\mathsf{T}\tilde{\mathbf{h}} + \boldsymbol{\mu}^\mathsf{T}\mathbf{h}$, $s := \|\tilde{\mathbf{C}}\tilde{\mathbf{h}}\|_2^2$ and $\tilde{\mathbf{h}} := \mathbf{P}\mathbf{h}$. Combining these terms the G-KL bound can be written

$$\mathcal{B}(\mathbf{m}, \mathbf{C}) = \tilde{\mathcal{B}}(\tilde{\mathbf{m}}, \tilde{\mathbf{C}}) = \sum_d \log\left(\tilde{C}_{dd}\right) - \frac{1}{2}\tilde{\mathbf{m}}^\mathsf{T}\tilde{\mathbf{m}} - \frac{1}{2}\sum_{ij}\tilde{C}_{ij}^2 + \sum_n \left\langle \log\phi_n(m_n + zs_n) \right\rangle_{\mathcal{N}(z,0)1}.$$

We are free then to optimise the G-KL bound just with respect to $\tilde{\mathbf{m}}, \tilde{\mathbf{C}}$ at a reduced cost. For a model with a full covariance Gaussian potential and non-sparse $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_N]$ computing the bound and gradient of $\tilde{\mathcal{B}}(\tilde{\mathbf{m}}, \tilde{\mathbf{C}})$ scales $O\left(D^2 + ND^2\right)$ whereas computing the bound and gradient of the untransformed bound scales $O\left(D^3 + ND^2\right)$—see the table in Appendix D.

This procedure requires some pre-processing—namely the Cholesky decomposition of $\boldsymbol{\Sigma}$ and the 'whitening' of the data set $\tilde{\mathbf{H}} = \mathbf{P}\mathbf{H}$ which scale $O\left(D^3\right)$ and $O\left(ND^2\right)$ respectively. And some post-processing—the final G-KL moments $\mathbf{m}$ and $\mathbf{C}$ are obtained using equations Equation (23) which require a matrix-vector and a matrix-matrix product which scale $O\left(D^2\right)$ and $O\left(D^3\right)$ respectively.

Since during optimisation the bound and its gradient are usually computed many more times than twice, the basis transformation procedure detailed above will result in a significant computational saving. Note that this procedure can speed up G-KL bound optimisation only in settings where $\mathbf{h}_n$ are not sparse. For example Gaussian process regression models, where $\mathbf{h}_n$ are standard normal basis vectors, will not benefit from this reparameterisation since $\tilde{\mathbf{h}}_n = \mathbf{P}\mathbf{h}_n$ are not sparse.

## Appendix F. Gaussian Process Regression

In this section we present equations necessary to implement Gaussian process regression models using G-KL approximate inference methods.

### F.1 Predictive Density

A Gaussian approximation to the posterior density on the latent function values of the training data may be used to obtain an approximation to the predictive density of the latent function value for a new test point. The GP predictive density to the target variable $y_*$ for a new input $\mathbf{x}_*$ is defined by the integral

$$p(y_*|\mathbf{x}_*,\mathbf{X},\mathbf{y}) = \int p(y_*|w_*)p(w_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)dw_*.$$

The distribution on the test point latent function value, $p(w_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)$, is approximated by marginalising out the training set latent variables using our Gaussian approximate posterior, $\mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S}) \approx p(\mathbf{w}|\mathbf{X},\mathbf{y},\boldsymbol{\theta})$, giving

$$\begin{aligned}
p(w_*|\mathbf{X},\mathbf{y},\mathbf{x}_*) &= \int p(w_*|\mathbf{w},\mathbf{X},\mathbf{x}_*)p(\mathbf{w}|\mathbf{y},\mathbf{X})d\mathbf{w} \\
&= \int \mathcal{N}\left(w_*|\boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{w}, \sigma_{**} - \boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}_*\right) p(\mathbf{w}|\mathbf{y},\mathbf{X})d\mathbf{w} \\
&\approx \int \mathcal{N}\left(w_*|\boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{w}, \sigma_{**} - \boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}_*\right) \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{S})\, d\mathbf{w} \\
&= \mathcal{N}\left(w_*|\boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{m}, \sigma_{**} - \boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}_* + \boldsymbol{\sigma}_*^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{S}\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}_*\right),
\end{aligned}$$

where $\boldsymbol{\sigma}_*$ and $\sigma_{**}$ are the prior covariance and variance terms of the test data point $\mathbf{x}_*$. The elements of $\boldsymbol{\sigma}_*$ are calculated by evaluating the covariance function, $k(\mathbf{x},\mathbf{x}')$, between the each of the training covariates and the test point covariate such that $[\boldsymbol{\sigma}_*]_m = k(\mathbf{x}_m,\mathbf{x}_*)$ and $\sigma_{**} = k(\mathbf{x}_*,\mathbf{x}_*)$.

### F.2 Hyperparameter Optimisation

For a general likelihood $p(\mathbf{y}|\mathbf{w}) = \prod_{n=1}^N \phi_n(w_n)$ and GP prior $\mathcal{N}(\mathbf{w}|\mathbf{0},\boldsymbol{\Sigma})$ with covariance function $\Sigma_{mn} = k(\mathbf{x}_m,\mathbf{x}_n)$ we get the G-KL bound

$$\begin{aligned}
\mathcal{B}_{KL}(\mathbf{m},\mathbf{C}) = \frac{D}{2} + \sum_n \log C_{nn} - \frac{1}{2}\log\det(\boldsymbol{\Sigma}) - \frac{1}{2}\mathbf{m}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{m} - \frac{1}{2}\mathrm{trace}\left(\boldsymbol{\Sigma}^{-1}\mathbf{S}\right) \\
+ \sum_n \left\langle \log\phi(m_n + z\sqrt{S_{nn}}) \right\rangle_{\mathcal{N}(z|0,1)}.
\end{aligned}$$

Taking the derivative of the above expression with respect to the covariance hyperparameters $\boldsymbol{\theta}$ we get

$$\frac{\partial\mathcal{B}_{KL}}{\partial\boldsymbol{\theta}} = -\frac{1}{2}\mathrm{trace}\left(\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\boldsymbol{\theta}}\right) + \frac{1}{2}\mathbf{m}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\boldsymbol{\theta}}\boldsymbol{\Sigma}^{-1}\mathbf{m} + \frac{1}{2}\mathrm{trace}\left(\mathbf{C}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\boldsymbol{\theta}}\boldsymbol{\Sigma}^{-1}\mathbf{C}\right). \tag{24}$$

Note that $\mathbf{m}$ and $\mathbf{C}$ implicitly depend on the covariance hyperparameters $\boldsymbol{\theta}$. However, cross terms such as

$$\frac{\partial \mathcal{B}_{KL}}{\partial \mathbf{m}} \frac{\partial \mathbf{m}}{\partial \boldsymbol{\theta}} \quad \text{or} \quad \frac{\partial \mathcal{B}_{KL}}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \boldsymbol{\theta}}$$

do not contribute to Equation (24) at the optimum of the G-KL bound since the gradients of $\mathcal{B}_{KL}$ with respect to $\mathbf{m}$ or $\mathbf{C}$ are zero at this point. Therefore, to evaluate the gradient of $\mathcal{B}_{KL}$ with respect to the covariance hyperparameters first the G-KL bound is optimised with respect to $\mathbf{m}, \mathbf{C}$ with $\boldsymbol{\theta}$ fixed, then at that optimum we use Equation (24) to calculate the derivative with respect to $\boldsymbol{\theta}$.

## Appendix G. Bayesian Logistic Regression Results

| | | | $N_{trn} = 125$ | | $N_{trn} = 250$ | | $N_{trn} = 1250$ | |
|---|---|---|---|---|---|---|---|---|
| | | | $K = 13$ | $K = 25$ | $K = 13$ | $K = 25$ | $K = 13$ | $K = 25$ |
| Time (s) | G-KL | Chev | 0.14±0.01 | 0.16±0.00 | 0.32±0.02 | 0.34±0.01 | 3.31±0.09 | 3.38±0.14 |
| | | Band | 0.21±0.01 | 0.28±0.01 | 0.41±0.01 | 0.53±0.01 | 4.05±0.09 | 4.64±0.09 |
| | | Sub | 0.42±0.05 | 0.46±0.02 | 0.69±0.03 | 0.81±0.04 | 4.24±0.15 | 5.17±0.28 |
| | | FA | 0.75±0.05 | 0.74±0.05 | 0.94±0.08 | 1.12±0.08 | 6.18±0.61 | 5.49±0.40 |
| | VB | | 0.27±0.01 | 0.28±0.00 | 0.29±0.00 | 0.31±0.01 | 0.46±0.01 | 0.45±0.00 |
| $\tilde{\mathcal{B}}$ | G-KL | Chev | −1.08±0.02 | −1.05±0.02 | −0.89±0.01 | −0.87±0.01 | −0.41±0.00 | −0.40±0.00 |
| | | Band | −1.05±0.02 | −1.00±0.01 | −0.88±0.01 | −0.85±0.01 | −0.41±0.00 | −0.40±0.00 |
| | | Sub | −2.93±0.01 | −2.11±0.02 | −1.83±0.01 | −1.43±0.01 | −0.60±0.00 | −0.52±0.00 |
| | | FA | −1.08±0.02 | −1.06±0.02 | −0.89±0.01 | −0.87±0.01 | −0.40±0.00 | −0.39±0.00 |
| | VB | | –±– | –±– | –±– | –±– | –±– | –±– |
| $\|\mathbf{m} - \mathbf{w}_{tr}\|_2/D$ | G-KL | Chev | 1.48±0.01 | 1.48±0.01 | 1.38±0.01 | 1.38±0.01 | 1.11±0.01 | 1.11±0.01 |
| | | Band | 1.48±0.01 | 1.48±0.01 | 1.38±0.01 | 1.38±0.01 | 1.11±0.01 | 1.11±0.01 |
| | | Sub | 1.49±0.01 | 1.48±0.01 | 1.43±0.01 | 1.41±0.01 | 1.20±0.01 | 1.18±0.01 |
| | | FA | 1.48±0.01 | 1.48±0.01 | 1.38±0.01 | 1.38±0.01 | 1.11±0.01 | 1.11±0.01 |
| | VB | | 1.52±0.01 | 1.51±0.01 | 1.45±0.01 | 1.45±0.02 | 1.21±0.01 | 1.21±0.01 |
| $\log p(\mathbf{y}^*\|\mathbf{X}^*)/N_{tst}$ | G-KL | Chev | −0.57±0.01 | −0.56±0.01 | −0.47±0.01 | −0.47±0.01 | −0.19±0.00 | −0.19±0.00 |
| | | Band | −0.56±0.01 | −0.56±0.01 | −0.47±0.01 | −0.46±0.01 | −0.19±0.00 | −0.19±0.00 |
| | | Sub | −0.67±0.02 | −0.63±0.02 | −0.57±0.02 | −0.54±0.02 | −0.21±0.01 | −0.20±0.01 |
| | | FA | −0.57±0.01 | −0.57±0.01 | −0.48±0.01 | −0.47±0.01 | −0.19±0.00 | −0.19±0.00 |
| | VB | | −0.68±0.02 | −0.68±0.02 | −0.57±0.01 | −0.56±0.01 | −0.21±0.01 | −0.21±0.01 |

Table 3: Bayesian logistic regression results for a unit variance Gaussian prior, with parameter dimension $D = 250$ and number of test points $N_{tst} = 2500$. Experimental setup and metrics are described in Section 6.2.

## References

D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

D. Barber and C. Bishop. Ensemble learning in bayesian neural networks. In *Neural Networks and Machine Learning*, pages 215–237. Springer, 1998.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

E. Challis and D. Barber. Concave gaussian variational approximations for inference in large-scale bayesian linear models. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.

| | | | $N_{trn}=500$ | | $N_{trn}=1000$ | | $N_{trn}=5000$ | |
|---|---|---|---|---|---|---|---|---|
| | | | $K=50$ | $K=100$ | $K=50$ | $K=100$ | $K=50$ | $K=100$ |
| Time (s) | G-KL | Chev | 2.68±0.04 | 3.37±0.05 | 6.41±0.11 | 7.28±0.14 | 75.23±1.51 | 78.38±2.10 |
| | | Band | 6.66±0.58 | 8.97±0.14 | 12.81±0.15 | 20.59±0.26 | 127.69±2.36 | 190.65±3.47 |
| | | Sub | 1.59±0.07 | 2.58±0.12 | 3.24±0.03 | 7.71±0.20 | 56.67±1.80 | 75.35±1.63 |
| | | FA | 9.94±1.00 | 12.24±0.63 | 16.21±0.74 | 18.64±1.09 | 70.87±3.92 | 82.13±5.83 |
| | VB | | 1.78±0.03 | 2.65±0.05 | 4.12±0.04 | 6.17±0.07 | 21.88±0.03 | 33.87±0.02 |
| $\tilde{\mathcal{B}}$ | G-KL | Chev | −1.28±0.01 | −1.24±0.01 | −0.99±0.00 | −0.96±0.00 | −0.42±0.00 | −0.41±0.00 |
| | | Band | −1.24±0.01 | −1.17±0.01 | −0.98±0.00 | −0.94±0.00 | −0.42±0.00 | −0.42±0.00 |
| | | Sub | −5.40±0.23 | −4.54±0.25 | −7.56±0.00 | −1.52±0.00 | −0.62±0.00 | −0.54±0.00 |
| | | FA | −1.29±0.01 | −1.26±0.01 | −1.00±0.00 | −0.97±0.00 | −0.42±0.00 | −0.41±0.00 |
| | VB | | –±– | –±– | –±– | –±– | –±– | –±– |
| $\|\mathbf{w}-\mathbf{w}_{tr}\|_2/D$ | G-KL | Chev | 0.53±0.00 | 0.53±0.00 | 0.49±0.00 | 0.49±0.00 | 0.38±0.00 | 0.38±0.00 |
| | | Band | 0.53±0.00 | 0.53±0.00 | 0.49±0.00 | 0.49±0.00 | 0.38±0.00 | 0.38±0.00 |
| | | Sub | 0.56±0.00 | 0.55±0.00 | 0.56±0.00 | 0.50±0.00 | 0.44±0.00 | 0.43±0.00 |
| | | FA | 0.53±0.00 | 0.53±0.00 | 0.49±0.00 | 0.49±0.00 | 0.38±0.00 | 0.38±0.00 |
| | VB | | 0.54±0.00 | 0.54±0.00 | 0.52±0.00 | 0.52±0.00 | 0.45±0.00 | 0.45±0.00 |
| $\log p(\mathbf{y}^*\|\mathbf{X}^*)/N_{tst}$ | G-KL | Chev | −0.62±0.01 | −0.61±0.01 | −0.51±0.01 | −0.49±0.01 | −0.18±0.00 | −0.18±0.00 |
| | | Band | −0.61±0.01 | −0.59±0.01 | −0.50±0.01 | −0.49±0.01 | −0.18±0.00 | −0.18±0.00 |
| | | Sub | −0.62±0.01 | −0.61±0.01 | −0.69±0.00 | −0.61±0.01 | −0.21±0.00 | −0.21±0.00 |
| | | FA | −0.62±0.01 | −0.61±0.01 | −0.52±0.01 | −0.51±0.01 | −0.18±0.00 | −0.18±0.00 |
| | VB | | −0.88±0.01 | −0.95±0.02 | −0.68±0.01 | −0.70±0.01 | −0.21±0.00 | −0.21±0.00 |

Table 4: Bayesian logistic regression results for a unit variance Gaussian prior, with parameter dimension $D=1000$ and number of test points $N_{tst}=5000$. Experimental setup and metrics are described in Section 6.2.

R. Fergus, B. Singh, A. Hertzmann, S. Roweis, and W. Freeman. Removing camera shake from a single photograph. In *ACM Transactions on Graphics*.

J. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991.

M. Gibbs and D. MacKay. Variational gaussian process classifiers. *IEEE Transactions on Neural Networks*, 11(6):1458–1464, 2000.

M. Girolami. A variational method for learning sparse and overcomplete representations. *Neural Computation*, 13(11):2517–2532, 2001.

G. Golub and C. Van Loan. *Matrix Computations*. John Hopkins University Press, 1996.

A. Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems 24*, 2011.

R. Herbrich. On gaussian expectation propagation. Technical report, Microsoft Research Cambridge, `research.microsoft.com/pubs/74554/EP.pdf`, 2005.

A. Honkela, T. Raiko, M. Kuusela, M. Tornio, and J. Karhunen. Approximate riemannian conjugate gradient learning for fixed-form variational bayes. *Journal of Machine Learning Research*, 11: 3235–3268, 2010.

T. Jaakkola and M. Jordan. A variational approach to bayesian logistic regression problems and their extensions. In *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, 1997.

P. Jylanki, J. Vanhatalo, and A. Vehtrari. Robust Gaussian Process Regression with a Student-t Likelihood. *Journal of Machine Learning Research*, 12:3187–3225, 2011.

K. Ko and M. Seeger. Large scale variational bayesian inference for structured scale mixture models. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

M. Kuss. *Gaussian Process Models for Robust Regression, Classification, and Reinforcement Learning*. PhD thesis, Technischen Universität Darmstadt, Darmstadt, Germany, 2006.

M. Kuss and C. Rasmussen. Assessing approximate inference for binary gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

D. MacKay and J. Oldfield. Generalization error and the number of hidden units in a multilayer perceptron. Technical report, Cambridge University, `www.inference.phy.cam.ac.uk/mackay/gen.ps.gz`, 1995.

B. Marlin, M. Khan, and K. Murphy. Piecewise bounds for estimating bernoulli-logistic latent gaussian models. In *Proceedings of the 28th International Conference on Machine Learning*, 2011.

T. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, 2001.

T. Minka. Power EP. Technical report, Department of Statistics, Carnegie Mellon University, `research.microsoft.com/pubs/67427/tr-2004-149.pdf`, 2004.

R. Neal. Monte carlo implementation of gaussian process models for bayesian regression and classification. Technical report, Department of Statistics and Department of Computer Science, University of Toronto, `arxiv.org/abs/physics/9701026v2`, 1997.

H. Nickisch. *Bayesian Inference and Experimental Design for Large Generalised Linear Models*. PhD thesis, Technische Universität Berlin, Berlin, Germany, 2010.

H. Nickisch. glm-ie: Generalised linear models inference and estimation toolbox. *Journal of Machine Learning Research*, 13:1699–1703, 13 2012.

H. Nickisch and C. Rasmussen. Approximations for binary gaussian process classification. *Journal of Machine Learning Research*, 9:2035–2078, 10 2008.

H. Nickisch and M. Seeger. Convex variational bayesian inference for large scale generalized linear models. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.

J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2006.

B. Olshausen and D. Field. Natural image statistics and efficient coding. *Network: Computation in Neural Systems*, 7:333–339, 2 1996.

M. Opper and C. Archambeau. The variational gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

M. Opper and O. Winther. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6:2177–2204, Dec. 2005.

J. Ormerod and M. Wand. Gaussian variational approximate inference for generalized linear mixed models. *Journal of Computational and Graphical Statistics*, 21(1):2–17, 2012.

A. Palmer, D. Wipf, K. Kreutz-Delgado, and B. Rao. Variational EM algorithms for non-Gaussian latent variable models. In *Advances in Neural Information Processing Systems 20*, 2006.

G. Papandreou and A. Yuille. Gaussian sampling by local perturbations. In *Advances in Neural Information Processing Systems 11*, 2010.

T. Park and G. Casella. The bayesian lasso. *Journal of the American Statistical Association*, 103: 681–686, 2008.

C. Rasmussen and H. Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, Nov. 2010.

C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

R. Salakhutdinov, S. Roweis, and Z. Ghahramani. Optimization with EM and expectation-conjugate-gradient. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

L. Saul, T. Jaakkola, and M. Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996.

M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *The 18th European Conference on Machine Learning*, 2007.

M. Seeger. Bayesian methods for support vector machines and gaussian processes. Master's thesis, University of Karlsruhe, 1999a.

M. Seeger. Bayesian model selection for support vector machines, gaussian processes and other kernel classifiers. In *Advances in Neural Information Processing Systems 12*. 1999b.

M. Seeger. Low rank updates for the cholsky decomposition. Technical report, University of California at Berkeley, `infoscience.epfl.ch/record/161468/files/cholupdate.pdf`, 2007.

M. Seeger. Bayesian inference and optimal design in the sparse linear model. *Journal of Machine Learning Research*, 9:759–813, Oct. 2008.

M. Seeger. Sparse linear models: Variational approximate inference and bayesian experimental design. *Journal of Physics: Conference Series*, 197(1), 2009.

M. Seeger. Gaussian covariance and scalable variational inference. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

M. Seeger and H. Nickisch. Compressed sensing and bayesian experimental design. In *Proceedings of the 25th International Conference on Machine Learning*, pages 912–919, 2008.

M. Seeger and H. Nickisch. Large scale variational inference and experimental design for sparse generalized linear models. Technical report, Max Planck Institute for Biological Cybernetics, //arxiv.org/abs/0810.0901, 2010.

M. Seeger and H. Nickisch. Fast convergent algorithms for expectation propagation approximate inference. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011a.

M. Seeger and H. Nickisch. Large scale bayesian inference and experimental design for sparse linear models. *SIAM Journal on Imaging Sciences*, 4(1):166–199, 2011b.

M. Seeger, S. Gerwinn, and M. Bethge. Bayesian inference for sparse generalized linear models. In *The 18th European Conference on Machine Learning*, 2007.

M. Tipping. Probabilistic visualisation of high-dimensional binary data. In *Advances in Neural Information Processing Systems 11*, 1999.

J. Vanhatalo, P. Jylnki, and A. Vehtari. Gaussian process regression with a student-t likelihood. In *Advances in Neural Information Processing Systems 22*, 2009.

D. Wipf. Sparse bayesian learning for basis selection. *IEEE Transactions on Signal Processing*, 52 (8):2153–2164, 2004.

Journal of Machine Learning Research 14 (2013) 2287-2314 Submitted 6/12; Revised 2/13; Published 8/13

# Message-Passing Algorithms for Quadratic Minimization

**Nicholas Ruozzi** NICHOLAS.RUOZZI@EPFL.CH
*Communication Theory Laboratory*
*École Polytechnique Fédérale de Lausanne*
*Lausanne 1015, Switzerland*

**Sekhar Tatikonda** SEKHAR.TATIKONDA@YALE.EDU
*Department of Electrical Engineering*
*Yale University*
*New Haven, CT 06520, USA*

**Editor:** Martin Wainwright

## Abstract

Gaussian belief propagation (GaBP) is an iterative algorithm for computing the mean (and variances) of a multivariate Gaussian distribution, or equivalently, the minimum of a multivariate positive definite quadratic function. Sufficient conditions, such as walk-summability, that guarantee the convergence and correctness of GaBP are known, but GaBP may fail to converge to the correct solution given an arbitrary positive definite covariance matrix. As was observed by Malioutov et al. (2006), the GaBP algorithm fails to converge if the computation trees produced by the algorithm are not positive definite. In this work, we will show that the failure modes of the GaBP algorithm can be understood via graph covers, and we prove that a parameterized generalization of the min-sum algorithm can be used to ensure that the computation trees remain positive definite whenever the input matrix is positive definite. We demonstrate that the resulting algorithm is closely related to other iterative schemes for quadratic minimization such as the Gauss-Seidel and Jacobi algorithms. Finally, we observe, empirically, that there always exists a choice of parameters such that the above generalization of the GaBP algorithm converges.

**Keywords:** belief propagation, Gaussian graphical models, graph covers

## 1. Introduction

Let $\Gamma \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix and $h \in \mathbb{R}^n$. The quadratic minimization problem is to find the $x \in \mathbb{R}^n$ that minimizes $f(x) = \frac{1}{2}x^T\Gamma x - h^T x$. Minimizing a positive definite quadratic function is equivalent to computing the mean of a multivariate Gaussian distribution with a positive definite covariance matrix, or equivalently, solving the positive definite linear system $\Gamma x = h$ for the vector $x$.

Because of the importance of solving linear systems, many different algorithms for the minimization of quadratic functions have been studied: Gaussian elimination, Gauss-Seidel iteration, Jacobi iteration, successive over-relaxation, etc. Distributed algorithms that can take advantage of the sparsity of the matrix $\Gamma$ are particularly desirable for solving large-scale systems as even Gaussian elimination may be too time consuming in practice. In this work, we study Gaussian graphical models and an inference algorithm known as Gaussian belief propagation (GaBP): an iterative message-passing scheme that can be used to estimate the mean of a Gaussian distribution as well as individual variances. In the study of graphical models, Gaussian graphical models are especially

©2013 Nicholas Ruozzi and Sekhar Tatikonda.

important because they represent one of the few continuous distributions for which exact message updates can be efficiently computed and more complicated distributions are often approximated by Gaussian distributions in practice.

In previous work, several authors have provided sufficient conditions for the convergence of GaBP. Weiss and Freeman (2001b) demonstrated that GaBP converges in the case that the covariance matrix is diagonally dominant. Malioutov et al. (2006) proved that the GaBP algorithm converges when the covariance matrix is walk-summable. Moallemi and Van Roy (2009, 2010) showed that scaled diagonal dominance was a sufficient condition for convergence and also characterized the rate of convergence via a computation tree analysis. The latter two sufficient conditions, walk-summability and scaled diagonal dominance, are known to be equivalent (Malioutov, 2008; Ruozzi et al., 2009).

While the above conditions are sufficient for the convergence of the GaBP algorithm they are not necessary: there are examples of positive definite matrices that are not walk-summable for which the GaBP algorithm still converges to the correct solution (Malioutov et al., 2006). A critical component of these examples is that the computation trees remain positive definite throughout the algorithm. Such behavior is guaranteed if the original matrix is scaled diagonally dominant, but arbitrary positive definite matrices can produce computation trees that are not positive definite (Malioutov et al., 2006). If this occurs, the standard GaBP algorithm fails to produce the correct solution. The purpose of this work is to understand why GaBP fails and to design iterative message-passing algorithms with improved convergence guarantees outside of the walk-summable case.

Related work studies the effect of preconditioning: the covariance matrix is preconditioned in order to force it to be scaled diagonally dominant and then the GaBP algorithm is applied to solve the preconditioned problem. Diagonal loading was proposed as one such useful preconditioner (Johnson et al., 2009). The key insight of diagonal loading is that scaled diagonal dominance can be achieved by sufficiently weighting the diagonal elements of the inverse covariance matrix. The diagonally loaded matrix can then be used as an input to a GaBP subroutine. The solution produced by GaBP is then used in a feedback loop to produce a new matrix, and the process is repeated until a desired level of accuracy is achieved. The performance is determined, in part, by the amount of diagonal loading needed to make the matrix scaled diagonally dominant. However, choosing the appropriate amount of diagonal loading that achieves the fastest rate of convergence remains an open question. As the approach in this work is to study reweighted versions of GaBP, our techniques can also be applied as part of this algorithmic scheme (or by themselves), and we provide experimental evidence that this technique results in performance gains over GaBP, allowing for less diagonal loading.

The GaBP algorithm can be seen as a special case of a standard message-passing algorithm known as the min-sum algorithm. Other recent work has studied provably convergent variants of the min-sum algorithm. The result has been the development of many different "convergent and correct" message-passing algorithms: MPLP (Globerson and Jaakkola, 2007), max-sum diffusion (Werner, 2007), norm-product belief propagation (Hazan and Shashua, 2010), and tree-reweighted belief propagation (Wainwright et al., 2005). Each of these algorithms can be viewed as a coordinate ascent/descent scheme for an appropriate lower/upper bound. Sontag and Jaakkola (2009) and Meltzer et al. (2009) provide a general overview of these techniques and their relationship to bound maximization. These algorithms guarantee convergence under an appropriate message-passing schedule, and they also guarantee correctness if a unique assignment can be extracted upon convergence. Such algorithms are plausible candidates in the search for convergent iterative algo-

rithms for the quadratic minimization problem, but as we show using the theory of graph covers, iterative schemes based on dual optimization techniques that guarantee the correctness of locally decodable beliefs cannot converge to the correct minimizing assignment outside of walk-summable models.

In this work, we investigate the behavior of reweighted message-passing algorithms for the quadratic minimization problem. The motivation for this study comes from the observation that belief propagation style algorithms typically do not explore all nodes in the factor graph with the same frequency (Frey et al., 2001). In many application areas, such uneven counting is undesirable and typically results in incorrect answers, but if we can use reweighting to overestimate the diagonal entries of the computation tree relative to the off diagonal entries, then we may be able to force the computation trees to be positive definite at each iteration of the algorithm. Although similar in spirit to diagonal loading, our approach obviates the need for preconditioning. We will show that there exists a choice of parameters for the reweighted algorithms that guarantees monotone convergence of the variance estimates on all positive definite models, even those for which the GaBP algorithm fails to converge. We empirically observe that there exists a choice of parameters that also guarantees the convergence of the mean estimates. In addition, we show that our graph cover analysis extends to other iterative algorithms for the quadratic minimization problem and that similar ideas can be used to reason about the min-sum algorithm for general convex minimization.

The outline of this paper is as follows. In Section 2 we review the min-sum algorithm, its reweighted generalizations, and the quadratic minimization problem. In Section 3 we discuss the relationship between pairwise message-passing algorithms and graph covers, and we show how to use graphs covers to characterize walk-summability. In Section 4, we examine the convergence of the means and the variances under the reweighted algorithm for the quadratic minimization problem, we explore the relationship between the reweighted algorithm and the Gauss-Seidel and Jacobi methods, and we compare the performance of the reweighted algorithm to the standard min-sum algorithm. Finally, in Section 5, we summarize the results and discuss extensions of this work to general convex functions as well as open problems. Detailed proofs of the two main theorems can be found in Appendices A and B.

## 2. Preliminaries

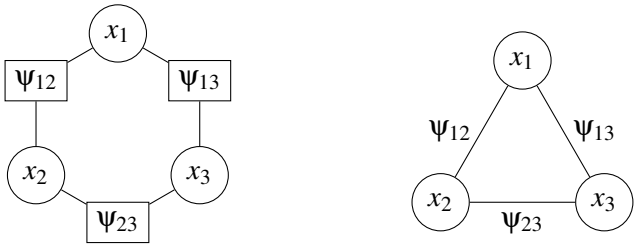
In this section, we review the min-sum algorithm and a reweighted variant over pairwise factor graphs. Of particular importance for later proofs will be the computation trees generated by each of these algorithms. We also review the quadratic minimization problem, and explain the closed-form message updates for this problem.

### 2.1 The Min-Sum Algorithm

The min-sum algorithm attempts to compute the minimizing assignment of an objective function $f : \prod_i \mathcal{X}_i \to \mathbb{R}$ that, given a graph $G = (V,E)$, can be factorized as a sum of self-potentials and edge potentials as follows.

$$f(x_1,...,x_n) = \sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E} \psi_{ij}(x_i,x_j)$$

We assume that this minimization problem is well-defined: $f$ is bounded from below and there exists an $x \in \prod_i \mathcal{X}_i$ that minimizes $f$.




(a) Factor graph. (b) The graph $G$.

Figure 1: The factor graph corresponding to $f(x_1,x_2,x_3) = \phi_1 + \phi_2 + \phi_3 + \psi_{12} + \psi_{23} + \psi_{13}$ and the graph $G$. The functions $\phi_1$,$\phi_2$, and $\phi_3$ each depend on only one variable, and are typically omitted from the factor graph representation for clarity.

To each factorization, we associate a bipartite graph known as the factor graph. In general, the factor graph consists of a node for each of the variables $x_1,...,x_n$, a node for each of the $\psi_{ij}$, and for all $(i,j) \in E$, an edge joining the node corresponding to $x_i$ to the node corresponding to $\psi_{ij}$. Because the $\psi_{ij}$ each depend on exactly two factors, we often omit the factor nodes from the factor graph construction and replace them with a single edge. This reduces the factor graph to the graph $G$. See Figure 1 for an example of this construction.

We can write the min-sum algorithm as a local message-passing algorithm over the graph $G$. During the execution of the min-sum algorithm, messages are passed back and forth between adjacent nodes of the graph. On the $t^{th}$ iteration of the algorithm, messages are passed along each edge of the factor graph as

$$m^t_{i\to j}(x_j) \quad = \quad \kappa + \min_{x_i}\left[\psi_{ij}(x_i,x_j) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} m^{t-1}_{k\to i}(x_i)\right],$$

where $\partial i$ denotes the set of neighbors of the node $i$ in $G$ and $\partial j \setminus i$ is abusive notation for the set-theoretic difference $\partial j \setminus \{i\}$. When the factor graph is a tree, these updates are guaranteed to converge, but understanding when these updates converge to the correct solution for an arbitrary graph is a central question underlying the study of the min-sum algorithm.

Each message update has an arbitrary normalization factor $\kappa$. Because $\kappa$ is not a function of any of the variables, it only affects the value of the minimum and not where the minimum is located. As such, we are free to choose it however we like for each message and each time step. In practice, these constants are used to avoid numerical issues that may arise during the execution of the algorithm.

We will think of the messages as a vector of functions indexed by the edge over which the message is passed. Any vector of real-valued messages is a valid choice for the vector of initial messages $m^0$, and the choice of initial messages can greatly affect the behavior of the algorithm. A typical assumption, that we will use in this work, is that the initial messages are chosen such that $m^0_{i\to j} \equiv 0$ for all $i$ and $j$.

Given any vector of messages, $m^t$, we can construct a set of beliefs that are intended to approximate the min-marginals of $f$ as

$$\begin{aligned}\tau^t_i(x_i) &= \kappa + \phi_i(x_i) + \sum_{j\in\partial i} m^t_{j\to i}(x_i),\\ \tau^t_{ij}(x_i,x_j) &= \kappa + \psi_{ij}(x_i,x_j) + \tau^t_j(x_j) - m^t_{i\to j}(x_j) + \tau^t_i(x_i) - m^t_{j\to i}(x_i).\end{aligned}$$

Additionally, we can approximate the optimal assignment by computing an estimate of the argmin,

$$x_i^t \quad \in \quad \arg\min_{x_i} \tau_i^t(x_i).$$

If the beliefs are equal to the true min-marginals of $f$ (i.e., $\tau_i^t(x_i) = \min_{x' : x'_i = x_i} f(x')$), then for any $y_i \in \arg\min_{x_i} \tau_i^t(x_i)$ there exists a vector $x^*$ such that $x_i^* = y_i$ and $x^*$ minimizes the function $f$. If $|\arg\min_{x_i} \tau_i^t(x_i)| = 1$ for all $i$, then we can take $x^* = y$, but, if the objective function has more than one optimal solution, then we may not be able to construct such an $x^*$ so easily.

**Definition 1** *A vector, $\tau = (\{\tau_i\}, \{\tau_{ij}\})$, of beliefs is **locally decodable** to $x^*$ if $\tau_i(x_i^*) < \tau_i(x_i)$ for all $i$, $x_i \neq x_i^*$. Equivalently, for each $i \in V$, $\tau_i$ is uniquely minimized at $x_i^*$.*

If the algorithm converges to a vector of beliefs that are locally decodable to $x^*$, then we hope that the vector $x^*$ is a global minimum of the objective function. This is indeed the case when the factor graph contains no cycles (Wainwright et al., 2004) but need not be the case for arbitrary graphical models.

### 2.1.1 COMPUTATION TREES

An important tool in the analysis of the min-sum algorithm is the notion of a computation tree. Intuitively, the computation tree is an unrolled version of the original graph that captures the evolution of the messages passed by the min-sum algorithm needed to compute the belief at time $t$ at a particular node of the factor graph. Computation trees describe the evolution of the beliefs over time, which, in some cases, can help us prove correctness and/or convergence of the message-passing updates. For example, the convergence of the min-sum algorithm on graphs containing a single cycle can be demonstrated by analyzing the computation trees produced by the min-sum algorithm at each time step (Weiss, 2000).

The depth $t$ computation tree rooted at node $i$ contains all of the length $t$ non-backtracking walks in the factor graph starting at node $i$. A walk is non-backtracking if it does not go back and forth successively between two vertices. For any node $v$ in the factor graph, the computation tree at time $t$ rooted at $i$, denoted by $T_i(t)$, is defined recursively as follows. $T_i(0)$ is just the node $i$, the root of the tree. The tree $T_i(t)$ at time $t > 0$ is generated from $T_i(t-1)$ by adding to each leaf of $T_i(t-1)$ a copy of each of its neighbors in $G$ (and the corresponding edge), except for the neighbor that is already present in $T_i(t-1)$. Each node of $T_i(t)$ is a copy of a node in $G$, and the potentials on the nodes in $T_i(t)$, which operate on a subset of the variables in $T_i(t)$, are copies of the potentials of the corresponding nodes in $G$. The construction of a computation tree for the graph in Figure 1 is pictured in Figure 2. Note that each variable node in $T_i(t)$ represents a distinct copy of some variable $x_j$ in the original graph.

Given any initialization of the messages, $T_i(t)$ captures the information available to node $i$ at time $t$. At time $t = 0$, node $i$ has received only the initial messages from its neighbors, so $T_i(0)$ consists only of $i$. At time $t = 1$, $i$ receives the round one messages from all of its neighbors, so $i$'s neighbors are added to the tree. These round one messages depend only on the initial messages, so the tree terminates at this point. By construction, we have the following lemma.

**Lemma 2** *The belief at node $i$ produced by the min-sum algorithm at time $t$ corresponds to the exact min-marginal at the root of $T_i(t)$ whose boundary messages are given by the initial messages.*

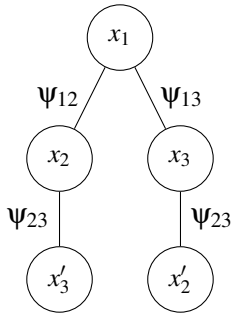



Figure 2: The computation tree at time $t = 2$ rooted at the variable node $x_1$ of the graph in Figure 1. The self-potentials corresponding to each variable node are given by the subscript of the variable.

**Proof** Tatikonda and Jordan (2002) and Weiss and Freeman (2001a) both provide a proof of this lemma. ■

Computation trees provide a dynamic view of the min-sum algorithm. After a finite number of time steps, we hope that the beliefs at the root of the computation trees stop changing and that the message vector converges to a fixed point of the message update equations (in practice, when the beliefs change by less than some small amount, we say that the algorithm has converged). For any real-valued objective function $f$ (i.e., $|f(x)| < \infty$ for all $x$), there always exists a fixed point of the message update equations (see Theorem 2 of Wainwright et al. 2004).

### 2.2 Reweighted Message-Passing Algorithms

Because the min-sum algorithm is not guaranteed to converge and, even it does, is not guaranteed to compute the correct minimizing assignment, recent research has focused on the design of alternative message-passing schemes that do not suffer from these drawbacks. Efforts to produce provably convergent message-passing schemes have resulted in the reweighted message-passing algorithm described in Algorithm 1. This algorithm is parameterized by a vector of non-zero real weights for each edge of the graph. Notice that if we set $c_{ij} = 1$ for all $i$ and $j$, then we obtain the standard min-sum algorithm. Wainwright et al. (2005) choose the $c_{ij}$ in a specific way in order to guarantee correctness of the algorithm (which they call TRMP in this special case). In this work, we will focus on choices of these weights that will guarantee convergence of the algorithm for the quadratic minimization problem. These choices will, surprisingly, not coincide with those of the TRMP algorithm. In fact, the choice of weights that guarantees correctness of the TRMP algorithm must necessarily cause the algorithm to either not converge or converge to the incorrect solution whenever the given matrix is not walk-summable.

The beliefs for the reweighted algorithm are defined analogously to those for the standard min-sum algorithm.

$$\begin{aligned}
\tau_i^t(x_i) &= \kappa + \phi_i(x_i) + \sum_{j \in \partial i} c_{ji} m_{j\to i}^t(x_i), \\
\tau_{ij}^t(x_i, x_j) &= \kappa + \frac{\psi_{ij}(x_i, x_j)}{c_{ij}} + \tau_j^t(x_j) - m_{i\to j}^t(x_j) + \tau_i^t(x_i) - m_{j\to i}^t(x_i)
\end{aligned}$$

**Algorithm 1** Synchronous Reweighted Message-Passing Algorithm

1: Initialize the messages to some finite vector.
2: For iteration $t = 1, 2, \ldots$ update the the messages as follows:

$$m_{i\to j}^t(x_j) := \kappa + \min_{x_i}\left[\frac{\psi_{ij}(x_i,x_j)}{c_{ij}} + \phi_i(x_i) + (c_{ij}-1)m_{j\to i}^{t-1}(x_i) + \sum_{k\in\partial i\setminus j} c_{ki}m_{k\to i}^{t-1}(x_i)\right].$$

The vector of messages at any fixed point of the message update equations has two important properties. First, the beliefs corresponding to these messages provide an alternative factorization of the objective function $f$. Second, the beliefs correspond to approximate marginals.

**Lemma 3** *For any vector of messages $m^t$ with corresponding beliefs $\tau^t$,*

$$f(x_1,\ldots,x_{|V|}) = \kappa + \sum_{i\in V}\tau_i^t(x_i) + \sum_{(i,j)\in E} c_{ij}\left[\tau_{ij}^t(x_i,x_j) - \tau_i^t(x_i) - \tau_j^t(x_j)\right].$$

**Lemma 4** *If $\tau$ is a set of beliefs corresponding to a fixed point of the message updates in Algorithm 1, then*

$$\min_{x_j}\tau_{ij}(x_i,x_j) = \kappa + \tau_i(x_i)$$

*for all $(i,j)\in G$ and all $x_i$.*

The proof of these two lemmas is a straightforward exercise in applying the definitions. Wainwright et al. (2004) provide similar results for the special case of the max-product algorithm.

### 2.2.1 COMPUTATION TREES

The computation trees produced by Algorithm 1 are different from their predecessors. Again, the computation tree captures the messages that would need to be passed in order to compute $\tau_i^t(x_i)$. However, the messages that are passed in the new algorithm are multiplied by a non-zero constant. As a result, the potential at a node $u$ in the computation tree corresponds to some potential in the original graph multiplied by a constant that depends on all of the nodes above $u$ in the computation tree. We summarize the changes as follows.

1. The message passed from $i$ to $j$ may now depend on the message from $j$ to $i$ at the previous time step. As such, we now form the time $t+1$ computation tree from the time $t$ computation tree by taking any leaf $u$, which is a copy of node $v$ in the factor graph, of the time $t$ computation tree, creating a new node for every $w\in\partial v$, and connecting $u$ to these new nodes. As a result, the new computation tree rooted at node $u$ of depth $t$ contains at least all of the non-backtracking walks of length $t$ in the factor graph starting from $u$ and, at most, all walks of length $t$ in the factor graph starting at $u$.

2. The messages are weighted by the elements of $c$. This changes the potentials at the nodes in the computation tree. For example, suppose the computation tree was rooted at variable node $i$ and that $\tau_i$ depends on the message from $j$ to $i$. Because $m_{ji}$ is multiplied by $c_{ij}$ in $\tau_i$,

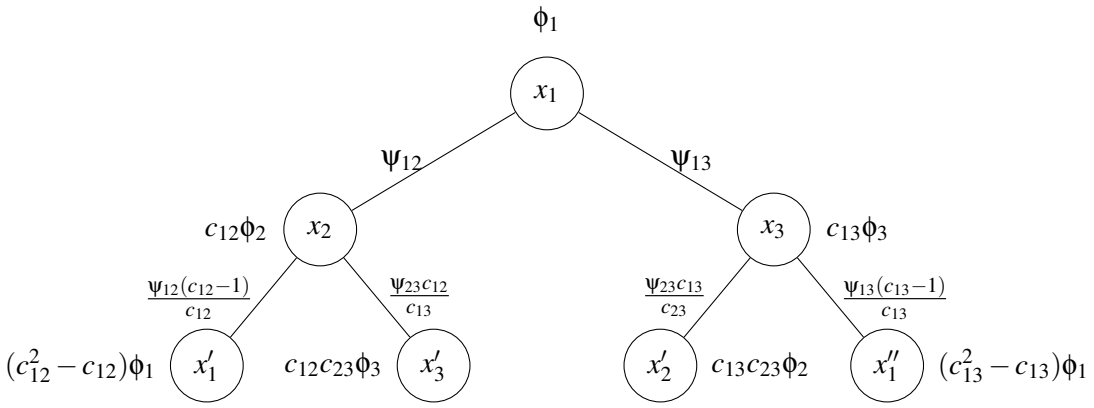



Figure 3: Construction of the computation tree rooted at node $x_1$ at time $t = 2$ produced by Algorithm 1 for the factor graph in Figure 1. Self-potentials are adjacent to the variable node to which they correspond. One can check that setting $c_{ij} = 1$ for all $(i, j) \in E$ reduces the above computation tree to that of Figure 2.

every potential along this branch of the computation tree is multiplied by $c_{ij}$. To make this concrete, we can associate a weight to every edge of the computation tree that corresponds to the constant that multiplies the message passed across that edge. To compute the new potential at a variable node $i$ in the computation tree, we now need to multiply the corresponding potential $\phi_i$ by each of the weights corresponding to the edges that appear along the path from $i$ to the root of the computation tree. An analogous process can be used to compute the potentials on each of the edges. The computation tree produced by Algorithm 1 at time $t = 2$ for the factor graph in Figure 1 is pictured in Figure 3. Compare this with computation tree produced by the standard min-sum algorithm in Figure 2.

If we make these adjustments, then the belief, $\tau_i^t(x_i)$, at node $i$ at time $t$ is given by the min-marginal at the root of $T_i(t)$. In this way, the beliefs correspond to marginals at the root of these computation trees.

### 2.3 Quadratic Minimization

We now address the quadratic minimization problem in the context of the reweighted min-sum algorithm. Recall that given a matrix $\Gamma$ the quadratic minimization problem is to find the vector $x$ that minimizes $f(x) = \frac{1}{2}x^T\Gamma x - h^T x$. Without loss of generality, we can assume that the matrix $\Gamma$ is symmetric as the quadratic function $\frac{1}{2}x^T\Gamma x - h^T x$ is equivalent to $\frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)\right]x - h^T x$ for any

$\Gamma \in \mathbb{R}^{n\times n}$.

$$\begin{aligned} f(x) &= \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)+\frac{1}{2}(\Gamma-\Gamma^T)\right]x - h^T x \\ &= \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)\right]x + \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma-\Gamma^T)\right]x - h^T x \\ &= \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)\right]x - h^T x. \end{aligned}$$

Every quadratic function admits a pairwise factorization

$$\begin{aligned} f(x_1,...,x_n) &= \frac{1}{2}x^T\Gamma x - h^T x \\ &= \sum_i [\frac{1}{2}\Gamma_{ii}x_i^2 - h_i x_i] + \sum_{i>j}\Gamma_{ij}x_i x_j, \end{aligned}$$

where $\Gamma \in \mathbb{R}^{n\times n}$ is a symmetric matrix. We note that we will abusively write min in the reweighted update equations even though the appropriate notion of minimization for the real numbers is inf.

We can explicitly compute the minimization required by the reweighted min-sum algorithm at each time step: the synchronous message update $m_{i\to j}^t(x_j)$ can be parameterized as a quadratic function of the form $\frac{1}{2}a_{i\to j}^t x_j^2 + b_{i\to j}^t x_j$. If we define

$$A_{i\backslash j}^t \triangleq \left[\Gamma_{ii} + \sum_{k\in\partial i} c_{ki}\cdot a_{k\to i}^{t-1}\right] - a_{j\to i}^{t-1}$$

and

$$B_{i\backslash j}^t \triangleq \left[h_i - \sum_{k\in\partial i} c_{ki}\cdot b_{k\to i}^{t-1}\right] - b_{j\to i}^{t-1},$$

then the updates at time $t$ are given by

$$\begin{aligned} a_{i\to j}^t &:= \frac{-\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{A_{i\backslash j}^t}, \\ b_{i\to j}^t &:= \frac{B_{i\backslash j}^t\frac{\Gamma_{ij}}{c_{ij}}}{A_{i\backslash j}^t}. \end{aligned}$$

These updates are only valid when $A_{i\backslash j} > 0$. If this is not the case, then the minimization given in Algorithm 1 is not bounded from below, and we set $a_{i\to j}^t = -\infty$. For the initial messages, we set $a_{i\to j}^0 = b_{i\to j}^0 = 0$.

Suppose that the beliefs generated from a fixed point of Algorithm 1 are locally decodable to $x^*$. One can show that the gradient of $f$ at $x^*$ is always equal to zero. If the gradient of $f$ at $x^*$ is zero and $\Gamma$ is positive definite, then $x^*$ must be a global minimum of $f$. In other words, the min-sum algorithm always computes the correct minimizing assignment if it converges to locally decodable beliefs. This result was previously proven for the GaBP algorithm (Weiss and Freeman, 2001b) and the tree-reweighted algorithm (Wainwright et al., 2003a).

**Theorem 5** *If Algorithm 1 converges to a collection of beliefs, $\tau$, that are locally decodable to $x^*$ for a quadratic function $f$, then $x^*$ is a local minimum of $f$.*

**Proof** For completeness, we sketch the proof. By Lemmas 3 and 4, we have that,

$$\min_{x_j} \tau_{ij}(x_i, x_j) = \kappa + \tau_i(x_i)$$

for all $(i, j) \in G$ and

$$f(x_1, \ldots, x_{|V|}) = \kappa + \sum_{i \in V} \tau_i(x_i) + \sum_{(i,j) \in E} c_{ij} \Big[ \tau_{ij}(x_i, x_j) - \tau_i(x_i) - \tau_j(x_j) \Big].$$

If $\tau$ is locally decodable to $x^*$, then for each $i \in V$, $\tau_i(x_i)$ must be a positive definite quadratic function that is minimized at $x_i^*$. Applying Lemma 4, we have that for each $(i, j) \in E$, $\tau_{ij}$ is also a positive definite quadratic function and $\tau_{ij}$ is minimized at $(x_i^*, x_j^*)$. For each $i \in V$,

$$\frac{d}{dx_i} f(x_1, \ldots, x_{|V|}) \quad = \quad \frac{d}{dx_i} \tau_i(x_i) + \sum_{j \in \partial i} c_{ij} \Big[ \frac{d}{dx_i} \tau_{ij}(x_i, x_j) - \frac{d}{dx_i} \tau_i(x_i) \Big].$$

By the above arguments, for each $i \in V$, $\frac{d}{dx_i} \tau_i(x_i) \Big|_{x^*} = 0$. Similarly, for all $(i, j) \in E$, $\frac{d}{dx_i} \tau_{ij}(x_i, x_j) \Big|_{x^*} = 0$. As a result, we must have $\nabla f(x_1^*, \ldots, x_{|V|}^*) = 0$. If $\Gamma$ is positive semidefinite, then $f$ is convex and $x^*$ must be a global minimum of $f$. ■

As a consequence of Theorem 5, even if $\Gamma$ not positive definite, if some fixed point of the reweighted algorithm is locally decodable to a vector $x^*$ then, $x^*$ solves the system $\Gamma x = h$.

As discussed in the introduction, the GaBP algorithm is known to converge under certain conditions on the matrix $\Gamma$. Consider the following definitions.

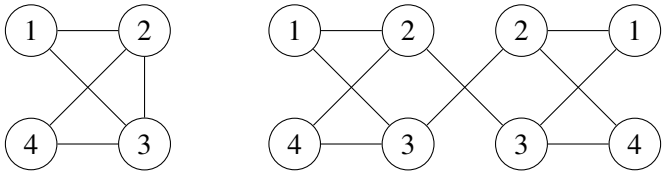
**Definition 6** $\Gamma \in \mathbb{R}^{n \times n}$ *is* ***scaled diagonally dominant*** *if* $\exists w > 0 \in \mathbb{R}^n$ *such that* $|\Gamma_{ii}| w_i > \sum_{j \neq i} |\Gamma_{ij}| w_j$.

**Definition 7** $\Gamma \in \mathbb{R}^{n \times n}$ *is* ***walk-summable*** *if the spectral radius* $\rho(|I - D^{-1/2} \Gamma D^{-1/2}|) < 1$. *Here,* $D^{-1/2}$ *is the diagonal matrix such that* $D_{ii}^{-1/2} = \frac{1}{\sqrt{\Gamma_{ii}}}$, *and* $|A|$ *denotes the matrix obtained from the matrix* $A$ *by taking the absolute value of each entry of* $A$.

For any matrix $\Gamma$ with strictly positive diagonal, Weiss and Freeman (2001b) demonstrated that GaBP converges when $\Gamma$ is diagonally dominant (scaled diagonally dominant with all scale factors equal to one), Malioutov et al. (2006) proved that the GaBP algorithm converges when $\Gamma$ is walk-summable, and Moallemi and Van Roy (2009, 2010) showed that GaBP converges when $\Gamma$ is scaled diagonally dominant. We would like to understand how to choose the parameters of the reweighted algorithm in order to extend the convergence results for GaBP to all positive definite matrices.

## 3. Graph Covers

In this section, we will explore graph covers and their relationship to iterative message-passing algorithms for the quadratic minimization problem. Before addressing the quadratic minimization problem specifically, we will first make a few observations about general pairwise graphical models. The greatest strength of the above message-passing algorithms, their reliance on only local information, can also be a weakness: local message-passing algorithms are incapable of distinguishing two graphs that have the same local structure. To make this precise, we will need the notion of graph covers.




(a) A graph $G$ (b) A 2-cover of $G$

Figure 4: An example of a graph cover. Nodes in the cover are labeled by the node that they are a copy of in $G$.

**Definition 8** *A graph $H$ **covers** a graph $G$ if there exists a graph homomorphism $\pi : H \to G$ such that $\pi$ is an isomorphism on neighborhoods (i.e., for all vertices $i \in H$, $\partial i$ is mapped bijectively onto $\partial\pi(i)$). If $\pi(i) = j$, then we say that $i \in H$ is a copy of $j \in G$. Further, $H$ is an $M$-cover of $G$ if every vertex of $G$ has exactly $M$ copies in $H$.*

Graph covers, in the context of graphical models, were originally studied in relation to local message-passing algorithms for coding problems (Vontobel and Koetter, 2005). Graph covers may be connected (i.e., there is a path between every pair of vertices) or disconnected. However, when a graph cover is disconnected, all of the connected components of the cover must themselves be covers of the original graph. For a simple example of a connected graph cover, see Figure 4.

Every finite cover of a connected graph is an $M$-cover for some integer $M$. For every base graph $G$, there exists a graph, possibly infinite, which covers all finite, connected covers of the base graph. This graph is known as the universal cover.

To any finite cover, $H$, of a factor graph $G$ we can associate a collection of potentials derived from the base graph; the potential at node $i \in H$ is equal to the potential at node $\pi(i) \in G$. Together, these potential functions define a new objective function for the factor graph $H$. In the sequel, we will use superscripts to specify that a particular object is over the factor graph $H$. For example, we will denote the objective function corresponding to a factor graph $H$ as $f^H$, and we will write $f^G$ for the objective function $f$.

Local message-passing algorithms such as the reweighted min-sum algorithm are incapable of distinguishing the two factor graphs $H$ and $G$ given that the initial messages to and from each node in $H$ are identical to the nodes that they cover in $G$: for every node $i \in G$ the messages received and sent by this node at time $t$ are exactly the same as the messages sent and received at time $t$ by any copy of $i$ in $H$. As a result, if we use a local message-passing algorithm to deduce an assignment for $i$, then the algorithm run on the graph $H$ must deduce the same assignment for each copy of $i$.

Now, consider an objective function $f$ that factors over the graph $G$. For any finite cover $H$ of $G$ with covering homomorphism $\pi : H \to G$, we can "lift" any vector of beliefs, $\tau^G$, from $G$ to $H$ by defining a new vector of beliefs, $\tau^H$, such that:

- For all variable nodes $i \in H$, $\tau^H_i = \tau^G_{\pi(i)}$.
- For all edges $(i, j) \in H$, $\tau^H_{ij} = \tau^G_{\pi(i)\pi(j)}$.

Analogously, we can lift any assignment $x^G$ to an assignment $x^H$ by setting $x^H_i = x^G_{\pi(i)}$.

### 3.1 Graph Covers and Quadratic Minimization

Let $G$ be the pairwise factor graph for the objective function $f^G(x_1,...,x_n) = \frac{1}{2}x^T\Gamma x - h^T x$ whose edges correspond to the nonzero entries of $\Gamma$. Let $H$ be an $M$-cover of $G$ with corresponding objective function $f^H(x_{11},...,x_{1M},...x_{nM}) = \frac{1}{2}x^T\widetilde{\Gamma}x - \widetilde{h}^T x$. Without loss of generality we can assume that $\widetilde{\Gamma}$ and $\widetilde{h}$ take the following form:

$$\begin{aligned} \widetilde{\Gamma} &= \begin{pmatrix} \Gamma_{11}P_{11} & \cdots & \Gamma_{1n}P_{1n} \\ \vdots & \ddots & \vdots \\ \Gamma_{n1}P_{n1} & \cdots & \Gamma_{nn}P_{nn} \end{pmatrix}, \\ \widetilde{h}_i &= h_{\lceil i/M \rceil}, \end{aligned} \tag{1}$$

where $P_{ij} = P_{ji}^T$ is an $M \times M$ permutation matrix for all $i \neq j$ and $P_{ii}$ is the $M \times M$ identity matrix for all $i$. If $\widetilde{\Gamma}$ is derived from $\Gamma$ in this way, then we will say that $\widetilde{\Gamma}$ covers $\Gamma$.

For the quadratic minimization problem, factor graphs and their covers share many of the same properties. Most notably, we can transform critical points of covers to critical points of the original problem. Let $H$ and $G$ be as above, and let $\pi$ be the graph homomorphism from $H$ to $G$. For $x \in \mathbb{R}^{|V_G|}$, define $\text{lift}_H : \mathbb{R}^{|V_G|} \to \mathbb{R}^{|V_H|}$ such that

$$\text{lift}_H(x)_i = x_{\pi(i)}$$

for all $i \in H$. Similarly, for each $y \in \mathbb{R}^{|V_H|}$, define $\text{proj}_G \mathbb{R}^{|V_G|} \to \mathbb{R}^{|V_G|}$ such that

$$\text{proj}(y)_i = \sum_{k \in H : \pi(k) = i} \frac{y_k}{|\{j \in H : \pi(j) = i\}|}$$

for all $i \in G$. With these definitions, we have the following lemma.

**Lemma 9** *If $\widetilde{\Gamma}y = \widetilde{h}$ for $y \in \mathbb{R}^{|V_H|}$, then $\Gamma \cdot proj_G(y) = h$. Conversely, if $\Gamma x = h$ for $y \in \mathbb{R}^{|V_G|}$, then $\widetilde{\Gamma} \cdot lift_H(x) = \widetilde{h}$.*

Notice that these solutions correspond to critical points of the cover and the original problem. Similarly, we can transform eigenvectors of covers to either eigenvectors of the original problem or the zero vector.

**Lemma 10** *Fix $\lambda \in \mathbb{R}$. If $\widetilde{\Gamma}y = \lambda y$, then either $\Gamma \cdot proj_G(y) = \lambda proj_G(y)$ or $\Gamma \cdot proj_G(y) = 0$. Conversely, if $\Gamma x = \lambda x$, then $\widetilde{\Gamma} \cdot lift_H(x) = \lambda lift_H(x)$.*

These lemmas demonstrate that we can average critical points and eigenvectors of covers to obtain critical points and eigenvectors (or the zero vector) of the original problem, and we can lift critical points and eigenvectors of the original problem in order to obtain critical points and eigenvectors of covers.

Unfortunately, even though the critical points of $G$ and its covers must correspond via Lemma 9, the corresponding minimization problems may not have the same solution. The example in Figure 5 illustrates that there exist positive definite matrices that are covered by matrices that are not positive definite. This observation seems to be problematic for the convergence of iterative message-passing schemes. Specifically, the fixed points of the reweighted algorithm on the base graph are also fixed

$$\Gamma = \begin{pmatrix} 1 & .6 & .6 \\ .6 & 1 & .6 \\ .6 & .6 & 1 \end{pmatrix} \quad \widetilde{\Gamma} = \begin{pmatrix} 1 & 0 & .6 & 0 & 0 & .6 \\ 0 & 1 & 0 & .6 & .6 & 0 \\ .6 & 0 & 1 & 0 & .6 & 0 \\ 0 & .6 & 0 & 1 & 0 & .6 \\ 0 & .6 & .6 & 0 & 1 & 0 \\ .6 & 0 & 0 & .6 & 0 & 1 \end{pmatrix}$$

Figure 5: An example of a positive definite matrix, $\Gamma$, which possesses a 2-cover, $\widetilde{\Gamma}$, that has negative eigenvalues.

points of the reweighted algorithm on any graph cover. As such, the reweighted algorithm may not converge to the correct minimizing assignment when the matrix corresponding to some cover of $G$ is not positive definite. Consequently, we will first consider the special case in which $\Gamma$ and all of its covers are positive definite. We can exactly characterize the matrices for which this property holds.

**Theorem 11** *Let $\Gamma$ be a symmetric matrix with positive diagonal. The following are equivalent.*

1. *$\Gamma$ is walk-summable.*
2. *$\Gamma$ is scaled diagonally dominant.*
3. *All covers of $\Gamma$ are positive definite.*
4. *All 2-covers of $\Gamma$ are positive definite.*

**Proof** The two non-trivial implications in the proof ($4 \Rightarrow 1$ and $1 \Rightarrow 2$) make use of the Perron-Frobenius theorem. For the complete details, see Appendix A. ■

This theorem has several important consequences. First, it provides us with a combinatorial characterization of scaled diagonal dominance and walk-summability. Second, it provides an intuitive explanation for why these conditions should be sufficient for the convergence of local message-passing algorithms.

More importantly, we can use Theorem 11 to conclude that MPLP, tree-reweighted max-product, and other message-passing algorithms that guarantee the correctness of locally decodable beliefs cannot converge to the correct solution when $\Gamma$ is positive definite but not walk-summable. From the discussion in Section 3, every collection of locally decodable beliefs on the base graph can be lifted to locally decodable beliefs on any graph cover. Each of these "convergent and correct" message-passing algorithms guarantees that the lift of $x^*$ to each graph cover must be a global minimum on that cover. By Theorem 11, there exists at least one graph cover with no global minimum. As a result, these algorithms cannot converge to locally decodable beliefs.

As we saw in Theorem 5, the reweighted message-passing algorithm only guarantees that $x^*$ is a local optimum. However, there exist simple choices for the reweighting parameters that guarantee correctness over all covers. As an example, if $c_{ij} \leq \frac{1}{\max_{i \in V} |\partial i|}$ for all $(i, j) \in E$, then one can show that the reweighted algorithm cannot converge to locally decodable beliefs unless all of the graph covers are convex. The traditional choice of parameters for the TRMP algorithm where each $c_{ij}$

corresponds to an edge appearance probability provides another example. Given this observation, in order to produce convergent message-passing schemes for the quadratic minimization problem, we will need to study choices of the parameters that do not guarantee correctness over all graph covers.

## 4. Convergence Properties of Reweighted Message-Passing Algorithms

Recall that the GaBP algorithm can converge to the correct minimizer of the objective function even if the original matrix is not scaled diagonally dominant. The most significant problem when the original matrix is positive definite but not scaled diagonally dominant is that the computation trees may eventually possess negative eigenvalues due to the existence of some 2-cover with at least one non-positive eigenvalue. If this happens, then some of the beliefs will not be bounded from below, and the corresponding estimate will be negative infinity. This is, of course, the correct answer on some 2-cover of the problem, but it is not the correct solution to the minimization problem of interest. Our goal in this section is to understand how the choice of the parameters and alternative message-passing orders affect the convergence of the reweighted algorithm.

### 4.1 Convergence of the Variances

First, we will provide conditions on the choice of the parameter vector such that all of the computation trees produced by the reweighted algorithm remain positive definite throughout the course of the algorithm. Positive definiteness of the computation trees corresponds to the convexity of the beliefs, and the convexity of the belief, $\tau_i^t$, is determined only by the vector $a^t$. As such, we begin by studying the sequence $a^0, a^1, \ldots$ where $a^0$ is the zero vector (based on our initialization). We will consider two different choices for the parameter vector: one in which $c_{ij} \geq 1$ for all $i$ and $j$ and one in which $c_{ij} < 0$ for all $i$ and $j$. The latter of these two requires all of the parameters to be negative. Such a choice is unusual among reweighted algorithms, but it does guarantee that the computation trees will remain positive definite throughout the algorithm. This observation and the experimental results in Section 4.4 suggest that such a choice might be worth studying in other contexts as well.

#### 4.1.1 POSITIVE PARAMETERS

**Lemma 12** *If $c_{ij} \geq 1$ for all $i$ and $j$, then for all $t > 0$, $a^t_{i\to j} \leq a^{t-1}_{i\to j} \leq 0$ for each $i$ and $j$.*

**Proof** This result follows by induction on $t$. First, suppose that $c_{ij} \geq 1$. If the update is not valid, then $a^t_{i\to j} = -\infty$ which trivially satisfies the inequality. Otherwise, we have

$$\begin{aligned} a^t_{i\to j} &= \frac{-\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + (c_{ji}-1)a^{t-1}_{j\to i}} \\ &\leq \frac{\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-2}_{k\to i} + (c_{ji}-1)a^{t-2}_{j\to i}} \\ &= a^{t-1}_{i\to j}, \end{aligned}$$

where the inequality follows from the observation that $\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + (c_{ji}-1)a^{t-1}_{j\to i} > 0$ and the induction hypothesis. ■

$$\begin{pmatrix} 1 & 0.39866 & -0.39866 & -0.39866 \\ 0.39866 & 1 & -0.39866 & 0 \\ -0.39866 & -0.39866 & 1 & -0.39866 \\ -0.39866 & 0 & -0.39866 & 1 \end{pmatrix}$$

Figure 6: A positive definite matrix for which the variances in the min-sum algorithm converge but the means do not (Malioutov, 2008).

If we consider only the vector $a^t$, then the algorithm may exhibit a weaker form of convergence.

**Lemma 13** *If $c_{ij} \geq 1$ for all $i$ and $j$ and all of the computation trees are positive definite, then the sequence $a^0_{i\to j}, a^1_{i\to j}, \ldots$ converges.*

**Proof** Suppose $c_{ij} \geq 1$. By Lemma 12, the $a^t_{i\to j}$ are monotonically decreasing. Because all of the computation trees are positive definite, we must have that for each $i$, $\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + c_{ji} a^{t-1}_{j\to i} > 0$. Therefore, for all $(i,j) \in E$, $a^t_{i\to j} \geq -\frac{\Gamma_{ii}}{c_{ij}}$. Consequently, the sequence $a^0_{i\to j}, a^1_{i\to j}, \ldots$ is monotonically decreasing and bounded from below. This implies that the sequence converges. ■

Because the estimates of the variances only depend on the vector $a^t$, if the $a^t_{i\to j}$ converge, then the estimates of the variances also converge. Therefore, requiring all of the computation trees to be positive definite is a sufficient condition for convergence of the variances. Note, however, that the estimates of the means which correspond to the sequence $b^t_{i\to j}$ need not converge even if all of the computation trees are positive definite (see Figure 6).

Our strategy will be to ensure that all of the computation trees are positive definite by leveraging the choice of parameters, $c_{ij}$. Specifically, we want to use these parameters to weight the diagonal elements of the computation tree much more than the off-diagonal elements in order to force the computation trees to be positive definite. If we can show that there is a choice of each $c_{ij} = c_{ji}$ that will cause all of the computation trees to be positive definite, then Algorithm 1 should behave almost as if the original matrix were scaled diagonally dominant. Indeed, there always exists a choice of the vector $c$ that achieves this.

**Theorem 14** *For any symmetric matrix $\Gamma$ with strictly positive diagonal, $\exists r \geq 1$ and an $\varepsilon > 0$ such that the eigenvalues of the computation trees are bounded from below by $\varepsilon$ when generated by Algorithm 1 with $c_{ij} = r$ for all $i$ and $j$.*

The proof of this theorem exploits the Geršgorin disc theorem in order to show that there exists a choice of $r$ such that each computation tree is scaled diagonally dominant. The complete proof can be found in Appendix B.

### 4.1.2 NEGATIVE PARAMETERS

For the case in which $c_{ij} < 0$ for all $i$ and $j$, we also have that the computation trees are always positive definite when the initial messages are uniformly equal to zero as characterized by the following lemmas.

**Lemma 15** *If $c_{ij} < 0$ for all $i$ and $j$, then for all $t > 0$, $a^t_{i\to j} \leq 0$.*

**Proof** This result follows by induction on $t$. First, suppose that $c_{ij} < 0$ for all $(i,j) \in E$. If the update is not valid, then $a^t_{i\to j} = -\infty$ which trivially satisfies the inequality. Otherwise, we have

$$\begin{aligned} a^t_{i\to j} &= \frac{-\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + (c_{ji}-1)a^{t-1}_{j\to i}} \\ &\leq 0, \end{aligned}$$

where the inequality follows from the induction hypothesis. ■

**Lemma 16** *For any symmetric matrix $\Gamma$ with strictly positive diagonal, if $c_{ij} < 0$ for all $i$ and $j$, then all of the computation trees are positive definite.*

**Proof** The computation trees are all positive definite if and only if $\Gamma_{ii} + \sum_{k\in\partial i} c_{ki} a^t_{k\to i} > 0$ for all $t$. By Lemma 15, $a^t_{i\to j} \leq 0$ for all $t$, and as result, $\Gamma_{ii} + \sum_{k\in\partial i} c_{ki} a^t_{k\to i} \geq \Gamma_{ii} > 0$ for all $t$. ■

As when $c_{ij} \geq 1$ for all $(i,j) \in E$, the eigenvalues on each computation tree are again bounded way from zero, but the $a^t_{i\to j}$ no longer form a monotonic decreasing sequence when $c_{ij} < 0$ for all $(i,j) \in E$. If all of the computation trees remain positive definite in the limit, then the beliefs will all be positive definite upon convergence. If the estimates for the means converge as well, then the converged beliefs must be locally decodable to the correct minimizing assignment. Notice that none of the above arguments for the variances require $\Gamma$ to be positive definite. Indeed, we have already seen an example of a matrix with a strictly positive diagonal and negative eigenvalues (see the matrix in Figure 5) such that the variance estimates converge.

### 4.2 Alternative Message Passing Schedules

The synchronous message-passing updates described in Algorithm 1 enforce a particular ordering on the updates performed at each time step. In practice, alternative message-passing schedules may improve the rate of convergence. One such alternative message-passing schedule is given by Algorithm 2. Because each computation tree produced by this algorithm is a principal submatrix of a synchronous computation tree and principal submatrices of positive definite matrices are positive definite, we can easily check that all of the results of the previous section extend to this modified schedule as well.

Algorithm 2 allows for quite a bit more flexibility in the scheduling of message updates, and as we will see experimentally in Section 4.4, it can have better convergence properties than the corresponding synchronous algorithm. To see why this might be the case, we will again exploit the properties of graph covers. Specifically, we will show that these two algorithms are related via a special 2-cover of the base factor graph.

Every pairwise factor graph, $G = (V_G, E_G)$, admits a bipartite 2-cover, $H = (V_G \times \{1,2\}, E_H)$, called the Kronecker double cover of $G$. We will denote copies of the variable $x_i$ in this 2-cover as $x_{i_1}$ and $x_{i_2}$. For every edge $(i,j) \in E_G$, $(i_1, j_2)$ and $(i_2, j_1)$ belong to $E_H$. In this way, nodes labeled with a one are only connected to nodes labeled with a two (see Figure 7). Note that if $G$ is already a bipartite graph, then the Kronecker double cover of $G$ is simply two disjoint copies of $G$.

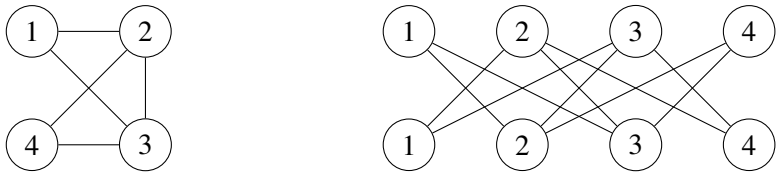
We can view the synchronous algorithm described in Algorithm 1 as a specific message-update schedule of messages on the Kronecker double cover where we perform the update in Algorithm 2 for every variable in the same partition on alternating iterations (see Algorithm 3).

**Algorithm 2** Alternative Reweighted Message-Passing Algorithm

1: Initialize the messages to some finite vector.
2: Choose some ordering of the variables such that each variable is updated infinitely often, and perform the following update for each variable $j$ in order
3: **for** each $i \in \partial j$ **do**
4: Update the message from $i$ to $j$:

$$m_{i\to j}(x_j) := \kappa + \min_{x_i}\left[\frac{\psi_{ij}(x_i,x_j)}{c_{ij}} + (c_{ij}-1)m_{j\to i}(x_i) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} c_{ki}m_{k\to i}(x_i)\right].$$

5: **end for**




(a) A pairwise factor graph $G$ (b) Bipartite 2-cover of $G$

Figure 7: The Kronecker double cover (b) of a pairwise factor graph (a). The node labeled $i \in G$ corresponds to the variable node $x_i$.

**Algorithm 3** Bipartite Message-Passing Algorithm

1: Initialize the messages to some finite vector.
2: Iterate the following until convergence: update all of the outgoing messages from nodes labeled one to nodes labeled two and then update all of the outgoing messages from nodes labeled two to nodes labeled one using the asynchronous update rule:

$$m_{i\to j}(x_j) := \kappa + \min_{x_i}\left[\frac{\psi_{ij}(x_i,x_j)}{c_{ij}} + (c_{ij}-1)m_{j\to i}(x_i) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} c_{ki}m_{k\to i}(x_i)\right].$$

By construction, the message vector produced by Algorithm 3 is simply a concatenation of two consecutive time steps of the synchronous algorithm. Specifically, for all $t \geq 1$

$$m_H^t = \begin{bmatrix} m_G^{2t-1} \\ m_G^{2t-2} \end{bmatrix}.$$

Therefore, the messages passed by Algorithm 1 are identical to those passed by a specific ordering of the updates in Algorithm 2 on the Kronecker double cover. From our earlier analysis, we know that even if $\Gamma$ is positive definite, not every cover necessarily corresponds to a convex objective function. If the Kronecker double cover is such a "bad" cover, then we might expect that the synchronous reweighted algorithm may not converge to the correct solution. This reasoning is not unique to the iterative message-passing algorithms described above. In the next section, we will see that it can also be applied to other iterative techniques for quadratic minimization.

**Algorithm 4** Jacobi Iteration

1: Choose an initial vector $x^0 \in \mathbb{R}^n$.
2: For iteration $t = 1, 2, \ldots$ set

$$x_j^t = \frac{h_j - \sum_k \Gamma_{jk} x_k^{t-1}}{\Gamma_{jj}}$$

for each $j \in \{1, \ldots, n\}$.

**Algorithm 5** Gauss-Seidel Iteration

1: Choose an initial vector $x \in \mathbb{R}^n$.
2: Choose some ordering of the variables, and perform the following update for each variable $j$, in order,

$$x_j = \frac{h_j - \sum_k \Gamma_{jk} x_k}{\Gamma_{jj}}.$$

#### 4.2.1 THE GAUSS-SEIDEL AND JACOBI METHODS

Because minimizing symmetric positive definite quadratic functions is equivalent to solving symmetric positive definite linear systems, well-studied algorithms such as Gaussian elimination, Cholesky decomposition, etc. can be used to compute the minimum. In addition, many iterative algorithms have been proposed to solve the linear system $\Gamma x = h$: Gauss-Seidel iteration, Jacobi iteration, the algebraic reconstruction technique, etc.

In this section, we will show that the previous graph cover analysis can also be used to reason about the Jacobi and Gauss-Seidel algorithms (Algorithms 4 and 5). In Section 4.3, we will see that there is an even deeper connection between these algorithms and reweighted message-passing algorithms. When $\Gamma$ is symmetric positive definite, the objective function, $\frac{1}{2}x^T\Gamma x - h^T x$, is a convex function of $x$. Consequently, we could use a coordinate descent scheme in an attempt to minimize the objective function. The standard cyclic coordinate descent algorithm for this problem is known as the Gauss-Seidel algorithm.

In the same way that Algorithm 1 is a synchronous version of Algorithm 2, the Jacobi algorithm is a synchronous version of the Gauss-Seidel algorithm. To see this, observe that the iterates produced by the Jacobi algorithm are related to the iterates of the Gauss-Seidel algorithm on a larger problem. Specifically, given a symmetric $\Gamma \in \mathbb{R}^{n\times n}$ and $h \in \mathbb{R}^n$, construct $\Gamma' \in \mathbb{R}^{2n\times 2n}$ and $h' \in \mathbb{R}^{2n}$ as follows

$$\begin{aligned} h_i' &= \begin{bmatrix} h \\ h \end{bmatrix}, \\ \Gamma' &= \begin{bmatrix} D & M \\ M & D \end{bmatrix}, \end{aligned}$$

where $D$ is a diagonal matrix with the same diagonal entries as $\Gamma$ and $M = \Gamma - D$.

$\Gamma'$ is the analog of the Kronecker double cover discussed in Section 4.2. Let $x^0 \in \mathbb{R}^n$ be an initial vector for the Jacobi algorithm performed on the matrix $\Gamma$ and fix $y^0 \in \mathbb{R}^{2n}$ such that $y^0 = \begin{bmatrix} x^0 \\ x^0 \end{bmatrix}$. Further, suppose that we update the variables in the order $1, 2, \ldots, 2n$ in the Gauss-Seidel algorithm.

If $y^t$ is the vector produced after $t$ complete cycles of the Gauss-Seidel algorithm, then $y^t = \begin{bmatrix} x^{2t-1} \\ x^{2t} \end{bmatrix}$. Also, observe that, for any $y^t$ such that $\Gamma' y^t = h'$, we must have that $\Gamma \left[ \frac{x^{2t-1}+x^{2t}}{2} \right] = h$.

With these two observations, any convergence result for the Gauss-Seidel algorithm can be extended to the Jacobi algorithm.

**Theorem 17** *Let $\Gamma$ be a symmetric positive semidefinite matrix with a strictly positive diagonal. The Gauss-Seidel algorithm converges to a vector $x^*$ such that $\Gamma x^* = h$ whenever such a vector exists.*

**Proof** See Section 10.5.1 of Byrne (2008). ■

Using our observations, we can immediately produce the following new result.

**Corollary 18** *Let $\Gamma$ be a symmetric positive semidefinite matrix with positive diagonal and let $\Gamma'$ be constructed as above. If $\Gamma'$ is a symmetric positive semidefinite matrix and there exists an $x^*$ such that $\Gamma x^* = h$, then the sequence $\frac{x^t + x^{t-1}}{2}$ converges to $x^*$ where $x^t$ is the $t^{th}$ iterate of the Jacobi algorithm.*

If $\Gamma'$ is not positive semidefinite, then the Gauss-Seidel algorithm (and by extension the Jacobi algorithm) may or may not converge when run on $\Gamma'$.

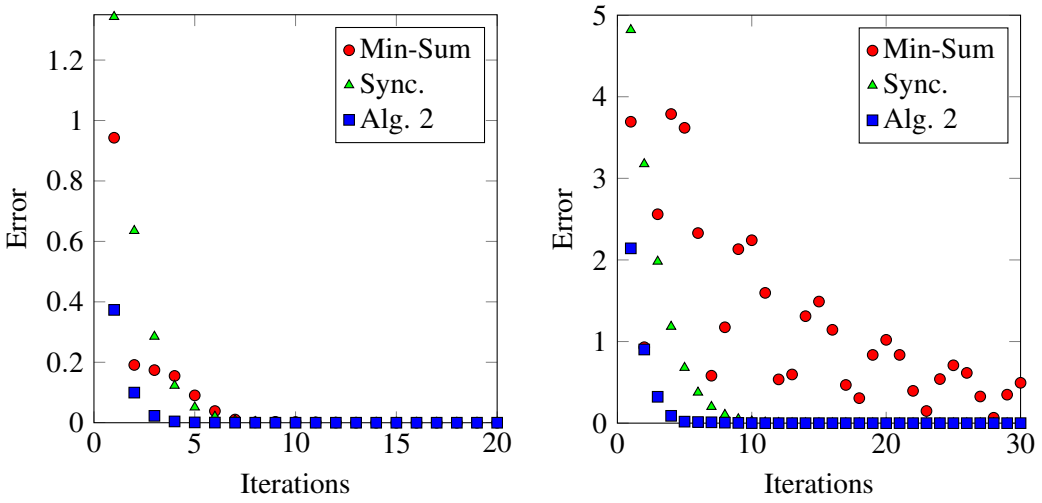
### 4.3 Convergence of the Means

If the variances converge, then the fixed points of the message updates for the means correspond to the solution of a particular linear system $Mb = d$. In fact, we can show that Algorithm 2 is exactly the Gauss-Seidel algorithm for this linear system. First, we construct the matrix $M \in \mathbb{R}^{2|E| \times 2|E|}$:

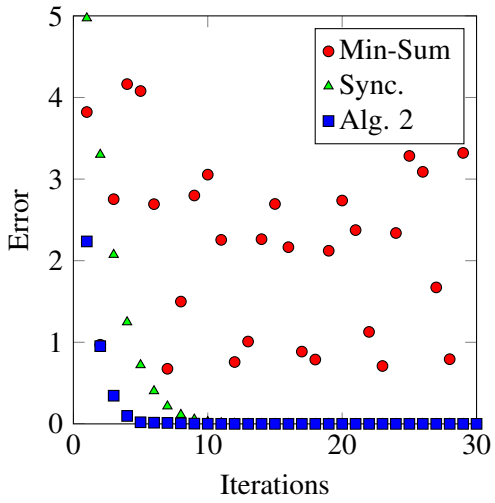
$$\begin{aligned} M_{ij,ij} &= A^*_{i \setminus j} \text{ for all } i \in V \text{ and } j \in \partial i, \\ M_{ij,ki} &= c_{ki} \frac{\Gamma_{ij}}{c_{ij}} \text{ for all } i \in V \text{ and for all } j,k \in \partial i \text{ such that } k \neq j, \\ M_{ij,ji} &= (c_{ij} - 1) \frac{\Gamma_{ij}}{c_{ij}} \text{ for all } i \in V \text{ and } j \in \partial i. \end{aligned}$$

Here, $A^*$ is constructed from the vector of converged variances, $a^*$. All other entries of the matrix are equal to zero. Next, we define the vector $d \in \mathbb{R}^{2|E|}$ by setting $d_{ij} = h_i \Gamma_{ij} / c_{ij}$ for all $i \in V$ and $j \in \partial i$.

By definition, any fixed point, $b^*$, of the message update equations for the means must satisfy $Mb^* = d$. With these definitions, Algorithm 2 is precisely the Gauss-Seidel algorithm for this matrix. Similarly, Algorithm 1 corresponds to the Jacobi algorithm. Unfortunately, $M$ is neither symmetric nor diagonally dominant, so the standard results for the convergence of the Gauss-Seidel algorithm do not necessarily apply to this situation. In practice, we have observed that Algorithm 2 converges for positive definite matrices if each $c_{ij}$ is sufficiently large (or sufficiently negative).




(a) $p = .3$ (b) $p = .398$




(c) $p = .4$

Figure 8: The error, measured by the 2-norm, between the current mean estimate and the true mean at each step of the min-sum algorithm, the alternative message-passing schedule in Algorithm 2 with $c_{ij} = 2$ for all $i \neq j$, and the synchronous algorithm with $c_{ij} = 2$ for all $i \neq j$ for the matrix in (2). Notice that all of the algorithms have a similar performance when $p$ is chosen such that the matrix is scaled diagonally dominant. When the matrix is not scaled diagonally dominant, the min-sum algorithm converges more slowly or does not converge at all.

### 4.4 Experimental Results

Even simple experiments demonstrate the advantages of the reweighted message-passing algorithm compared to the typical min-sum algorithm. Throughout this section, we will assume that $h$ is

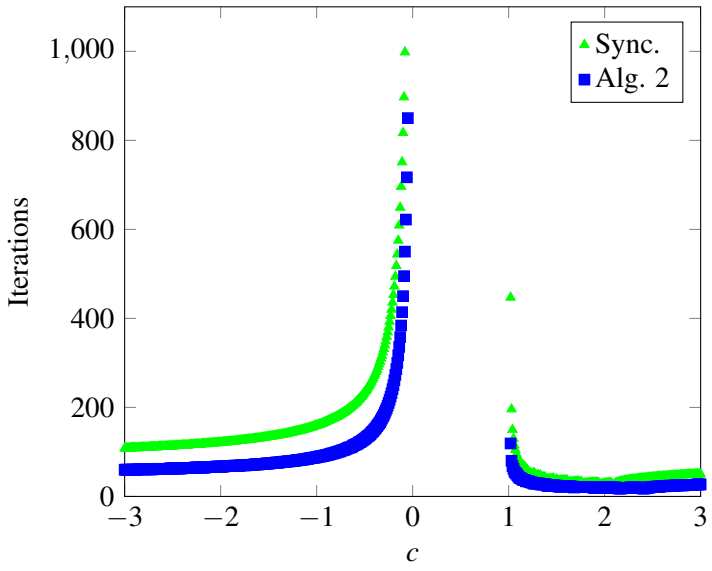



Figure 9: The number of iterations needed to reduce the error of the mean estimates below $10^{-6}$ using the reweighted algorithms as a function of $c$ for the matrix in (2) with $p = .4$. The gap in the plot is predicted by the arguments at the end of Section 3.1.

chosen to be the vector of all ones. Let $\Gamma$ be the following matrix.

$$\begin{pmatrix} 1 & p & -p & -p \\ p & 1 & -p & 0 \\ -p & -p & 1 & -p \\ -p & 0 & -p & 1 \end{pmatrix}. \tag{2}$$

The standard min-sum algorithm converges to the correct solution for $0 \leq p < .39865$ (Malioutov et al., 2006). Figure 8 illustrates the behavior of the min-sum algorithm, Algorithm 2 with $c_{ij} = 2$ for all $i \neq j$, and the synchronous algorithm with $c_{ij} = 2$ for all $i \neq j$ for different choices of the constant $p$. Each iteration of Algorithm 2 algorithm consists of cyclically updating the incoming messages to all nodes. In the examples in Figure 8, the synchronous algorithm and Algorithm 2 always converge rapidly to the correct mean while the min-sum algorithm converges slowly or not at all as $p$ approaches .5.

While this is a simple graph, the behavior of the algorithm for different choices of the vector $c$ is already apparent. If we set $c_{ij} = 3$ for all $i \neq j$, then empirically, both the synchronous algorithm and Algorithm 2 converge for all $p \in (-.5, .5)$, the entire positive definite region for this matrix. However, different choices of the parameter vector can greatly increase or decrease the number of iterations required for convergence. Figure 9 illustrates the iterations to convergence for the reweighted algorithms at $p = .4$ versus $c$.

Although both the synchronous algorithm and Algorithm 2 converge for the entire positive definite region in the above example, they can have very different convergence properties and damping may be required in order to force the synchronous algorithm to converge over arbitrary graphs, even

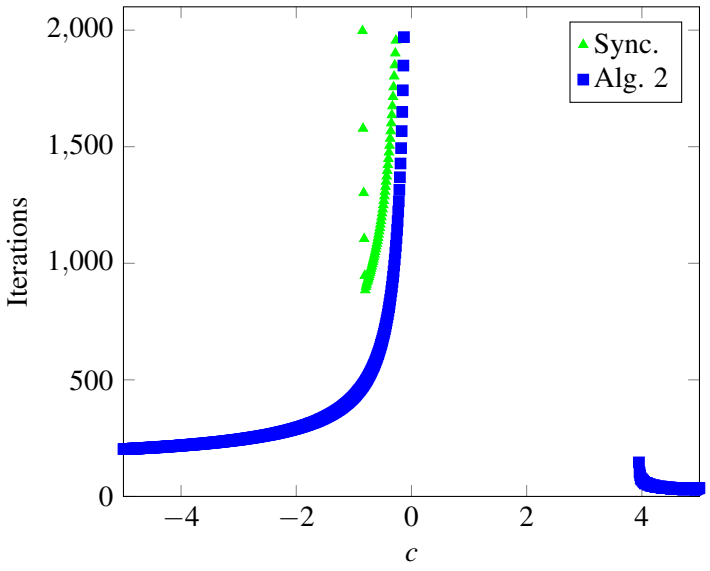



Figure 10: The number of iterations needed to reduce the error of the mean estimates below $10^{-6}$ using the reweighted algorithms as a function of $c$ for the matrix in (3). Again, the gap in the plot is predicted by the arguments at the end of Section 3.1.

for sufficiently large $c$. Figure 10 illustrates these convergence issues for the matrix,

$$\begin{pmatrix} 45 & 21 & 23 & -42 \\ 21 & 83 & 8 & -32 \\ 23 & 8 & 14 & -29 \\ -42 & -32 & -29 & 134 \end{pmatrix}. \tag{3}$$

The above matrix was randomly generated. Similar observations can be made for many other positive definite matrices as well.

## 5. Conclusions and Future Research

In this work, we explored the properties of reweighted message-passing algorithms for the quadratic minimization problem. Our motivation was to address the convergence issues in the GaBP algorithm by leveraging the reweighting parameters. To this end, we employed graph covers to prove that standard approaches to convergence and correctness that exploit duality and coordinate ascent/descent such as MPLP (Globerson and Jaakkola, 2007), tree-reweighted max-product (Wainwright et al., 2003b), and Sontag and Jaakkola (2009) are doomed to fail outside of walk-summable models. While the GaBP variances may not converge outside of walk-summable matrices, we showed that there always exists a choice of reweighting parameters that guarantees monotone convergence of the variances. Empirically, a similar strategy seems to guarantee convergence of the means for positive definite matrices as well. As a result, our approach demonstrably outperforms the GaBP algorithm on this problem. We conclude this work with a discussion of a few open problems and directions for future research.

### 5.1 Convergence

The main open questions surrounding the performance of the reweighted algorithm relate to questions of convergence. First, for all positive definite $\Gamma$, we conjecture that there exists a sufficiently large (or sufficiently negative) choice of the parameters such that the means always converge under Algorithm 2.

Second, in practice, one typically uses a damped version of the message updates in order to attempt to force convergence. For the min-sum algorithm, the damped updates are given by

$$m_{i\to j}^t(x_j) \quad = \quad \kappa + \delta m_{i\to j}^t(x_j) + (1-\delta)\Big[\min_{x_i} \psi_{ij}(x_i,x_j) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} m_{k\to i}^{t-1}(x_i)\Big]$$

for some $\delta \in [0,1)$. The damped min-sum algorithm with damping factor $\delta = 1/2$ empirically seems to converge if $\Gamma$ is positive definite and all of the computation trees remain positive definite (Malioutov et al., 2006). We make the same observation for the damped version of Algorithm 1.

In practice, the damped synchronous algorithm with $\delta = 1/2$ and Algorithm 2 appear to converge for all sufficiently large choices of the parameter vector as long as $\Gamma$ is positive definite. We conjecture that this is indeed the case: for all positive definite $\Gamma$ there exists a $c$ such that if $c_{ij} = c$ for all $i \neq j$, then the damped algorithm and Algorithm 2 always converge. In this line of exploration, the relationship between the synchronous algorithm and Algorithm 2 described in Section 4.2 may be helpful.

Finally, Moallemi and Van Roy (2010) were able to provide rates of convergence in the case that $\Gamma$ is walk-summable by using a careful analysis of the computation trees. Perhaps similar ideas could be adapted for the computation trees produced by the reweigthed algorithm.

### 5.2 General Convex Minimization

The reweighted algorithm can, in theory, be applied to minimize general convex functions, but in practice, computing and storing the message vector for these more general models may be inefficient. Despite this, many of the previous observations can be extended to the general case of a convex function $f : C \to \mathbb{R}$ such that $C \subseteq \mathbb{R}^n$ is a convex set.

As was the case for quadratic minimization, convexity of the objective function $f^G$ does not necessarily guarantee convexity of the objective function $f^H$ for every finite cover $H$ of $G$. Recall that the existence of graph covers that are not bounded from below can be problematic for the reweighted message-passing algorithm. For quadratic functions, this cannot occur if the matrix is scaled diagonally dominant or, equivalently, if the objective function corresponding to every finite graph cover is positive definite. This equivalence suggests a generalization of scaled diagonal dominance for arbitrary convex functions based on the convexity of their graph covers. Such convex functions would have desirable properties with respect to iterative message-passing schemes.

**Lemma 19** *Let $f$ be a convex function that factorizes over a graph $G$. Suppose that for every finite cover $H$ of $G$, $f^H$ is convex. If $x^G \in \arg\min_x f(x)$, then for every finite cover $H$ of $G$, $x^H$, the lift of $x^G$ to $H$, minimizes $f^H$.*

**Proof** This follows from the observation that all convex functions are subdifferentiable over their domains and that $x^H$ is a minimum of $f^H$ if and only if the zero vector is contained in the subgradient of $f^H$ at $x^H$. ■

Even if the objective function is not convex for some cover, we may still be able to use the same trick as in Theorem 14 in order to force the computation trees to be convex. Let $C \subseteq \mathbb{R}^n$ be a convex set. If $f : C \to \mathbb{R}$ is twice continuously differentiable, then $f$ is convex if and only if its Hessian, the matrix of second partial derivatives, is positive semidefinite on the interior of $C$. For each fixed $x \in C$, Theorem 14 demonstrates that there exists a choice of the vector $c$ such that all of the computation trees are convex at $x$, but it does not guarantee the existence of a $c$ that is independent of $x$.

For twice continuously differentiable functions, Moallemi and Van Roy (2010) provide sufficient conditions for the convergence of the min-sum algorithm that are based on a generalization of scaled diagonal dominance, and extending the above ideas is the subject of future research.

## Appendix A. Proof of Theorem 11

Without loss of generality, we can assume that $\Gamma$ has a unit diagonal. We break the proof into several pieces:

- $(1 \Rightarrow 2)$ Without loss of generality we can assume that $|I-\Gamma|$ is irreducible (if not we can make this argument on each of its connected components). Let $1 > \lambda > 0$ be an eigenvalue of $|I-\Gamma|$ with eigenvector $x > 0$ whose existence is guaranteed by the Perron-Frobenius theorem. For any row $i$, we have:

  $$x_i > \lambda x_i = \sum_{j \neq i} |\Gamma_{ij}| x_j.$$

  Since $\Gamma_{ii} = 1$ this is the definition of scaled diagonal dominance with $w = x$.

- $(2 \Rightarrow 3)$ If $\Gamma$ is scaled diagonally dominant then so is every one of its covers. Scaled diagonal dominance of a symmetric matrix with a positive diagonal implies that the matrix is symmetric positive definite. Therefore, all covers must be symmetric positive definite.

- $(3 \Rightarrow 4)$ Trivial.

- $(4 \Rightarrow 1)$ Let $\widetilde{\Gamma}$ be any 2-cover of $\Gamma$. Without loss of generality, we can assume that $\widetilde{\Gamma}$ has the form (1).

  First, observe that by the Perron-Frobenius theorem there exists an eigenvector $x > 0 \in \mathbb{R}^n$ of $|I-\Gamma|$ with eigenvalue $\rho(|I-\Gamma|)$. Let $y \in \mathbb{R}^{2n}$ be constructed by duplicating the values of $x$ so that $y_{2i} = y_{2i+1} = x_i$ for each $i \in \{0...n\}$. By Lemma 10, $y$ is an eigenvector of $|I-\widetilde{\Gamma}|$ with eigenvalue equal to $\rho(|I-\Gamma|)$. We claim that this implies $\rho(|I-\widetilde{\Gamma}|) = \rho(|I-\Gamma|)$. Assume without loss of generality that $|I-\widetilde{\Gamma}|$ is irreducible; if not, then we can apply the following argument to each connected component of $|I-\widetilde{\Gamma}|$. By the Perron-Frobenius theorem again, $|I-\widetilde{\Gamma}|$ has a unique positive eigenvector (up to scalar multiple), with eigenvalue equal to the spectral radius. Thus, $\rho(|I-\Gamma|) = \rho(|I-\widetilde{\Gamma}|)$ because $y > 0$.

  We will now construct a specific cover $\widetilde{\Gamma}$ such that $\widetilde{\Gamma}$ is positive definite if and only if $\Gamma$ is walk-summable. To do this, we'll choose the $P_{ij}$ as in (1) such that $P_{ij} = I$ if $\Gamma_{ij} < 0$ and $P_{ij} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ otherwise. Now define $z \in \mathbb{R}^{2n}$ by setting $z_i = (-1)^i c y_i$, where the constant $c$ ensures that $\|z\| = 1$.

Consider the following:

$$\begin{aligned} z^T\widetilde{\Gamma}z &= \sum_{i=1}^{n}\sum_{j\neq i}\Gamma_{ij}[z_{2i},z_{2i+1}]P_{ij}\begin{bmatrix} z_{2j} \\ z_{2j+1}\end{bmatrix} + \sum_i \Gamma_{ii}z_i^2 \\ &= 1-2\sum_{i>j}|\Gamma_{ij}|c^2y_iy_j. \end{aligned}$$

Recall that $y$ is the eigenvector of $|I-\widetilde{\Gamma}|$ corresponding to the largest eigenvalue and $\|cy\|=1$. By definition and the above,

$$\begin{aligned} \rho(|I-\Gamma|) &= \rho(|I-\widetilde{\Gamma}|) \\ &= \frac{cy^T|I-\widetilde{\Gamma}|cy}{c^2y^Ty} \\ &= 2\sum_{i>j}|\Gamma_{ij}|c^2y_iy_j. \end{aligned}$$

Combining all of the above we see that $z^T\widetilde{\Gamma}z = 1-\rho(|I-\Gamma|)$. Now, $\widetilde{\Gamma}$ positive definite implies that $z^T\widetilde{\Gamma}z > 0$, so $1-\rho(|I-\Gamma|) > 0$. In other words, $\Gamma$ is walk-summable.

## Appendix B. Proof of Theorem 14

Let $T_v(t)$ be the depth $t$ computation tree rooted at $v$, and let $\Gamma'$ be the matrix corresponding to $T_v(t)$ (i.e., the matrix generated by the potentials in the computation tree). We will show that the eigenvalues of $\Gamma'$ are bounded from below by some $\varepsilon > 0$. For any $i \in T_v(t)$ at depth $d$ define:

$$w_i = \left(\frac{s}{r}\right)^d,$$

where $r$ is as in the statement of the theorem and $s$ is a positive real to be determined below. Let $W$ be a diagonal matrix whose entries are given by the vector $w$. By the Geršgorin disc theorem (Horn and Johnson, 1990), all of the eigenvalues of $W^{-1}\Gamma' W$ are contained in

$$\cup_{i\in T_v(t)}\left\{z\in\mathbb{R} : |z-\Gamma'_{ii}| \leq \frac{1}{w_i}\sum_{j\neq i} w_j|\Gamma'_{ij}|\right\}.$$

Because all of the eigenvalues are contained in these discs, we need to show that there is a choice of $s$ and $r$ such that for all $i \in T_v(t)$, $|\Gamma'_{ii}| - \frac{1}{w_i}\sum_{j\neq i} w_j|\Gamma'_{ij}| \geq \varepsilon$.

Recall from Section 2.2.1 that $|\Gamma'_{ij}| = \eta\frac{|\Gamma_{ij}|}{r}$ for some constant $\eta$ that depends on $r$. Further, all potentials below the potential on the edge $(i,j)$ are multiplied by $\eta\gamma$ for some constant $\gamma$. We can divide out by this common constant to obtain equations that depend on $r$ and the elements of $\Gamma$. Note that some self-potentials will be multiplied by $r-1$ while others will be multiplied by $r$. With this rewriting, there are three possibilities:

1. $i$ is a leaf of $T_v(t)$. In this case, we need $|\Gamma_{ii}| > \frac{1}{w_i}\frac{|\Gamma_{ip(i)}|}{r}w_{p(i)}$. Plugging in the definition of $w_i$, we have

$$|\Gamma_{ii}| > \frac{|\Gamma_{ip(i)}|}{s}. \tag{4}$$

2. $i$ is not a leaf of $T_v(t)$ or the root. In this case, we need

$$|\Gamma_{ii}| \quad > \quad \frac{1}{w_i}\Big[\frac{|\Gamma_{ip(i)}|}{r}w_{p(i)} + \frac{s^2(r-1)}{r^3}|\Gamma_{ip(i)}|w_{p(i)} + \sum_{k\in\partial i - p(i)}|\Gamma_{ki}|w_k\Big].$$

Again, plugging the definition of $w_i$ into the above yields

$$|\Gamma'_{ii}| \quad > \quad \frac{|\Gamma_{ip(i)}|}{s} + \frac{s}{r}\Big[\frac{r-1}{r}|\Gamma_{ip(i)}| + \sum_{k\in\partial i - p(i)}|\Gamma_{ki}|\Big].$$

3. $i$ is the root of $T_v(t)$. Similar to the previous case, we need $|\Gamma_{ii}|w_i > \sum_{k\in\partial i}|\Gamma_{ki}|w_k$. Again, plugging the definition of $w_i$ into the above yields

$$|\Gamma_{ii}| > \frac{s}{r}\sum_{k\in\partial i}|\Gamma_{ki}|.$$

None of these bounds are time dependent. As such, if we choose $s$ and $r$ to satisfy the above constraints, then there must exist some $\varepsilon > 0$ such that smallest eigenvalue of any computation tree is at least $\varepsilon$. Fix $s$ to satisfy (4) for all leaves of $T_v(t)$. This implies that $(|\Gamma_{ii}| - \frac{|\Gamma_{ip(i)}|}{s}) > 0$ for any $i \in T_v(t)$. Finally, we can choose a sufficiently large $r$ that satisfies the remaining two cases for all $i \in T_v(t)$.

## References

C. L. Byrne. *Applied Iterative Methods*. A K Peters, Ltd., 2008.

B. J. Frey, R. Koetter, and A. Vardy. Signal-space characterization of iterative decoding. *Information Theory, IEEE Transactions on*, 47(2):766–781, Feb. 2001.

A. Globerson and T. S. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Proc. Neural Information Processing Systems (NIPS)*, Vancouver, B. C., Canada, Dec. 2007.

T. Hazan and A. Shashua. Norm-product belief propagation: Primal-dual message-passing for approximate inference. *Information Theory, IEEE Transactions on*, 56(12):6294 –6316, Dec. 2010.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

J. K. Johnson, D. Bickson, and D. Dolev. Fixing convergence of Gaussian belief propagation. In *Proc. Information Theory, IEEE International Symposium on (ISIT)*, pages 1674–1678, Seoul, South Korea, July 2009.

D. M. Malioutov. Approximate inference in Gaussian graphical models. Ph.D. thesis, EECS, MIT, 2008.

D. M. Malioutov, J. K. Johnson, and A. S. Willsky. Walk-sums and belief propagation in Gaussian graphical models. *Journal of Machine Learning Research (JMLR)*, 7:2031–2064, 2006.

T. Meltzer, A. Globerson, and Y. Weiss. Convergent message passing algorithms: a unifying view. In *Proc. of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, Montreal, Canada, June 2009.

C. C. Moallemi and B. Van Roy. Convergence of min-sum message passing for quadratic optimization. *Information Theory, IEEE Transactions on*, 55(5):2413 –2423, May 2009.

C. C. Moallemi and B. Van Roy. Convergence of min-sum message-passing for convex optimization. *Information Theory, IEEE Transactions on*, 56(4):2041 –2050, April 2010.

N. Ruozzi, J. Thaler, and S. Tatikonda. Graph covers and quadratic minimization. In *Proc. Communication, Control, and Computing, 47th Annual Allerton Conference on*, Allerton, IL, Sept. 2009.

D. Sontag and T. S. Jaakkola. Tree block coordinate descent for MAP in graphical models. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Clearwater Beach, Florida, April 2009.

S. Tatikonda and M. I. Jordan. Loopy belief propagation and Gibbs measures. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 493–500, Edmonton, Alberta, Canada, 2002.

P. O. Vontobel and R. Koetter. Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes. *CoRR*, abs/cs/0512078, 2005.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *Information Theory, IEEE Transactions on*, 49 (5):1120 – 1146, May 2003a.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation via pseudo-moment matching. In *Proceedings of the 9th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Key West, Florida, Jan. 2003b.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing*, 14(2): 143–166, 2004.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on (hyper)trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, 51(11): 3697–3717, Nov. 2005.

Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Comput.*, 12(1):1–41, 2000.

Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Information Theory, IEEE Transactions on*, 47(2):736 –744, Feb. 2001a.

Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Comput.*, 13(10):2173–2200, Oct. 2001b.

T. Werner. A linear programming approach to max-sum problem: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(7):1165–1179, 2007.

Journal of Machine Learning Research 14 (2013) 2315-2347 Submitted 8/11; Revised 5/13; Published 8/13

# The Rate of Convergence of AdaBoost

**Indraneel Mukherjee** IMUKHERJ@CS.PRINCETON.EDU
*Princeton University*
*Department of Computer Science*
*Princeton, NJ 08540 USA*

**Cynthia Rudin** RUDIN@MIT.EDU
*Massachusetts Institute of Technology*
*MIT Sloan School of Management*
*Cambridge, MA 02139 USA*

**Robert E. Schapire** SCHAPIRE@CS.PRINCETON.EDU
*Princeton University*
*Department of Computer Science*
*Princeton, NJ 08540 USA*

**Editor:** Tong Zhang

## Abstract

The AdaBoost algorithm was designed to combine many "weak" hypotheses that perform slightly better than random guessing into a "strong" hypothesis that has very low error. We study the rate at which AdaBoost iteratively converges to the minimum of the "exponential loss." Unlike previous work, our proofs do not require a weak-learning assumption, nor do they require that minimizers of the exponential loss are finite. Our first result shows that the exponential loss of AdaBoost's computed parameter vector will be at most $\varepsilon$ more than that of any parameter vector of $\ell_1$-norm bounded by $B$ in a number of rounds that is at most a polynomial in $B$ and $1/\varepsilon$. We also provide lower bounds showing that a polynomial dependence is necessary. Our second result is that within $C/\varepsilon$ iterations, AdaBoost achieves a value of the exponential loss that is at most $\varepsilon$ more than the best possible value, where $C$ depends on the data set. We show that this dependence of the rate on $\varepsilon$ is optimal up to constant factors, that is, at least $\Omega(1/\varepsilon)$ rounds are necessary to achieve within $\varepsilon$ of the optimal exponential loss.

**Keywords:** AdaBoost, optimization, coordinate descent, convergence rate

## 1. Introduction

The AdaBoost algorithm of Freund and Schapire (1997) was designed to combine many "weak" hypotheses that perform slightly better than random guessing into a "strong" hypothesis that has very low error. Despite extensive theoretical and empirical study, basic properties of AdaBoost's convergence are not fully understood. In this work, we focus on one of those properties, namely, to find convergence rates that hold in the absence of any simplifying assumptions. Such assumptions, relied upon in much of the preceding work, make it easier to prove a fast convergence rate for AdaBoost, but often do not hold in the cases where AdaBoost is commonly applied.

AdaBoost can be viewed as a coordinate descent (or functional gradient descent) algorithm that iteratively minimizes an objective function $L : \mathbb{R}^n \to \mathbb{R}$ called the *exponential loss* (Breiman, 1999; Frean and Downs, 1998; Friedman et al., 2000; Friedman, 2001; Mason et al., 2000; Onoda

©2013 Indraneel Mukherjee, Cynthia Rudin and Robert E. Schapire.

et al., 1998; Rätsch et al., 2001; Schapire and Singer, 1999). Given $m$ labeled training examples $(x_1, y_1), \ldots, (x_m, y_m)$, where the $x_i$'s are in some domain $\mathcal{X}$ and $y_i \in \{-1, +1\}$, and a finite (but typically very large) space of weak hypotheses $\mathcal{H} = \{\hbar_1, \ldots, \hbar_N\}$, where each $\hbar_j : \mathcal{X} \to \{-1, +1\}$, the exponential loss is defined as

$$L(\boldsymbol{\lambda}) \triangleq \frac{1}{m} \sum_{i=1}^{m} \exp\left( - \sum_{j=1}^{N} \lambda_j y_i \hbar_j(x_i) \right)$$

where $\boldsymbol{\lambda} = \langle \lambda_1, \ldots, \lambda_N \rangle$ is a vector of weights or parameters. This definition can also be extended to the setting where the weak hypotheses are confidence rated, that is, they output real numbers, with the sign denoting prediction and magnitude indicating the confidence in that prediction. In each iteration, a coordinate descent algorithm moves some distance along some coordinate direction $\lambda_j$. For AdaBoost, the coordinate directions correspond to the individual weak hypotheses. Thus, on each round, AdaBoost chooses some weak hypothesis and step length, and adds these to the current weighted combination of weak hypotheses, which is equivalent to updating a single weight. The direction and step length are so chosen that the resulting vector $\boldsymbol{\lambda}^t$ in iteration $t$ yields a lower value of the exponential loss than in the previous iteration, $L(\boldsymbol{\lambda}^t) < L(\boldsymbol{\lambda}^{t-1})$. This repeats until it reaches a minimizer if one exists. It was shown by Collins et al. (2002), and later by Zhang and Yu (2005), that AdaBoost asymptotically converges to the minimum possible exponential loss. That is,

$$\lim_{t \to \infty} L(\boldsymbol{\lambda}^t) = \inf_{\boldsymbol{\lambda} \in \mathbb{R}^N} L(\boldsymbol{\lambda}).$$

However, the work by Collins et al. (2002) did not address a rate of convergence to the minimum exponential loss.

Our work specifically addresses a recent conjecture of Schapire (2010) stating that there exists a positive constant $c$ and a polynomial $\mathrm{poly}()$ such that for all training sets and all finite sets of weak hypotheses, and for all $B > 0$,

$$L(\boldsymbol{\lambda}^t) \leq \min_{\boldsymbol{\lambda} : \|\boldsymbol{\lambda}\|_1 \leq B} L(\boldsymbol{\lambda}) + \frac{\mathrm{poly}(\log N, m, B)}{t^c}. \tag{1}$$

In other words, the exponential loss of AdaBoost will be at most $\varepsilon$ more than that of any other parameter vector $\boldsymbol{\lambda}$ of $\ell_1$-norm bounded by $B$ in a number of rounds that is bounded by a polynomial in $\log N$, $m$, $B$ and $1/\varepsilon$. (We require $\log N$ rather than $N$ since the number of weak hypotheses will typically be extremely large.) Along with an upper bound that is polynomial in these parameters, we also provide lower bound constructions showing some polynomial dependence on $B$ and $1/\varepsilon$ is necessary. Without any additional assumptions on the exponential loss $L$, and without altering AdaBoost's minimization algorithm for $L$, the best known convergence rate of AdaBoost prior to this work that we are aware of is that of Bickel et al. (2006) who prove a bound on the rate of the form $O(1/\sqrt{\log t})$.

We provide also a convergence rate of AdaBoost to the minimum value of the exponential loss. Namely, within $C/\varepsilon$ iterations, AdaBoost achieves a value of the exponential loss that is at most $\varepsilon$ more than the best possible value, where $C$ depends on the data set. This convergence rate is different from the one discussed above in that it has better dependence on $\varepsilon$ (in fact the dependence is optimal, as we show), and does not depend on the best solution within a ball of size $B$. However, this second convergence rate cannot be used to prove (1) since in certain worst case situations, we show the constant $C$ may be larger than $2^m$ (although usually it will be much smaller).

Within the proof of the second convergence rate, we provide a lemma (called the *decomposition lemma*) that shows that the training set can be split into two sets of examples: the "finite margin set," and the "zero loss set." Examples in the finite margin set always make a positive contribution to the exponential loss, and they never lie too far from the decision boundary. Examples in the zero loss set do not have these properties. If we consider the exponential loss where the sum is only over the finite margin set (rather than over all training examples), it is minimized by a finite $\boldsymbol{\lambda}$. The fact that the training set can be decomposed into these two classes is the key step in proving the second convergence rate.

This problem of determining the rate of convergence is relevant in the proof of the consistency of AdaBoost given by Bartlett and Traskin (2007), where it has a direct impact on the rate at which AdaBoost converges to the Bayes optimal classifier (under suitable assumptions).

There have been several works that make additional assumptions on the exponential loss in order to attain a better bound on the rate, but those assumptions are not true in general, and cases are known where each of these assumptions are violated. For instance, better bounds are proved by Rätsch et al. (2002) using results from Luo and Tseng (1992), but these require that the exponential loss be minimized by a finite $\boldsymbol{\lambda}$, and also depend on quantities that are not easily measured. There are many cases where $L$ does not have a finite minimizer; in fact, one such case is provided by Schapire (2010). Shalev-Shwartz and Singer (2008) have proved bounds for a variant of AdaBoost. Zhang and Yu (2005) also have given rates of convergence, but their technique requires a bound on the change in the size of $\boldsymbol{\lambda}^t$ at each iteration that does not necessarily hold for AdaBoost. Many classic results are known on the convergence of iterative algorithms generally (see, for instance, Luenberger and Ye, 2008; Boyd and Vandenberghe, 2004); however, these typically start by assuming that the minimum is attained at some finite point in the (usually compact) space of interest, assumptions that do not generally hold in our setting. When the weak learning assumption holds, there is a parameter $\gamma > 0$ that governs the improvement of the exponential loss at each iteration. Freund and Schapire (1997) and Schapire and Singer (1999) showed that the exponential loss is at most $e^{-2t\gamma^2}$ after $t$ rounds, so AdaBoost rapidly converges to the minimum possible loss under this assumption.

In Section 2 we summarize the coordinate descent view of AdaBoost. Section 3 contains the proof of the conjecture of Schapire (2010), with associated lower bounds proved in Section 3.3. Section 4 provides the $C/\varepsilon$ convergence rate. The proof of the decomposition lemma is given in Section 4.2.

## 2. Coordinate Descent View of AdaBoost

From the examples $(x_1, y_1), \ldots, (x_m, y_m)$ and hypotheses $\mathcal{H} = \{\hbar_1, \ldots, \hbar_N\}$, AdaBoost iteratively computes the function $F : \mathcal{X} \to \mathbb{R}$, where $\operatorname{sign}(F(x))$ can be used as a classifier for a new instance $x$. The function $F$ is a linear combination of the hypotheses. At each iteration $t$, AdaBoost chooses one of the weak hypotheses $h_t$ from the set $\mathcal{H}$, and adjusts its coefficient by a specified value $\alpha_t$. Then $F$ is constructed after $T$ iterations as: $F(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$. Figure 1 shows the AdaBoost algorithm (Freund and Schapire, 1997).

Since each $h_t$ is equal to $\hbar_{j_t}$ for some $j_t$, $F$ can also be written $F(x) = \sum_{j=1}^{N} \lambda_j \hbar_j(x)$ for a vector of values $\boldsymbol{\lambda} = \langle \lambda_1, \ldots \lambda_N \rangle$ (such vectors will sometimes also be referred to as *combinations*, since they represent combinations of weak hypotheses). In different notation, we can write AdaBoost as a coordinate descent algorithm on vector $\boldsymbol{\lambda}$. We define the *feature matrix* $\mathbf{M}$ elementwise by $M_{ij} = y_i \hbar_j(x_i)$, so that this matrix contains all of the inputs to AdaBoost (the training examples and

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathcal{X}$, $y_i \in \{-1, +1\}$
set $\mathcal{H} = \{\hbar_1, \ldots, \hbar_N\}$ of weak hypotheses $\hbar_j : \mathcal{X} \to \{-1, +1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1, \ldots, m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution $D_t$; that is, find weak hypothesis $h_t \in \mathcal{H}$ whose correlation $r_t \triangleq \mathbb{E}_{i \sim D_t}[y_i h_t(x_i)]$ has maximum magnitude $|r_t|$.
- Choose $\alpha_t = \frac{1}{2} \ln\{(1 + r_t) / (1 - r_t)\}$.
- Update, for $i = 1, \ldots, m$: $D_{t+1}(i) = D_t(i) \exp(-\alpha_t y_i h_t(x_i))/Z_t$
  where $Z_t$ is a normalization factor (chosen so that $D_{t+1}$ will be a distribution).

Output the final hypothesis: $F(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$.

Figure 1: The boosting algorithm AdaBoost.

hypotheses). Then the exponential loss can be written more compactly as:

$$L(\boldsymbol{\lambda}) = \frac{1}{m} \sum_{i=1}^{m} e^{-(\mathbf{M}\boldsymbol{\lambda})_i}$$

where $(\mathbf{M}\boldsymbol{\lambda})_i$, the $i^{\text{th}}$ coordinate of the vector $\mathbf{M}\boldsymbol{\lambda}$, is the (unnormalized) *margin* achieved by vector $\boldsymbol{\lambda}$ on training example $i$.

Coordinate descent algorithms choose a coordinate at each iteration where the directional derivative is the steepest, and choose a step that maximally decreases the objective along that coordinate. To perform coordinate descent on the exponential loss, we determine the coordinate $j_t$ at iteration $t$ as follows, where $\mathbf{e}_j$ is a vector that is 1 in the $j^{\text{th}}$ position and 0 elsewhere:

$$j_t \in \operatorname*{argmax}_j \left| \left( -\frac{dL(\boldsymbol{\lambda}^{t-1} + \alpha \mathbf{e}_j)}{d\alpha}\Big|_{\alpha=0} \right) \right| = \operatorname*{argmax}_j \frac{1}{m} \left| \sum_{i=1}^{m} e^{-(\mathbf{M}\boldsymbol{\lambda}^{t-1})_i} M_{ij} \right|. \tag{2}$$

We can show that this is equivalent to the weak learning step of AdaBoost. Unraveling the recursion in Figure 1 for AdaBoost's weight vector $D_t$, we can see that $D_t(i)$ is proportional to

$$\exp\left( -\sum_{t' < t} \alpha_{t'} y_i h_{t'}(x_i) \right).$$

The term in the exponent can also be rewritten in terms of the vector $\boldsymbol{\lambda}^t$, where $\lambda_j^t$ is the sum of $\alpha_t$'s where hypothesis $\hbar_j$ was chosen: $\sum_{t' < t} \alpha_{t'} \mathbf{1}_{[\hbar_j = h_{t'}]} = \lambda_{t-1,j}$. The term in the exponent is:

$$\sum_{t' < t} \alpha_{t'} y_i h_{t'}(x_i) = \sum_j \sum_{t' < t} \alpha_{t'} \mathbf{1}_{[\hbar_j = h_{t'}]} y_i \hbar_j(x_i) = \sum_j \lambda_j^{t-1} M_{ij} = (\mathbf{M}\boldsymbol{\lambda}^{t-1})_i,$$

where $(\cdot)_i$ denotes the $i$th component of a vector. This means $D_t(i)$ is proportional to $e^{-(\mathbf{M}\boldsymbol{\lambda}^{t-1})_i}$. Equation (2) can now be rewritten as

$$j_t \in \operatorname*{argmax}_j \left| \sum_i D_t(i) M_{ij} \right| = \operatorname*{argmax}_j \left| \mathbb{E}_{i \sim D_t}[M_{ij}] \right| = \operatorname*{argmax}_j \left| \mathbb{E}_{i \sim D_t}[y_i h_j(x_i)] \right|,$$

which is exactly the way AdaBoost chooses a weak hypothesis in each round (see Figure 1). The correlation $\sum_i D_t(i)M_{ij_t}$ will be denoted by $r_t$ and its absolute value $|r_t|$ denoted by $\delta_t$. The quantity $\delta_t$ is commonly called the *edge* for round $t$. The distance $\alpha_t$ to travel along direction $j_t$ is found for coordinate descent via a linesearch (see for instance Mason et al., 2000):

$$0 = -\frac{dL(\boldsymbol{\lambda}^t + \alpha_t \mathbf{e}_{j_t})}{d\alpha_t} = \sum_i e^{-\left(M(\boldsymbol{\lambda}^t + \alpha_t \mathbf{e}_{j_t})\right)_i} M_{ij_t}$$

and dividing both sides by the normalization factor,

$$0 = \sum_{i:M_{ij}=1} D_t(i)e^{-\alpha_t} - \sum_{i:M_{ij}=-1} D_t(i)e^{\alpha_t} = \frac{(1+r_t)}{2}e^{-\alpha_t} - \frac{(1-r_t)}{2}e^{\alpha_t},$$

that is

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1+r_t}{1-r_t}\right),$$

just as in Figure 1. Thus, AdaBoost is equivalent to coordinate descent on $L(\boldsymbol{\lambda})$. With this choice of step length, it can be shown (Freund and Schapire, 1997) that the exponential loss drops by an amount depending on the edge. First notice that for any value of $\alpha_t$ we have:

$$\begin{aligned} L(\boldsymbol{\lambda}^t) &= L\left(\boldsymbol{\lambda}^{t-1} + \alpha_t \mathbf{e}_{\mathbf{j_t}}\right) \\ &= \left(\sum_{i:M_{ij_t}=1} D_t(i)e^{-\alpha_t} + \sum_{i:M_{ij_t}=-1} D_t(i)e^{\alpha_t}\right) L(\boldsymbol{\lambda}^{t-1}) \\ &= \left(\frac{(1+r_t)}{2}e^{-\alpha_t} + \frac{(1-r_t)}{2}e^{\alpha_t}\right) L(\boldsymbol{\lambda}^{t-1}). \end{aligned} \tag{3}$$

Plugging in the choice of $\alpha_t$ calculated before, we have

$$L(\boldsymbol{\lambda}^t) = \left(\sqrt{(1+r_t)(1-r_t)}\right) L(\boldsymbol{\lambda}^{t-1}) = \left(\sqrt{1-r_t^2}\right) L(\boldsymbol{\lambda}^{t-1}) = \left(\sqrt{1-\delta_t^2}\right) L(\boldsymbol{\lambda}^{t-1}).$$

Our rate bounds also hold when the weak-hypotheses are confidence-rated, that is, giving real-valued predictions in $[-1,+1]$, so that $h : \mathcal{X} \to [-1,+1]$. In that case, the criterion for picking a weak hypothesis in each round remains the same, that is, at round $t$, an $h_{j_t}$ maximizing the absolute correlation $j_t \in \operatorname{argmax}_j \left|\sum_{i=1}^m e^{-(\mathbf{M}\boldsymbol{\lambda}^{t-1})_i} M_{ij}\right|$, is chosen, where $M_{ij}$ may now be non-integral. An exact analytical line search is no longer possible, but if the step size is chosen in the same way,

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1+r_t}{1-r_t}\right), \tag{4}$$

then Freund and Schapire (1997) and Schapire and Singer (1999) show that a similar drop in the loss is still guaranteed:

$$L(\boldsymbol{\lambda}^t) \leq L(\boldsymbol{\lambda}^{t-1})\sqrt{1-\delta_t^2}. \tag{5}$$

With confidence rated hypotheses, other implementations may choose the step size in a different way. However, in this paper, by "AdaBoost" we will always mean the version in Freund and Schapire (1997) and Schapire and Singer (1999) which chooses step sizes as in (4), and enjoys

the loss guarantee as in (5). That said, all our proofs work more generally, and are robust to numerical inaccuracies in the implementation. In other words, even if the previous conditions are violated by a small amount, similar bounds continue to hold, although we leave out explicit proofs of this fact to simplify the presentation.

Before proceeding to the statements and proofs of convergence we make a few technical observations that will simplify all the proofs considerably. All the convergence statements in this paper are of the following form. Within a specific number of rounds $T$, AdaBoost will achieve loss at most $L_0$ for some non-negative $L_0$: that is, $L(\boldsymbol{\lambda}^T) \leq L_0$. The non-negativity is necessary since the exponential-loss $L$ only takes non-negative values, and hence the minimum attainable value is 0. Since the loss is non-decreasing through various rounds of AdaBoost, we may assume, for the sake of proving the kind of bound mentioned above, that the losses $L(\boldsymbol{\lambda}^1), \ldots, L(\boldsymbol{\lambda}^T)$ are all strictly greater than zero. Otherwise, within $T$ rounds the minimum possible loss of zero has already been attained and there is nothing to prove. By virtue of (5), the positivity assumption on the losses in turn implies that we may assume that the edges $\delta_1, \ldots, \delta_T$ are all strictly less than 1. Finally, note that $\delta_t = 0$ implies that the optimal solution has been attained. To have a nontrivial convergence problem, we may assume that all the edges are positive. Thus, we assume throughout that $0 < \delta_t < 1$ to ensure that the statements and proofs are non-trivial.

## 3. First Convergence Rate: Convergence To Any Target Loss

In this section, we bound the number of rounds of AdaBoost required to get within $\varepsilon$ of the loss attained by a parameter vector $\boldsymbol{\lambda}^*$ as a function of $\varepsilon$ and the $\ell_1$-norm $\|\boldsymbol{\lambda}^*\|_1$. The vector $\boldsymbol{\lambda}^*$ serves as a reference based on which we define the target loss $L(\boldsymbol{\lambda}^*)$, and we will show that its $\ell_1$-norm measures the difficulty of attaining the target loss in a specific sense. We prove a bound polynomial in $1/\varepsilon$, $\|\boldsymbol{\lambda}^*\|_1$ and the number of examples $m$, showing (1) holds, thereby resolving affirmatively the open problem posed in Schapire (2010). Later in the section we provide lower bounds showing how a polynomial dependence on both parameters is necessary.

### 3.1 Upper Bound

The main result of this section is the following rate upper bound.

**Theorem 1** *For any $\boldsymbol{\lambda}^* \in \mathbb{R}^N$, AdaBoost achieves loss at most $L(\boldsymbol{\lambda}^*) + \varepsilon$ in at most $13\|\boldsymbol{\lambda}^*\|_1^6 \varepsilon^{-5}$ rounds.*

The high level idea behind the proof of the theorem is as follows. To show a fast rate, we require a large edge in each round, as indicated by (5). A large edge is guaranteed if the size of the current solution $\|\boldsymbol{\lambda}^t\|_1$ of AdaBoost is small. Therefore AdaBoost makes good progress if the size of its solution does not grow too fast. On the other hand, the increase in size of its solution is given by the step length, which in turn is proportional to the edge achieved in that round. Therefore, if the solution size grows fast, the loss also drops fast. Either way the algorithm makes good progress. In the rest of the section we make these ideas concrete through a sequence of lemmas.

We provide some more notation. Throughout, $\boldsymbol{\lambda}^*$ is fixed, and its $\ell_1$-norm is denoted by $B$ (matching the notation in Schapire, 2010). One key parameter is the *suboptimality* $R_t$ of AdaBoost's solution measured via the logarithm of the exponential loss:

$$R_t \triangleq \ln L(\boldsymbol{\lambda}^t) - \ln L(\boldsymbol{\lambda}^*).$$

Another key parameter is the $\ell_1$-distance $S_t$ of AdaBoost's solution from the closest combination that achieves the target loss:

$$S_t \triangleq \inf_{\boldsymbol{\lambda}} \left\{ \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^t\|_1 : L(\boldsymbol{\lambda}) \leq L(\boldsymbol{\lambda}^*) \right\}.$$

We will also be interested in how they change as captured by

$$\Delta R_t \triangleq R_{t-1} - R_t, \qquad \Delta S_t \triangleq S_t - S_{t-1}.$$

Notice that $\Delta R_t$ is always non-negative since AdaBoost's loss is always decreasing, thus the suboptimality also decreases in each round. We assume without loss of generality that $R_0, \ldots, R_t$ and $S_0, \ldots, S_t$ are all strictly positive up to at least $t = 13\|\boldsymbol{\lambda}^*\|_1^6 \varepsilon^{-5}$, since otherwise the theorem holds trivially. In the rest of the section, we restrict our attention entirely to rounds of boosting when these positivity conditions hold. We first show that a $\mathrm{poly}(B, \varepsilon^{-1})$ rate of convergence follows if the edge is always polynomially large compared to the suboptimality.

**Lemma 2** *If for some constants $c_1, c_2$, where $c_2 \geq 1$, the edge satisfies $\delta_t \geq B^{-c_1} R_{t-1}^{c_2}$ in each round $t$, then AdaBoost achieves at most $L(\boldsymbol{\lambda}^*) + \varepsilon$ loss after $2B^{2c_1}(\varepsilon \ln 2)^{1-2c_2}$ rounds.*

We will need the following expression within the proofs. From the definition of $R_t$ and (5) we have

$$\Delta R_t = \ln L(\boldsymbol{\lambda}^{t-1}) - \ln L(\boldsymbol{\lambda}^t) \geq -\frac{1}{2}\ln(1 - \delta_t^2). \tag{6}$$

**Proof** Combining (6) with the inequality $e^x \geq 1 + x$, and the assumption on the edge,

$$\Delta R_t \geq -\frac{1}{2}\ln(1 - \delta_t^2) \geq \frac{1}{2}\delta_t^2 \geq \frac{1}{2}B^{-2c_1} R_{t-1}^{2c_2}.$$

Let $T = \lceil 2B^{2c_1}(\varepsilon \ln 2)^{1-2c_2} \rceil$ be the bound on the number of rounds in the lemma. If any of $R_0, \ldots, R_T$ are negative, then by monotonicity $R_T < 0$ and we are done. Otherwise, they are all non-negative. Then, applying Lemma 32 from the Appendix to the sequence $R_0, \ldots, R_T$, and using $c_2 \geq 1$ we get

$$R_T^{1-2c_2} \geq R_0^{1-2c_2} + \left(c_2 - \frac{1}{2}\right) B^{-2c_1} T \geq (1/2) B^{-2c_1} T \geq (\varepsilon \ln 2)^{1-2c_2} \implies R_T \leq \varepsilon \ln 2.$$

If either $\varepsilon$ or $L(\boldsymbol{\lambda}^*)$ is greater than 1, then the lemma follows since $L(\boldsymbol{\lambda}^T) \leq L(\boldsymbol{\lambda}^0) = 1 < L(\boldsymbol{\lambda}^*) + \varepsilon$. Otherwise,

$$L(\boldsymbol{\lambda}^T) \leq L(\boldsymbol{\lambda}^*) e^{\varepsilon \ln 2} \leq L(\boldsymbol{\lambda}^*)(1 + \varepsilon) \leq L(\boldsymbol{\lambda}^*) + \varepsilon,$$

where the second inequality uses $e^x \leq 1 + (1/\ln 2)x$ for $x \in [0, \ln 2]$. ■

We next show that large edges are achieved provided $S_t$ is small compared to $R_t$.

**Lemma 3** *In each round $t$, the edge satisfies $\delta_t S_{t-1} \geq R_{t-1}$.*

**Proof** For any combination $\boldsymbol{\lambda}$, define $D_{\boldsymbol{\lambda}}$ as the distribution on examples $\{1, \ldots, m\}$ that puts weight proportional to the loss $D_{\boldsymbol{\lambda}}(i) = e^{-(\mathbf{M}\boldsymbol{\lambda})_i}/(mL(\boldsymbol{\lambda}))$. Choose any $\boldsymbol{\lambda}$ suffering at most the target loss $L(\boldsymbol{\lambda}) \leq L(\boldsymbol{\lambda}^*)$. By non-negativity of relative entropy we get

$$\begin{aligned} 0 \leq \mathrm{RE}(D_{\boldsymbol{\lambda}^{t-1}} \,\|\, D_{\boldsymbol{\lambda}}) &= \sum_{i=1}^m D_{\boldsymbol{\lambda}^{t-1}}(i) \ln \left( \frac{e^{-(\mathbf{M}\boldsymbol{\lambda}^{t-1})_i}/mL(\boldsymbol{\lambda}^{t-1})}{e^{-(\mathbf{M}\boldsymbol{\lambda})_i}/mL(\boldsymbol{\lambda})} \right) \\ &\leq -R_{t-1} + \sum_{i=1}^m D_{\boldsymbol{\lambda}^{t-1}}(i) \left(\mathbf{M}\boldsymbol{\lambda} - \mathbf{M}\boldsymbol{\lambda}^{t-1}\right)_i. \end{aligned} \tag{7}$$

Note that $D_{\boldsymbol{\lambda}^{t-1}}$ is the distribution $D_t$ that AdaBoost creates in round $t$. The above summation can be rewritten as

$$\begin{aligned}\sum_{i=1}^{m} D_{\boldsymbol{\lambda}^{t-1}}(i) \sum_{j=1}^{N} \left(\lambda_j - \lambda_j^{t-1}\right) M_{ij} &= \sum_{j=1}^{N} \left(\lambda_j - \lambda_j^{t-1}\right) \sum_{i=1}^{m} D_t(i) M_{ij} \\ &\leq \left(\sum_{j=1}^{N} \left|\lambda_j - \lambda_j^{t-1}\right|\right) \max_j \left|\sum_{i=1}^{m} D_t(i) M_{ij}\right| \\ &= \delta_t \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^{t-1}\|_1. \end{aligned} \tag{8}$$

Since the previous holds for any $\boldsymbol{\lambda}$ suffering less than the target loss, we can choose in particular a $\boldsymbol{\lambda}$ so that $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^{t-1}\|_1$ is arbitrarily close to $S_{t-1}$ showing that the last expression is at most $\delta_t S_{t-1}$. Combining this with (7) completes the proof. ■

To complete the proof of Theorem 1, we will need that $S_t$ is small compared to $R_t$. In fact we prove:

**Lemma 4** *Whenever $R_0, \ldots, R_t$ and $S_0, \ldots, S_t$ are all strictly positive, $S_t \leq B^3 R_t^{-2}$.*

Before we prove Lemma 4, we show how to prove Theorem 1.

**Proof** [Of Theorem 1] Again if the positivity conditions on $R_0, \ldots, R_t$ and $S_0, \ldots, S_t$ do not hold, then the result is trivial. Thus, assume these quantities are positive at least for $t \leq 13\|\boldsymbol{\lambda}^*\|_1^6 \varepsilon^{-5}$. Combining Lemma 3 and Lemma 4 yields

$$\delta_t \geq B^{-3} R_{t-1}^3.$$

Notice this matches the condition of Lemma 2 for $c_1 = c_2 = 3$. Lemma 2 provides the desired bound on the number of rounds:

$$2(\varepsilon \ln 2)^{1-2\cdot 3} B^{2\cdot 3} < 13 B^6 \varepsilon^{-5}.$$

■

We still need to prove Lemma 4. The bound on $S_t$ in Lemma 4 can be proved if we can first show $S_t$ grows slowly compared to the rate at which the suboptimality $R_t$ falls. Intuitively, when the edge $\delta_t$ is large, it leads to a large step size, causing growth in $S_t$, and a proportionately larger improvement in the suboptimality. To be precise, we prove the following about the relationship between $S_t$ and $R_t$.

**Lemma 5** *Whenever $R_0, \ldots, R_t$ and $S_0, \ldots, S_t$ are all strictly positive, we have*

$$\frac{2\Delta R_t}{R_{t-1}} \geq \frac{\Delta S_t}{S_{t-1}}.$$

**Proof** Firstly, it follows from the definition of $S_t$ that $\Delta S_t \leq \|\boldsymbol{\lambda}^t - \boldsymbol{\lambda}^{t-1}\|_1 = |\alpha_t|$. Next, using (6) and (4) we may write $\Delta R_t \geq \Upsilon(\delta_t)\,|\alpha_t|$, where the function $\Upsilon$ has been defined in Rätsch and Warmuth (2005) as

$$\Upsilon(x) = \frac{-\ln(1-x^2)}{\ln\left(\frac{1+x}{1-x}\right)}.$$

It is known (Rätsch and Warmuth, 2005; Rudin et al., 2007) that $\Upsilon(x) \geq x/2$ for $x \in [0,1]$. Combining and using Lemma 3,

$$\Delta R_t \geq |\alpha_t|\,\delta_t/2 \geq \delta_t \Delta S_t/2 \geq R_{t-1}\Delta S_t/2S_{t-1}.$$

Rearranging completes the proof. ■

Using this we prove Lemma 4.

**Proof** [Of Lemma 4] We first show $S_0 \leq B^3 R_0^{-2}$. Note, $S_0 \leq \|\boldsymbol{\lambda}^* - \boldsymbol{\lambda}^0\|_1 = B$, so that it suffices to show $R_0 \leq B$. By definition the quantity $R_0 = -\ln\left(\frac{1}{m}\sum_i e^{-(\mathbf{M}\boldsymbol{\lambda}^*)_i}\right)$. The quantity $(\mathbf{M}\boldsymbol{\lambda}^*)_i$ is the inner product of row $i$ of matrix $\mathbf{M}$ with the vector $\boldsymbol{\lambda}^*$. Since the entries of $\mathbf{M}$ lie in $[-1,+1]$, this is at most $\|\boldsymbol{\lambda}^*\|_1 = B$. Therefore $R_0 \leq -\ln\left(\frac{1}{m}\sum_i e^{-B}\right) = B$, which is what we needed.

To complete the proof, we show that $R_t^2 S_t$ is non-increasing. It suffices to show for any $t$ the inequality $R_t^2 S_t \leq R_{t-1}^2 S_{t-1}$. This holds by the following chain:

$$\begin{aligned} R_t^2 S_t &= (R_{t-1} - \Delta R_t)^2 (S_{t-1} + \Delta S_t) = R_{t-1}^2 S_{t-1} \left(1 - \frac{\Delta R_t}{R_{t-1}}\right)^2 \left(1 + \frac{\Delta S_t}{S_{t-1}}\right) \\ &\leq R_{t-1}^2 S_{t-1} \exp\left(-\frac{2\Delta R_t}{R_{t-1}} + \frac{\Delta S_t}{S_{t-1}}\right) \leq R_{t-1}^2 S_{t-1}, \end{aligned}$$

where the first inequality follows from $e^x \geq 1 + x$, and the second one from Lemma 5. ■

This completes the proof of Theorem 1. Although our bound provides a rate polynomial in $B, \varepsilon^{-1}$ as desired by the conjecture in Schapire (2010), the exponents are rather large, and (we believe) not tight. One possible source of slack is the bound on $S_t$ in Lemma 4. Qualitatively, the distance $S_t$ to some solution having target loss should decrease with rounds, whereas Lemma 4 only says it does not increase too fast. Improving this will directly lead to a faster convergence rate. In particular, showing that $S_t$ never increases would imply a $B^2/\varepsilon$ rate of convergence. Whether or not the monotonicity of $S_t$ holds, we believe that the obtained rate bound is probably true, and state it as a conjecture.

**Conjecture 6** *For any $\boldsymbol{\lambda}^*$ and $\varepsilon > 0$, AdaBoost converges to within $L(\boldsymbol{\lambda}^*) + \varepsilon$ loss in $O(B^2/\varepsilon)$ rounds, where the order notation hides only absolute constants.*

As evidence supporting the conjecture, we show in the next section how a minor modification to AdaBoost can achieve the above rate.

### 3.2 Faster Rates For A Variant

In this section we introduce a new algorithm, AdaBoost.*S* , which will enjoy the much faster rate of convergence mentioned in Conjecture 6. AdaBoost.*S* is the same as AdaBoost, except that at the end of each round, the current combination of weak hypotheses is *scaled back*, that is, multiplied by a scalar in $[0,1]$ if doing so will reduce the exponential loss further. The code is largely the same as in Section 2, maintaining a combination $\boldsymbol{\lambda}^{t-1}$ of weak hypotheses, and greedily choosing $\alpha_t$ and $\hbar_{j_t}$ on each round to form a new combination $\tilde{\boldsymbol{\lambda}}^t = \boldsymbol{\lambda}^{t-1} + \alpha_t \hbar_{j_t}$. However, after creating the new combination $\tilde{\boldsymbol{\lambda}}^t$, the result is multiplied by the value $s_t$ in $[0,1]$ that causes the greatest decrease in the exponential loss: $s_t = \operatorname{argmin}_s L(s\tilde{\boldsymbol{\lambda}}^t)$, and then we assign $\boldsymbol{\lambda}^t = s_t \tilde{\boldsymbol{\lambda}}^t$. Since $L(s\tilde{\boldsymbol{\lambda}}^t)$, as a function of $s$, is convex, its minimum on $[0,1]$ can be found easily, for instance, using a simple binary search. The new distribution $D_{t+1}$ on the examples is constructed using $\boldsymbol{\lambda}^t$ as before; the weight $D_{t+1}(i)$ on

example $i$ is proportional to its exponential loss $D_{t+1}(i) \propto e^{-(\mathbf{M}\boldsymbol{\lambda}^t)_i}$. With this modification we prove the following:

**Theorem 7** *For any $\boldsymbol{\lambda}^*, \varepsilon > 0$, AdaBoost.*S *achieves at most $L(\boldsymbol{\lambda}^*) + \varepsilon$ loss within $3\|\boldsymbol{\lambda}^*\|_1^2/\varepsilon$ rounds.*

The proof is similar to that in the previous section. Reusing the same notation, note that the proof of Lemma 2 continues to hold (with very minor modifications to that are straightforward). Next we can exploit the changes in AdaBoost.*S* to show an improved version of Lemma 3. Intuitively, scaling back has the effect of preventing the weights on the weak hypotheses from becoming "too large", and we may show

**Lemma 8** *In each round $t$, the edge satisfies $\delta_t \geq R_{t-1}/B$.*

**Proof** We will reuse parts of the proof of Lemma 3. Setting $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$ in (7) we may write

$$R_{t-1} \leq \sum_{i=1}^{m} D_{\boldsymbol{\lambda}^{t-1}}(i) \left(\mathbf{M}\boldsymbol{\lambda}^*\right)_i + \sum_{i=1}^{m} -D_{\boldsymbol{\lambda}^{t-1}}(i) \left(\mathbf{M}\boldsymbol{\lambda}^{t-1}\right)_i.$$

The first summation can be upper bounded as in (8) by $\delta_t \|\boldsymbol{\lambda}^*\| = \delta_t B$. We will next show that the second summation is non-positive, which will complete the proof. The scaling step was added just so that this last fact would be true.

If we define $G : [0,1] \to \mathbb{R}$ to be $G(s) = L\left(s\tilde{\boldsymbol{\lambda}}^t\right) = (1/m)\sum_i e^{-s(\mathbf{M}\tilde{\boldsymbol{\lambda}}^t)_i}$, then observe that the scaled derivative $G'(s)/G(s)$ is exactly equal to the second summation. Since $G(s) \geq 0$, it suffices to show the derivative $G'(s) \leq 0$ at the optimum value of $s$, denoted by $s^*$. Since $G$ is a strictly convex function ($\forall s : G''(s) > 0$), it is either strictly increasing or strictly decreasing throughout $[0,1]$, or it has a local minimum. In the case when it is strictly decreasing throughout, then $G'(s) \leq 0$ everywhere. Otherwise, if $G$ has a local minimum, then $G'(s) = 0$ at $s^*$. We finish the proof by showing that $G$ cannot be strictly increasing throughout $[0,1]$. If it were, we would have $L(\tilde{\boldsymbol{\lambda}}^t) = G(1) > G(0) = 1$, an impossibility since the loss decreases through rounds. ■

The above lemma implies the conditions in Lemma 2 hold if we set $c_1 = c_2 = 1$. The result of Lemma 2 now implies Theorem 7, where we used that $2\ln 2 < 3$.

In experiments we ran, the scaling back never occurs. For such data sets, AdaBoost and AdaBoost.*S* are identical. We believe that even for contrived examples, the rescaling could happen only a few times, implying that both AdaBoost and AdaBoost.*S* would enjoy the convergence rates of Theorem 7. In the next section, we construct rate lower bound examples to show that this is nearly the best rate one can hope to show.

### 3.3 Lower Bounds

Here we show that the dependence of the rate in Theorem 1 on the norm $\|\boldsymbol{\lambda}^*\|_1$ of the solution achieving target accuracy is necessary for a wide class of data sets. The arguments in this section are not tailored to AdaBoost, but hold more generally for any coordinate descent algorithm, and can be readily generalized to any loss function $\tilde{L}$ of the form $\tilde{L}(\boldsymbol{\lambda}) = (1/m)\sum_i \phi(-\mathbf{M}\boldsymbol{\lambda})$, where $\phi : \mathbb{R} \to \mathbb{R}$ is any non-decreasing function. For instance, with the exponential loss, $\phi$ is the exponential function, and the lower-bound results have a logarithmic term in them. For general non-decreasing functions $\phi$, similar arguments yield bounds which are identical to the ones in this section, except the logarithmic terms are replaced by the inverse function $\phi^{-1}$.

The first lemma connects the size of a reference solution to the required number of rounds of boosting, and shows that for a wide variety of data sets the convergence rate to a target loss can be lower bounded by the $\ell_1$-norm of the smallest solution achieving that loss.

**Lemma 9** *Suppose the feature matrix* $\mathbf{M}$ *corresponding to a data set has two rows with* $\{-1,+1\}$ *entries which are complements of each other, that is, there are two examples on which any hypothesis gets one wrong and one correct prediction. Then the number of rounds required to achieve a target loss* $L^*$ *is at least* $\inf\{\|\boldsymbol{\lambda}\|_1 : L(\boldsymbol{\lambda}) \leq L^*\}/(2\ln m)$.

**Proof** We first show that the two examples corresponding to the complementary rows in $\mathbf{M}$ both satisfy a certain margin boundedness property. Since each hypothesis predicts oppositely on these, in any round $t$ their margins will be of equal magnitude and opposite sign. Unless both margins lie in $[-\ln m, \ln m]$, one of them will be smaller than $-\ln m$. But then the exponential loss $L(\boldsymbol{\lambda}^t) = (1/m)\sum_j e^{-(\mathbf{M}\boldsymbol{\lambda}^t)_j}$ in that round will exceed 1, a contradiction since the losses are non-increasing through rounds, and the loss at the start was 1. Thus, assigning one of these examples the index $i$, we have the absolute margin $|(\mathbf{M}\boldsymbol{\lambda}^t)_i|$ is bounded by $\ln m$ in any round $t$. Letting $\mathbf{M}(i)$ denote the $i$th row of $\mathbf{M}$, the step length $\alpha_t$ in round $t$ therefore satisfies

$$|\alpha_t| = \left|M_{ij_t}\alpha_t\right| = \left|\left\langle \mathbf{M}(i), \alpha_t \vec{e}_{j_t}\right\rangle\right| = \left|(\mathbf{M}\boldsymbol{\lambda}^t)_i - (\mathbf{M}\boldsymbol{\lambda}^{t-1})_i\right| \leq \left|(\mathbf{M}\boldsymbol{\lambda}^t)_i\right| + \left|(\mathbf{M}\boldsymbol{\lambda}^{t-1})_i\right| \leq 2\ln m,$$

and the statement of the lemma directly follows. ■

When the weak hypotheses are *abstaining* (Schapire and Singer, 1999), a hypothesis can make a definitive prediction that the label is $-1$ or $+1$, or it can "abstain" by predicting zero. No other levels of confidence are allowed, and the resulting feature matrix has entries in $\{-1,0,+1\}$. The next theorem constructs a feature matrix satisfying the properties of Lemma 9 and where additionally the smallest size of a solution achieving $L^* + \varepsilon/m$ loss is at least $\Omega(2^m)\ln(1/\varepsilon)$, for some fixed $L^*$ and every $\varepsilon > 0$.

**Theorem 10** *Consider the following matrix* $\mathbf{M}$ *with* $m$ *rows (or examples) (for* $m \geq 2$*) labeled* $0,\ldots,m-1$ *and* $m-1$ *columns labeled* $1,\ldots,m-1$*. The square sub-matrix ignoring row zero is an upper triangular matrix, with* $1$*'s on the diagonal,* $-1$*'s above the diagonal, and* $0$ *below the diagonal. Therefore row 1 is* $(+1,-1,-1,\ldots,-1)$*. Row 0 is defined to be just the complement of row 1. Then, for any* $\varepsilon > 0$*, a loss of* $2/m + \varepsilon/m$ *is achievable on this data set, but with large norms*

$$\inf\{\|\boldsymbol{\lambda}\|_1 : L(\boldsymbol{\lambda}) \leq 2/m + \varepsilon/m\} \geq (2^{m-2}-1)\ln(1/\varepsilon).$$

*Therefore, by Lemma 9, the minimum number of rounds required for reaching loss at most* $2/m+\varepsilon$ *is at least* $\left(\frac{2^{m-2}-1}{2\ln m}\right)\ln(1/\varepsilon)$.

A picture of the matrix constructed in the above lemma for $m = 6$ is shown in Figure 2. Theorem 10 shows that when $\varepsilon$ is a small constant (say $\varepsilon = 0.01$), and $\boldsymbol{\lambda}^*$ is some vector with loss $L^* + \varepsilon/(2m)$, AdaBoost takes at least $\Omega(2^m/\ln m)$ steps to get within $\varepsilon/(2m)$ of the loss achieved by $\boldsymbol{\lambda}^*$. Since $m$ and $\varepsilon$ are independent quantities, this shows that worst case bounds can be exponential in $m$. Thus, a polynomial dependence on the norm of the reference solution is unavoidable, and this norm might be exponential in the number of training examples in the worst case.

$$\begin{array}{r} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ m-1=5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5=m-1 \end{array} \\ \begin{pmatrix} - & + & + & + & + \\ + & - & - & - & - \\ 0 & + & - & - & - \\ 0 & 0 & + & - & - \\ 0 & 0 & 0 & + & - \\ 0 & 0 & 0 & 0 & + \end{pmatrix} \end{array}$$

Figure 2: The matrix used in Theorem 10 when $m = 6$. Note that $+$ is the same as $+1$, referring to a correct prediction, and $-$ the same as $-1$, referring to an incorrect prediction.

**Corollary 11** *Consider feature matrices containing only $\{-1,0,+1\}$ entries. If, for some constants $c$ and $\beta$, the bound in Theorem 1 can be replaced by $O\left(\|\boldsymbol{\lambda}^*\|_1^c \varepsilon^{-\beta}\right)$ for all such matrices, then $c \geq 1$. Further, for such matrices, the bound $\mathrm{poly}(1/\varepsilon, \|\boldsymbol{\lambda}^*\|_1)$ in Theorem 1 cannot be replaced by a uniform $\mathrm{poly}(1/\varepsilon, m, N)$ bound that holds for all values of $m, \varepsilon$ and $N$.*

We now prove Theorem 10.

*Proof of Theorem 10.* We first lower bound the norm of solutions achieving loss at most $2/m+\varepsilon$. Observe that since rows 0 and 1 are complementary, then $-(\mathbf{M}\boldsymbol{\lambda})_0 = (\mathbf{M}\boldsymbol{\lambda})_1$, and any solution's loss on just examples 0 and 1 will add up to $1/m[\exp(-(\mathbf{M}\boldsymbol{\lambda})_1) + \exp(-(\mathbf{M}\boldsymbol{\lambda})_2)]$ which is at least $2/m$. Therefore, to get within $2/m+\varepsilon/m$, the margins on examples $2,\ldots,m-1$ should be at least $\ln(1/\varepsilon)$. Now, the feature matrix is designed so that the margins due to a combination $\boldsymbol{\lambda}$ satisfy the following recursive relationships:

$$\begin{aligned} (\mathbf{M}\boldsymbol{\lambda})_{m-1} &= \lambda_{m-1}, \\ (\mathbf{M}\boldsymbol{\lambda})_i &= \lambda_i - (\lambda_{i+1} + \ldots + \lambda_{m-1}), \text{ for } 1 \leq i \leq m-2. \end{aligned}$$

Therefore, the margin of example $m-1$ is at least $\ln(1/\varepsilon)$, then by the first equation above, $\lambda_{m-1} \geq \ln(1/\varepsilon)$. Similarly, $\lambda_{m-2} \geq \ln(1/\varepsilon) + \lambda_{m-1} \geq 2\ln(1/\varepsilon)$. Continuing this way, if the margin of example $i$ is at least $\ln(1/\varepsilon)$, then

$$\lambda_i \geq \ln\left(\frac{1}{\varepsilon}\right) + \lambda_{i+1} + \ldots + \lambda_{m-1} \geq \ln\left(\frac{1}{\varepsilon}\right)\left\{1 + 2^{(m-1)-(i+1)} + \ldots + 2^0\right\} = \ln\left(\frac{1}{\varepsilon}\right) 2^{m-1-i},$$

for $i = m-1,\ldots,2$. Hence $\|\boldsymbol{\lambda}\|_1 \geq \ln(1/\varepsilon)(1+2+\ldots+2^{m-3}) = (2^{m-2}-1)\ln(1/\varepsilon)$.

We end by showing that a loss of at most $2/m+\varepsilon/m$ is achievable. The above argument implies that if $\lambda_i = 2^{m-1-i}$ for $i = 2,\ldots,m-1$, then examples $2,\ldots,m-1$ attain margin exactly 1. If we choose $\lambda_1 = \lambda_2 + \ldots + \lambda_{m-1} = 2^{m-3} + \ldots + 1 = 2^{m-2} - 1$, then the recursive relationship implies a zero margin on example 1 (and hence example 0). Therefore the combination $(2^{m-2}-1, 2^{m-3}, 2^{m-4}, \ldots, 1)$ achieves margins $(0,0,1,1,1,\ldots,1)$. Hence when scaled by a factor of $\ln((m-2)/\varepsilon)$, this combination achieves a loss $\left(2+(m-2)\frac{\varepsilon}{m-2}\right)/m = 2/m+\varepsilon/m$, for any $\varepsilon > 0$. ■

We finally show that if the weak hypotheses are confidence-rated with arbitrary levels of confidence, so that the feature matrix is allowed to have non-integral entries in $[-1,+1]$, then the

$$\begin{pmatrix} -1 & +1 \\ +1 & -1 \\ -1+\nu & +1 \\ +1 & -1+\nu \end{pmatrix}$$

Figure 3: A picture of the matrix used in Theorem 12.

minimum norm of a solution achieving a fixed accuracy can be arbitrarily large. Our constructions will satisfy the requirements of Lemma 9, so that the norm lower bound translates into a rate lower bound.

**Theorem 12** *Let $\nu > 0$ be an arbitrary number, and let $\mathbf{M}$ be the (possibly) non-integral matrix with 4 examples and 2 weak hypotheses shown in Figure 3. Then for any $\varepsilon > 0$, a loss of $1/2+\varepsilon$ is achievable on this data set, but with large norms*

$$\inf\{\|\boldsymbol{\lambda}\|_1 : L(\boldsymbol{\lambda}) \leq 1/2+\varepsilon\} \geq 2\ln(1/(4\varepsilon))\nu^{-1}.$$

*Therefore, by Lemma 9, the number of rounds required to achieve loss at most $1/2+\varepsilon$ is at least $\ln(1/(4\varepsilon))\nu^{-1}/\ln(m)$.*

**Proof** We first show a loss of $1/2+\varepsilon$ is achievable. Observe that the vector $\boldsymbol{\lambda} = (c,c)$, with $c = \nu^{-1}\ln(1/(2\varepsilon))$, achieves margins $0, 0, \ln(1/(2\varepsilon)), \ln(1/(2\varepsilon))$ on examples $1,2,3,4$, respectively. Therefore $\boldsymbol{\lambda}$ achieves loss $1/2+\varepsilon$. We next show a lower bound on the norm of a solution achieving this loss. Observe that since the first two rows are complementary, the loss due to just the first two examples is at least $1/2$. Therefore, any solution $\boldsymbol{\lambda} = (\lambda_1, \lambda_2)$ achieving at most $1/2+\varepsilon$ loss overall must achieve a margin of at least $\ln(1/(4\varepsilon))$ on both the third and fourth examples. By inspecting these two rows, this implies

$$\begin{aligned} \lambda_2 - \lambda_1 + \lambda_1\nu &\geq \ln(1/(4\varepsilon)), \\ \lambda_1 - \lambda_2 + \lambda_2\nu &\geq \ln(1/(4\varepsilon)). \end{aligned}$$

Adding the two equations we find

$$\nu(\lambda_1+\lambda_2) \geq 2\ln(1/(4\varepsilon)) \implies \lambda_1+\lambda_2 \geq 2\nu^{-1}\ln(1/(4\varepsilon)).$$

By the triangle inequality, $\|\boldsymbol{\lambda}\|_1 \geq \lambda_1+\lambda_2$, and the lemma follows. ■

Note that if $\nu = 0$, then an optimal solution is found in zero rounds of boosting and has optimal loss 1. However, even the tiniest perturbation $\nu > 0$ causes the optimal loss to fall to $1/2$, and causes the rate of convergence to increase drastically. In fact, by Theorem 12, the number of rounds required to achieve any fixed loss below 1 grows as $\Omega(1/\nu)$, which is arbitrarily large when $\nu$ is infinitesimal. We may conclude that with non-integral feature matrices, the dependence of the rate on the norm of a reference solution is absolutely necessary.

**Corollary 13** *When using confidence rated weak-hypotheses with arbitrary confidence levels, the bound $\mathrm{poly}(1/\varepsilon, \|\boldsymbol{\lambda}^*\|_1)$ in Theorem 1 cannot be replaced by any function of purely m, N and $\varepsilon$ alone.*

The construction in Figure 3 can be generalized to produce data sets with any number of examples that suffer the same poor rate of convergence as the one in Theorem 12. We discussed the smallest such construction, since we feel that it best highlights the drastic effect non-integrality can have on the rate.

In this section we saw how the norm of the reference solution is an important parameter for bounding the convergence rate. In the next section we investigate the optimal dependence of the rate on the parameter $\varepsilon$ and show that $\Omega(1/\varepsilon)$ rounds are necessary in the worst case.

## 4. Second Convergence Rate: Convergence to Optimal Loss

In the previous section, our rate bound depended on both the approximation parameter $\varepsilon$, as well as the size of the smallest solution achieving the target loss. For many data sets, the optimal target loss $\inf_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda})$ cannot be realized by any finite solution. In such cases, if we want to bound the number of rounds needed to achieve within $\varepsilon$ of the optimal loss, the only way to use Theorem 1 is to first decompose the accuracy parameter $\varepsilon$ into two parts $\varepsilon = \varepsilon_1 + \varepsilon_2$, find some finite solution $\boldsymbol{\lambda}^*$ achieving within $\varepsilon_1$ of the optimal loss, and then use the bound $\text{poly}(1/\varepsilon_2, \|\boldsymbol{\lambda}^*\|_1)$ to achieve at most $L(\boldsymbol{\lambda}^*) + \varepsilon_2 = \inf_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda}) + \varepsilon$ loss. However, this introduces implicit dependence on $\varepsilon$ through $\|\boldsymbol{\lambda}^*\|_1$ which may not be immediately clear. In this section, we show bounds of the form $C/\varepsilon$, where the constant $C$ depends only on the feature matrix $\mathbf{M}$, and not on $\varepsilon$. Additionally, we show that this dependence on $\varepsilon$ is optimal in Lemma 31 of the Appendix, where $\Omega(1/\varepsilon)$ rounds are shown to be necessary for converging to within $\varepsilon$ of the optimal loss on a certain data set. Finally, we note that the lower bounds in the previous section indicate that $C$ can be $\Omega(2^m)$ in the worst case for integer matrices (although it will typically be much smaller), and hence this bound, though stronger than that of Theorem 1 with respect to $\varepsilon$, cannot be used to prove the conjecture in Schapire (2010), since the constant is not polynomial in the number of examples $m$.

### 4.1 Upper Bound

The main result of this section is the following rate upper bound. A similar approach to solving this problem was taken independently by Telgarsky (2011).

**Theorem 14** *AdaBoost reaches within $\varepsilon$ of the optimal loss in at most $C/\varepsilon$ rounds, where $C$ only depends on the feature matrix.*

Our techniques build upon earlier work on the rate of convergence of AdaBoost, which have mainly considered two particular cases. In the first case, the *weak learning assumption* holds, that is, the edge in each round is at least some fixed constant. In this situation, Freund and Schapire (1997) and Schapire and Singer (1999) show that the optimal loss is zero, that no solution with finite size can achieve this loss, but AdaBoost achieves at most $\varepsilon$ loss within $O(\ln(1/\varepsilon))$ rounds. In the second case some finite combination of the weak classifiers achieves the optimal loss, and Rätsch et al. (2002), using results from Luo and Tseng (1992), show that AdaBoost achieves within $\varepsilon$ of the optimal loss again within $O(\ln(1/\varepsilon))$ rounds.

Here we consider the most general situation, where the weak learning assumption may fail to hold, and yet no finite solution may achieve the optimal loss. The data set used in Lemma 31 and shown in Figure 4 exemplifies this situation. Our main technical contribution shows that the examples in any data set can be partitioned into a *zero-loss set* and *finite-margin set*, such that a certain form of the weak learning assumption holds within the zero-loss set, while the optimal loss

considering only the finite-margin set can be obtained by some finite solution. The two partitions provide different ways of making progress in every round, and one of the two kinds of progress will always be sufficient for us to prove Theorem 14.

We next state our decomposition result, illustrate it with an example, and then state several lemmas quantifying the nature of the progress we can make in each round. Using these lemmas, we prove Theorem 14.

**Lemma 15** *(Decomposition Lemma) For any data set, there exists a partition of the set of training examples $X$ into a (possibly empty)* zero-loss set *$Z$ and a (possibly empty)* finite-margin set *$F \triangleq Z^c = X \setminus Z$ such that the following hold simultaneously :*

1. *There exists some positive constant $\gamma > 0$, and some vector $\boldsymbol{\eta}^\dagger$ with unit $\ell_1$-norm $\|\boldsymbol{\eta}^\dagger\|_1 = 1$ that attains at least $\gamma$ margin on each example in $Z$, and exactly zero margin on each example in $F$*

$$\forall i \in Z : (\mathbf{M}\boldsymbol{\eta}^\dagger)_i \geq \gamma, \qquad\qquad \forall i \in F : (\mathbf{M}\boldsymbol{\eta}^\dagger)_i = 0.$$

2. *The optimal loss considering only examples within $F$ is achieved by some finite combination $\boldsymbol{\eta}^*$. (Note that $\boldsymbol{\eta}^*$ may not be unique. There may be a whole subspace of vectors like $\boldsymbol{\eta}^*$ that achieve the optimal loss on $F$.)*

3. *There is a constant $\mu_{\max} < \infty$, such that for any combination $\boldsymbol{\eta}$ with bounded loss on the finite-margin set, specifically obeying $\sum_{i \in F} e^{-(\mathbf{M}\boldsymbol{\eta})_i} \leq m$, the margin $(\mathbf{M}\boldsymbol{\eta})_i$ for any example $i$ in $F$ lies in the bounded interval $[-\ln m, \mu_{\max}]$.*

A proof is deferred to the next section. The decomposition lemma immediately implies that the vector $\boldsymbol{\eta}^* + \infty \cdot \boldsymbol{\eta}^\dagger$, which denotes $\left(\boldsymbol{\eta}^* + c\boldsymbol{\eta}^\dagger\right)$ in the limit $c \to \infty$, is an optimal solution, achieving zero loss on the zero-loss set, but only finite margins (and hence positive losses) on the finite-margin set (thereby justifying the names).

| | $\hbar_1$ | $\hbar_2$ |
|---|---|---|
| $a$ | $+$ | $-$ |
| $b$ | $-$ | $+$ |
| $c$ | $+$ | $+$ |

Figure 4: Matrix $\mathbf{M}$ for a data set requiring $\Omega(1/\varepsilon)$ rounds for convergence.

Before proceeding, we give an example data set and indicate the zero-loss set, finite-margin set, $\boldsymbol{\eta}^*$ and $\boldsymbol{\eta}^\dagger$ to illustrate our definitions. Consider a data set with three examples $\{a, b, c\}$ and two hypotheses $\{\hbar_1, \hbar_2\}$ and the feature matrix $\mathbf{M}$ in Figure 4. Here $+$ means correct ($M_{ij} = +1$) and $-$ means wrong ($M_{ij} = -1$). The optimal solution is $\infty \cdot (\hbar_1 + \hbar_2)$ with a loss of $2/3$. The finite-margin set is $\{a, b\}$, the zero-loss set is $\{c\}$, $\boldsymbol{\eta}^\dagger = (1/2, 1/2)$ and $\boldsymbol{\eta}^* = (0, 0)$; for this data set these are unique. This data set also serves as a lower-bound example in Lemma 31, where we show that $2/(9\varepsilon)$ rounds are necessary for AdaBoost to achieve loss at most $(2/3) + \varepsilon$, where the infimum of the loss is 2/3. Observe that this data set is similar to the data set in Theorem 10 in that the zero-loss set (corresponding to the complementary rows) and finite-loss sets (remaining examples) are both non-empty.

Before providing proofs, we introduce some notation. By $\|\cdot\|$ we will mean $\ell_2$-norm; every other norm will have an appropriate subscript, such as $\|\cdot\|_1, \|\cdot\|_\infty$, etc. The set of all training examples will be denoted by $X$. By $\ell^{\boldsymbol{\lambda}}(i)$ we mean the exp-loss $e^{-(\mathbf{M}\boldsymbol{\lambda})_i}$ on example $i$. For any subset $S \subseteq X$ of examples, $\ell^{\boldsymbol{\lambda}}(S) = \sum_{i\in S} \ell^{\boldsymbol{\lambda}}(i)$ denotes the total exp-loss on the set $S$. Notice $L(\boldsymbol{\lambda}) = (1/m)\ell^{\boldsymbol{\lambda}}(X)$, and that $D_{t+1}(i) = \ell^{\boldsymbol{\lambda}^t}(i)/\ell^{\boldsymbol{\lambda}^t}(X)$, where $\boldsymbol{\lambda}^t$ is the combination found by AdaBoost at the end of round $t$. By $\delta_S(\boldsymbol{\eta};\boldsymbol{\lambda})$ we mean the edge obtained on the set $S$ by the vector $\boldsymbol{\eta}$, when the weights over the examples are given by $\ell^{\boldsymbol{\lambda}}(\cdot)/\ell^{\boldsymbol{\lambda}}(S)$:

$$\delta_S(\boldsymbol{\eta};\boldsymbol{\lambda}) = \left| \frac{1}{\ell^{\boldsymbol{\lambda}}(S)} \sum_{i\in S} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\boldsymbol{\eta})_i \right|.$$

In the rest of the section, by "loss" we mean the unnormalized loss $\ell^{\boldsymbol{\lambda}}(X) = mL(\boldsymbol{\lambda})$ and show that in $C/\varepsilon$ rounds AdaBoost converges to within $\varepsilon$ of the optimal unnormalized loss $\inf_{\boldsymbol{\lambda}} \ell^{\boldsymbol{\lambda}}(X)$, henceforth denoted by $K$. Note that this means AdaBoost takes $C/\varepsilon$ rounds to converge to within $\varepsilon/m$ of the optimal normalized loss, that is to say at most $\inf_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda}) + \varepsilon/m$ loss. Replacing $\varepsilon$ by $m\varepsilon$, it takes $C/(m\varepsilon)$ steps to attain normalized loss at most $\inf_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda}) + \varepsilon$. Thus, whether we use normalized or unnormalized does not substantively affect the result in Theorem 14. The progress due to the zero-loss set is now immediate from Item 1 of the decomposition lemma:

**Lemma 16** *In any round $t$, the maximum edge $\delta_t$ is at least $\gamma \ell^{\boldsymbol{\lambda}^{t-1}}(Z)/\ell^{\boldsymbol{\lambda}^{t-1}}(X)$, where $\gamma$ is as in Item 1 of the decomposition lemma.*

**Proof** Recall the distribution $D_t$ created by AdaBoost in round $t$ puts weight $D_t(i) = \ell^{\boldsymbol{\lambda}^{t-1}}(i)/\ell^{\boldsymbol{\lambda}^{t-1}}(X)$ on each example $i$. From Item 1 we get

$$\delta_X(\boldsymbol{\eta}^\dagger;\boldsymbol{\lambda}^{t-1}) = \left| \frac{1}{\ell^{\boldsymbol{\lambda}^{t-1}}(X)} \sum_{i\in X} \ell^{\boldsymbol{\lambda}^{t-1}}(i)(\mathbf{M}\boldsymbol{\eta}^\dagger)_i \right| \geq \frac{1}{\ell^{\boldsymbol{\lambda}^{t-1}}(X)} \sum_{i\in Z} \gamma \ell^{\boldsymbol{\lambda}^{t-1}}(i) = \gamma \left( \frac{\ell^{\boldsymbol{\lambda}^{t-1}}(Z)}{\ell^{\boldsymbol{\lambda}^{t-1}}(X)} \right).$$

Since $(\mathbf{M}\boldsymbol{\eta}^\dagger)_i = \sum_j \eta_j^\dagger (\mathbf{M}\vec{e}_j)_i$, we may rewrite the edge $\delta_X(\boldsymbol{\eta}^\dagger;\boldsymbol{\lambda}^{t-1})$ as follows:

$$\begin{aligned}
\delta_X(\boldsymbol{\eta}^\dagger;\boldsymbol{\lambda}^{t-1}) &= \left| \frac{1}{\ell^{\boldsymbol{\lambda}^{t-1}}(X)} \sum_{i\in X} \ell^{\boldsymbol{\lambda}^{t-1}}(i) \sum_j \eta_j^\dagger (\mathbf{M}\vec{e}_j)_i \right| \\
&= \left| \sum_j \eta_j^\dagger \frac{1}{\ell^{\boldsymbol{\lambda}^{t-1}}(X)} \sum_{i\in X} \ell^{\boldsymbol{\lambda}^{t-1}}(i)(\mathbf{M}\vec{e}_j)_i \right| \\
&= \left| \sum_j \eta_j^\dagger \delta_X(\vec{e}_j;\boldsymbol{\lambda}^{t-1}) \right| \leq \sum_j \left|\eta_j^\dagger\right| \delta_X(\vec{e}_j;\boldsymbol{\lambda}^{t-1}).
\end{aligned}$$

Since the $\ell_1$-norm of $\boldsymbol{\eta}^\dagger$ is 1, the weights $\left|\eta_j^\dagger\right|$ form some distribution $p$ over the columns $1,\ldots,N$. We may therefore conclude

$$\gamma \left( \frac{\ell^{\boldsymbol{\lambda}^{t-1}}(Z)}{\ell^{\boldsymbol{\lambda}^{t-1}}(X)} \right) \leq \delta_X(\boldsymbol{\eta}^\dagger;\boldsymbol{\lambda}^{t-1}) \leq \mathbb{E}_{j\sim p}\left[\delta_X(\vec{e}_j;\boldsymbol{\lambda}^{t-1})\right] \leq \max_j \delta_X(\vec{e}_j;\boldsymbol{\lambda}^{t-1}) = \delta_t.$$

■

If the set $F$ were empty, then Lemma 16 implies an edge of $\gamma$ is available in each round. This in fact means that the weak learning assumption holds, and using (5), we can show an $O(\ln(1/\varepsilon)\gamma^{-2})$ bound matching the rate bounds of Freund and Schapire (1997) and Schapire and Singer (1999). This is in fact the "separable" case where positive and negative examples can be separated by AdaBoost. Henceforth, we assume that $F$ is non-empty. This implies that the optimal loss $K$ is at least 1 (since we are in the nonseparable case, and any solution will get non-positive margin on some example in $F$), a fact that we will use later in the proofs. In the separable case, the minimum normalized margin becomes an important performance instead of the exponential loss. Convergence with respect to the normalized margin has been studied by Schapire et al. (1998), Grove and Schuurmans (1998), Rudin et al. (2004), Rätsch and Warmuth (2005), and others.

Lemma 16 says that the edge is large if the loss on the zero-loss set is large. On the other hand, when it is small, Lemmas 17 and 18 together show how AdaBoost can make good progress using the finite margin set. Lemma 17 uses second order methods to show how progress is made in the case where there is a finite solution. Similar arguments, under additional assumptions, have earlier appeared in Rätsch et al. (2002).

**Lemma 17** *Suppose $\boldsymbol{\lambda}$ is a combination such that $m \geq \ell^{\boldsymbol{\lambda}}(F) \geq K$. Then in some coordinate direction $\vec{e}_j$ the edge $\delta_F(\vec{e}_j;\boldsymbol{\lambda})$ is at least $\sqrt{C_0\left(\ell^{\boldsymbol{\lambda}}(F)-K\right)/\ell^{\boldsymbol{\lambda}}(F)}$, where $C_0$ is a constant depending only on the feature matrix $\mathbf{M}$.*

**Proof** Let $\mathbf{M}_F \in \mathbb{R}^{|F|\times N}$ be the matrix $\mathbf{M}$ restricted to only the rows corresponding to the examples in $F$. Choose $\boldsymbol{\eta}$ such that $\boldsymbol{\lambda}+\boldsymbol{\eta}=\boldsymbol{\eta}^*$ is an optimal solution over $F$. Without loss of generality assume that $\boldsymbol{\eta}$ lies in the orthogonal subspace of the null-space $\{\vec{u} : \mathbf{M}_F\vec{u}=\mathbf{0}\}$ of $\mathbf{M}_F$ (since we can translate $\boldsymbol{\eta}^*$ along the null space if necessary for this to hold). If $\boldsymbol{\eta}=\mathbf{0}$, then $\ell^{\boldsymbol{\lambda}}(F)=K$ and we are done. Otherwise $\|\mathbf{M}_F\boldsymbol{\eta}\| \geq \lambda_{\min}\|\boldsymbol{\eta}\|$, where $\lambda^2_{\min}$ is the smallest positive eigenvalue of the symmetric matrix $\mathbf{M}_F^T\mathbf{M}_F$ (which exists since $\mathbf{M}_F\boldsymbol{\eta}\neq\mathbf{0}$). Now define $f:[0,1]\to\mathbb{R}$ as the loss along the (rescaled) segment $[\boldsymbol{\eta}^*,\boldsymbol{\lambda}]$

$$f(x) \triangleq \ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(F) = \sum_{i\in F}\ell^{\boldsymbol{\eta}^*}(i)e^{x(\mathbf{M}_F\boldsymbol{\eta})_i}.$$

This implies that $f(0)=K$ and $f(1)=\ell^{\boldsymbol{\lambda}}(F)$. Notice that the first and second derivatives of $f(x)$ are given by:

$$f'(x)=\sum_{i\in F}(\mathbf{M}_F\boldsymbol{\eta})_i\ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(i), \qquad f''(x)=\sum_{i\in F}(\mathbf{M}_F\boldsymbol{\eta})_i^2\ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(i).$$

Since $f''(x)\geq 0$, $f$ is convex; further it attains the minimum value at 0. Therefore $f$ is increasing, and $f'(x)\geq 0$ at all points. We next lower bound possible values of the second derivative as follows:

$$f''(x)=\sum_{i'\in F}(\mathbf{M}_F\boldsymbol{\eta})_{i'}^2\ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(i') \geq \sum_{i'\in F}(\mathbf{M}_F\boldsymbol{\eta})_{i'}^2\min_i\ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(i) = \|\mathbf{M}_F\boldsymbol{\eta}\|^2\min_i\ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(i).$$

Since both $\boldsymbol{\lambda}=\boldsymbol{\eta}^*-\boldsymbol{\eta}$, and $\boldsymbol{\eta}^*$ suffer total loss at most $m$, by convexity, so does $\boldsymbol{\eta}^*-x\boldsymbol{\eta}$ for any $x\in[0,1]$. Hence we may apply Item 3 of the decomposition lemma to the vector $\boldsymbol{\eta}^*-x\boldsymbol{\eta}$, for any $x\in[0,1]$, to conclude that $\ell^{(\boldsymbol{\eta}^*-x\boldsymbol{\eta})}(i)=\exp\{-(\mathbf{M}_F(\boldsymbol{\eta}^*-x\boldsymbol{\eta}))_i\}\geq e^{-\mu_{\max}}$ on every example $i$. Therefore we have

$$f''(x)\geq\|\mathbf{M}_F\boldsymbol{\eta}\|^2e^{-\mu_{\max}}\geq\lambda^2_{\min}e^{-\mu_{\max}}\|\boldsymbol{\eta}\|^2 \text{ (by choice of } \boldsymbol{\eta}) . \tag{9}$$

A standard second-order result for a twice-differentiable function $f$ is (see, e.g., Boyd and Vandenberghe, 2004, Equation (9.9))

$$\left|f'(1)\right|^2 \geq 2\left(\inf_{x\in[0,1]} f''(x)\right)(f(1)-f(0)). \tag{10}$$

Collecting our results so far, including the definition of $f'$ and the fact that it is non-negative, and equations (9) and (10), we get

$$\sum_{i\in F} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\boldsymbol{\eta})_i = f'(1) = \left|f'(1)\right| \geq \|\boldsymbol{\eta}\|\sqrt{2\lambda_{\min}^2 e^{-\mu_{\max}}(\ell^{\boldsymbol{\lambda}}(F)-K)}.$$

Next let $\tilde{\boldsymbol{\eta}} = \boldsymbol{\eta}/\|\boldsymbol{\eta}\|_1$ be $\boldsymbol{\eta}$ rescaled to have unit $\ell_1$ norm. Then dividing the previous equation by $\|\boldsymbol{\eta}\|_1$

$$\sum_{i\in F} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\tilde{\boldsymbol{\eta}})_i = \frac{1}{\|\boldsymbol{\eta}\|_1}\sum_i \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\boldsymbol{\eta})_i \geq \frac{\|\boldsymbol{\eta}\|}{\|\boldsymbol{\eta}\|_1}\sqrt{2\lambda_{\min}^2 e^{-\mu_{\max}}(\ell^{\boldsymbol{\lambda}}(F)-K)}.$$

Applying the Cauchy-Schwarz inequality, we may lower bound $\frac{\|\boldsymbol{\eta}\|}{\|\boldsymbol{\eta}\|_1}$ by $1/\sqrt{N}$ (since $\boldsymbol{\eta}\in\mathbb{R}^N$). Along with the fact $\ell^{\boldsymbol{\lambda}}(F)\leq m$, we may write

$$\frac{1}{\ell^{\boldsymbol{\lambda}}(F)}\sum_{i\in F} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\tilde{\boldsymbol{\eta}})_i \geq \sqrt{2\lambda_{\min}^2 N^{-1}m^{-1}e^{-\mu_{\max}}}\sqrt{(\ell^{\boldsymbol{\lambda}}(F)-K)/\ell^{\boldsymbol{\lambda}}(F)}.$$

If we define $p$ to be a distribution on the columns $\{1,\ldots,N\}$ of $\mathbf{M}_F$ which puts probability $p(j)$ proportional to $\left|\tilde{\boldsymbol{\eta}}_j\right|$ on column $j$, then we have

$$\frac{1}{\ell^{\boldsymbol{\lambda}}(F)}\sum_{i\in F} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\tilde{\boldsymbol{\eta}})_i \leq \mathbb{E}_{j\sim p}\left|\frac{1}{\ell^{\boldsymbol{\lambda}}(F)}\sum_{i\in F} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\vec{e}_j)_i\right| \leq \max_j\left|\frac{1}{\ell^{\boldsymbol{\lambda}}(F)}\sum_{i\in F} \ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}_F\vec{e}_j)_i\right|.$$

Notice the quantity inside the max is precisely the edge $\delta_F(\vec{e}_j;\boldsymbol{\lambda})$ in direction $j$. Combining the previous two lines, the maximum possible edge is

$$\max_j \delta_F(\vec{e}_j;\boldsymbol{\lambda}) \geq \sqrt{C_0\,(\ell^{\boldsymbol{\lambda}}(F)-K)/\ell^{\boldsymbol{\lambda}}(F)},$$

where we define

$$C_0 = 2\lambda_{\min}^2 N^{-1}m^{-1}e^{-\mu_{\max}}. \tag{11}$$

■

**Lemma 18** *There are constants $C_1, C_2$ depending only on the feature matrix $M$ with the following property. Suppose, at any stage of boosting, the combination found by AdaBoost is $\boldsymbol{\lambda}$, and the loss is $K+\theta$. Let $\Delta\theta$ denote the drop in the suboptimality $\theta$ after one more round; that is, the loss after one more round is $K+\theta-\Delta\theta$. In this situation, if $\ell^{\boldsymbol{\lambda}}(Z)\leq C_1\theta$, for $C_1\leq 1$, then $\Delta\theta\geq C_2\theta$.*

**Proof** Let $\boldsymbol{\lambda}$ be the current solution found by boosting. Using Lemma 17, pick a direction $j$ in which the edge $\delta_F(\vec{e}_j;\boldsymbol{\lambda})$ restricted to the finite margin set is at least

$$\delta_F(\vec{e}_j;\boldsymbol{\lambda}) \geq \sqrt{C_0(\ell^{\boldsymbol{\lambda}}(F)-K)/\ell^{\boldsymbol{\lambda}}(F)}. \tag{12}$$

We can bound the edge $\delta_X(\vec{e}_j;\boldsymbol{\lambda})$ on the entire set of examples as follows:

$$\begin{aligned}
\delta_X(\vec{e}_j;\boldsymbol{\lambda}) &= \frac{1}{\ell^{\boldsymbol{\lambda}}(X)}\left|\sum_{i\in F}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i+\sum_{i\in Z}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right| \\
&\geq \frac{1}{\ell^{\boldsymbol{\lambda}}(X)}\left|\left|\sum_{i\in F}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right|-\left|\sum_{i\in Z}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right|\right| \text{ (using the triangle inequality)}, \\
&\geq \frac{1}{\ell^{\boldsymbol{\lambda}}(X)}\left(\left|\sum_{i\in F}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right|-\left|\sum_{i\in Z}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right|\right).
\end{aligned}$$

We can bound the first summation as

$$\left|\sum_{i\in F}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right| = \ell^{\boldsymbol{\lambda}}(F)\delta_F(\vec{e}_j;\boldsymbol{\lambda}) \geq \sqrt{C_0(\ell^{\boldsymbol{\lambda}}(F)-K)\ell^{\boldsymbol{\lambda}}(F)} \text{ (by (12))}.$$

We can bound the second summation as

$$\left|\sum_{i\in Z}\ell^{\boldsymbol{\lambda}}(i)(\mathbf{M}\vec{e}_j)_i\right| \leq \sum_{i\in Z}\ell^{\boldsymbol{\lambda}}(i)\|\mathbf{M}\|_\infty \leq \ell^{\boldsymbol{\lambda}}(Z).$$

Substituting the two bounds above,

$$\delta_X(\vec{e}_j;\boldsymbol{\lambda}) \geq \frac{1}{\ell^{\boldsymbol{\lambda}}(X)}\left(\sqrt{C_0(\ell^{\boldsymbol{\lambda}}(F)-K)\ell^{\boldsymbol{\lambda}}(F)}-\ell^{\boldsymbol{\lambda}}(Z)\right). \tag{13}$$

Now, assume as in the statement of the lemma that $\ell^{\boldsymbol{\lambda}}(Z) \leq C_1\theta$. Thus, $\ell^{\boldsymbol{\lambda}}(F)-K = \theta-\ell^{\boldsymbol{\lambda}}(Z) \geq (1-C_1)\theta$. Set $C_1 = \min\left\{1/4,(1/4)\sqrt{C_0/m}\right\}$. Then we have

$$\begin{aligned}
\sqrt{C_0(\ell^{\boldsymbol{\lambda}}(F)-K)\ell^{\boldsymbol{\lambda}}(F)} &\geq \sqrt{C_0(1-C_1)\theta} \\
\ell^{\boldsymbol{\lambda}}(Z) &\leq C_1\theta.
\end{aligned}$$

The last inequality holds because $\theta \leq m$, which follows since the total loss $K+\theta$ at any stage of boosting is less than the initial loss $m$. Using the above, we simplify (13) as follows:

$$\delta_X(\vec{e}_j;\boldsymbol{\lambda}) \geq \frac{1}{\ell^{\boldsymbol{\lambda}}(X)}\left(\sqrt{C_0(\ell^{\boldsymbol{\lambda}}(F)-K)\ell^{\boldsymbol{\lambda}}(F)}-\ell^{\boldsymbol{\lambda}}(Z)\right) \geq \frac{1}{K+\theta}\left(\sqrt{C_0(1-C_1)\theta}-C_1\theta\right)$$

Using $\theta \leq m$, we can bound the square of the term in brackets on the previous line as

$$\begin{aligned}
\left(\sqrt{C_0(1-C_1)\theta}-C_1\theta\right)^2 &\geq C_0(1-C_1)\theta-2C_1\theta\sqrt{C_0(1-C_1)\theta} \\
&\geq C_0(1-1/4)\theta-2\left((1/4)\sqrt{C_0/m}\right)\theta\sqrt{C_0(1-0)m} \\
&= C_0\theta/4.
\end{aligned}$$

So, if $\delta$ is the maximum edge in any direction, then

$$\delta \geq \delta_X(\vec{e}_j;\boldsymbol{\lambda}) \geq \sqrt{C_0\theta/(2(K+\theta)^2)} \geq \sqrt{C_0\theta/(2m(K+\theta))},$$

where, for the last inequality, we again used $K+\theta \leq m$. Therefore, from (5), the loss after one more step is at most $(K+\theta)\sqrt{1-\delta^2} \leq (K+\theta)(1-\delta^2/2) \leq K+\theta-\frac{C_0}{4m}\theta$. Setting $C_2 = C_0/(4m)$ completes the proof. ■

**Proof of Theorem 14.** At any stage of boosting, let $\boldsymbol{\lambda}$ be the current combination, and $K+\theta$ be the current loss. We show that the new loss is at most $K+\theta-\Delta\theta$ for $\Delta\theta \geq C_3\theta^2$ for some constant $C_3$ depending only on the data set (and not $\theta$). To see this, either $\ell^{\boldsymbol{\lambda}}(Z) \leq C_1\theta$, in which case Lemma 18 applies, and $\Delta\theta \geq C_2\theta \geq (C_2/m)\theta^2$ (since $\theta = \ell^{\boldsymbol{\lambda}}(X) - K \leq m$). Or $\ell^{\boldsymbol{\lambda}}(Z) > C_1\theta$, in which case applying Lemma 16 yields $\delta \geq \gamma C_1\theta/\ell^{\boldsymbol{\lambda}}(X) \geq (\gamma C_1/m)\theta$. By (5), $\Delta\theta \geq \ell^{\boldsymbol{\lambda}}(X)(1-\sqrt{1-\delta^2}) \geq \ell^{\boldsymbol{\lambda}}(X)\delta^2/2 \geq (K/2)(\gamma C_1/m)^2\theta^2$. Using $K \geq 1$ and choosing $C_3$ appropriately as $\min\left\{C_2/m, (1/2)(\gamma C_1/m)^2\right\}$ gives the required condition for $\Delta\theta \geq C_3\theta^2$. Note that plugging in the estimates for $C_1$ and $C_2$ from the proof of Lemma 18 yields

$$C_3 \geq \max\left\{\frac{C_0}{4m^2}, \frac{\gamma^2 C_0}{32m^3}\right\} = \frac{C_0}{4m^2}\max\left\{1, \frac{\gamma^2}{8m}\right\}. \tag{14}$$

If $K+\theta_t$ denotes the loss in round $t$, then the above claim implies $\theta_t - \theta_{t+1} \geq C_3\theta_t^2$. We will show that $T = C_3^{-1}/\varepsilon$ rounds suffice for the loss to be at most $\varepsilon$, that is, $\theta_T \leq \varepsilon$. Since $\theta_t$ is non-increasing and non-negative, if it is non-positive at any point before $T$ iterations, then the bound follows trivially. So assume $\theta_t$'s are positive for $t \leq T$. Applying Lemma 32 to the sequence $\{\theta_t\}$ we have $1/\theta_T - 1/\theta_0 \geq C_3 T = 1/\varepsilon$. Since $\theta_0 \geq 0$, we have $\theta_T \leq \varepsilon$, completing the proof. ■

### 4.2 Proof Of The Decomposition Lemma

Throughout this section we only consider (unless otherwise stated) *admissible* combinations $\boldsymbol{\lambda}$ of weak classifiers, which have loss $\ell^{\boldsymbol{\lambda}}(X)$ bounded by $m$ (since these are the ones found by boosting). We prove Lemma 15 in three steps. We begin with a simple lemma that rigorously defines the zero-loss and finite-margin sets.

**Lemma 19** *For any infinite sequence $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \ldots$, of admissible combinations of weak classifiers, we can find a subsequence $\boldsymbol{\eta}_{(1)} = \boldsymbol{\eta}_{t_1}, \boldsymbol{\eta}_{(2)} = \boldsymbol{\eta}_{t_2}, \ldots$, whose losses converge to zero on all examples in some fixed (possibly empty) subset $Z$ (the zero-loss set), and whose losses are bounded away from zero in its complement $X \setminus Z$(the finite-margin set)*

$$\forall x \in Z : \lim_{t\to\infty} \ell^{\boldsymbol{\eta}_{(t)}}(x) = 0, \qquad \forall x \in X \setminus Z : \inf_t \ell^{\boldsymbol{\eta}_{(t)}}(x) > 0. \tag{15}$$

**Proof** We will build a zero-loss set and the final subsequence incrementally. Initially the set is empty. Pick the first example. If the infimal loss ever attained on the example in the sequence is bounded away from zero, then we do not add it to the set. Otherwise we add it, and consider only the subsequence whose $t^{\text{th}}$ element attains loss less than $1/t$ on the example. Beginning with this subsequence, we now repeat with other examples. The final sequence is the required subsequence, and the examples we have added form the zero-loss set. ■

Lemma 19 can be applied to any sequence to yield a new sequence with respect to which the examples can be decomposed into zero-loss and finite-margin sets satisfying (15). This way we can get *nicer* sequences out of ones with possibly complicated tail behavior. The next lemma shows, that given such a nice sequence, one may extract a single vector that satisfies properties similar to those required in Item 1 of the decomposition lemma.

**Lemma 20** *Let* $\mathbf{M}$ *be a feature matrix, with rows indexed by examples in* $X$*. Suppose* $Z$ *is a subset of the examples, and* $\boldsymbol{\eta}_{(1)}, \boldsymbol{\eta}_{(2)}, \ldots$*, is a sequence of combinations of weak classifiers such that* $Z$ *is its zero loss set, and* $X \setminus Z$ *its finite margin set, that is,* (15) *holds with respect to the entire sequence itself. Then there is a combination* $\boldsymbol{\eta}^\dagger$ *of weak classifiers that achieves positive margin on every example in* $Z$*, and zero margin on every example in its complement* $X \setminus Z$*, that is:*

$$(\mathbf{M}\boldsymbol{\eta}^\dagger)_i \begin{cases} > 0 & \text{if } i \in Z, \\ = 0 & \text{if } i \in X \setminus Z. \end{cases}$$

**Proof** Firstly assume $Z$ is non-empty, since otherwise setting $\boldsymbol{\eta}^\dagger$ to the zero-vector proves the lemma. Since the $\boldsymbol{\eta}_{(t)}$ achieve arbitrarily large positive margins on $Z$, the sequence $\|\boldsymbol{\eta}_{(t)}\|$ will be unbounded, and it will be hard to extract a useful single solution out of them. On the other hand, the rescaled combinations $\boldsymbol{\eta}_{(t)}/\|\boldsymbol{\eta}_{(t)}\|$ lie on a compact set, and therefore have a limit point, which might have useful properties. We formalize this next.

We prove the statement of the lemma by induction on the total number of training examples $|X|$. To be more precise, the lemma makes an assertion about feature matrices $M$, whose rows are indexed by the set $X$. We will prove this assertion for all feature matrices $M$ by induction on the number $|X|$ of rows it contains. If $X$ is empty, then the lemma holds vacuously for any $\boldsymbol{\eta}^\dagger$. Assume the statement of the lemma holds inductively for all subsets of $X$ of size less than $m > 0$, meaning that a vector analogous to $\boldsymbol{\eta}^\dagger$ exists for the subset, and consider $X$ of size $m$.

First, we find a unit vector $\boldsymbol{\eta}'$ that we will show has a positive margin on a non-empty subset $S$ of $Z$ and zero margins on $X/Z$. Since translating a vector along the null space of $\mathbf{M}$, $\ker\mathbf{M} = \{\vec{x} : \mathbf{M}\vec{x} = \mathbf{0}\}$, has no effect on the margins produced by the vector, assume without loss of generality that the $\boldsymbol{\eta}_{(t)}$'s are orthogonal to $\ker\mathbf{M}$. Since the margins produced on $Z$, which we have assumed is non-empty, are unbounded, so are the norms of $\boldsymbol{\eta}_{(t)}$. Therefore assume (by picking a subsequence and relabeling if necessary) that $\|\boldsymbol{\eta}_{(t)}\| > t$. Let $\boldsymbol{\eta}'$ be a limit point of the sequence $\boldsymbol{\eta}_{(t)}/\|\boldsymbol{\eta}_{(t)}\|$, a unit vector that is also orthogonal to the null-space. Then firstly $\boldsymbol{\eta}'$ achieves non-negative margin on every example; otherwise by continuity for some extremely large $t$, the margin of $\boldsymbol{\eta}_{(t)}/\|\boldsymbol{\eta}_{(t)}\|$ on that example is also negative and bounded away from zero, and therefore $\boldsymbol{\eta}_{(t)}$'s loss is more than $m$, which is a contradiction to admissibility. Secondly, the margin of $\boldsymbol{\eta}'$ on each example in $X \setminus Z$ is zero; otherwise, by continuity, for arbitrarily large $t$ the margin of $\boldsymbol{\eta}_{(t)}/\|\boldsymbol{\eta}_{(t)}\|$ on an example in $X \setminus Z$ is positive and bounded away from zero, and hence that example attains arbitrarily small loss in the sequence, a contradiction to (15). Finally, if $\boldsymbol{\eta}'$ achieves zero margin everywhere in $Z$, then $\boldsymbol{\eta}'$, being orthogonal to the null-space, must be $\mathbf{0}$, a contradiction since $\boldsymbol{\eta}'$ is a unit vector. Therefore $\boldsymbol{\eta}'$ must achieve positive margin on some non-empty subset $S$ of $Z$, and zero margins on every other example.

Next we use induction on the feature matrix restricted to the reduced set of examples $X' = X \setminus S$. Since $S$ is non-empty, $|X'| < m$. Further, using the same sequence $\boldsymbol{\eta}_{(t)}$, the zero-loss and finite-loss sets, restricted to $X'$, are $Z' = Z \setminus S$ and $(X \setminus Z) \setminus S = X \setminus Z$ (since $S \subseteq Z$) $= X' \setminus Z'$. The set $X'$ is smaller than the set $X$, and thus the inductive hypothesis holds for $X'$, meaning that there exists some $\boldsymbol{\eta}''$ that achieves positive margins on every element in $Z'$, and zero margins on every element in $X' \setminus Z' = X \setminus Z$. Therefore, by setting $\boldsymbol{\eta}^\dagger = \boldsymbol{\eta}' + c\boldsymbol{\eta}''$ for a suitable $c$, we can achieve a positive margin on every element in $S \cup Z' = Z$, and zero margins on every element in $X \setminus Z$, completing the proof. ■

We may now use the previous two results to prove Item 1 of the decomposition lemma. First, we

apply Lemma 19 to some admissible sequence converging to the optimal loss (for instance, the one found by AdaBoost). Let us call the resulting subsequence $\boldsymbol{\eta}^*_{(t)}$, the obtained zero-loss set $Z$, and the finite-margin set $F = X \setminus Z$. Now, applying Lemma 20 to the sequence $\boldsymbol{\eta}^*_{(t)}$ yields some convex combination $\boldsymbol{\eta}^\dagger$ having margin at least $\gamma > 0$ (for some $\gamma$) on $Z$ and zero margin on its complement, proving Item 1 of the decomposition lemma. The next lemma proves Item 2.

**Lemma 21** *The optimal loss considering only examples within $F$ is achieved by some finite combination $\boldsymbol{\eta}^*$.*

**Proof** The existence of $\boldsymbol{\eta}^\dagger$ with properties as in Lemma 20 implies that the optimal loss is the same whether considering all the examples, or just examples in $F$. Therefore it suffices to show the existence of finite $\boldsymbol{\eta}^*$ that achieves loss $K$ on $F$, that is, $\ell^{\boldsymbol{\eta}^*}(F) = K$.

Recall $\mathbf{M}_F$ denotes the matrix $\mathbf{M}$ restricted to the rows corresponding to examples in $F$. Let $\ker \mathbf{M}_F = \{\vec{x} : \mathbf{M}_F \vec{x} = 0\}$ be the null-space of $\mathbf{M}_F$. Let $\boldsymbol{\eta}^{(t)}$ be the projection of $\boldsymbol{\eta}^*_{(t)}$ onto the orthogonal subspace of $\ker \mathbf{M}_F$. Then the losses $\ell^{\boldsymbol{\eta}^{(t)}}(F) = \ell^{\boldsymbol{\eta}^*_{(t)}}(F)$ converge to the optimal loss $K$. If $\mathbf{M}_F$ is identically zero, then each $\boldsymbol{\eta}^{(t)} = \mathbf{0}$, and then $\boldsymbol{\eta}^* = \mathbf{0}$ has loss $K = |F|$ on $F$. Otherwise, let $\lambda^2$ be the smallest positive eigenvalue of the symmetric matrix $\mathbf{M}_F^T \mathbf{M}_F$. Then $\|\mathbf{M}\boldsymbol{\eta}^{(t)}\| \geq \lambda \|\boldsymbol{\eta}^{(t)}\|$. By the definition of the finite margin set, $\inf_t \min_{i \in F} \ell^{\boldsymbol{\eta}^{(t)}}(i) = \inf_t \min_{i \in F} \ell^{\boldsymbol{\eta}^*_{(t)}}(i) > 0$. Therefore, the norms of the margin vectors $\|\mathbf{M}\boldsymbol{\eta}^{(t)}\|$, and hence that of $\boldsymbol{\eta}^{(t)}$, are bounded. Therefore the $\boldsymbol{\eta}^{(t)}$'s have a (finite) limit point $\boldsymbol{\eta}^*$ that must have loss $K$ over $F$. ■

As a corollary, we prove Item 3.

**Lemma 22** *There is a constant $\mu_{\max} < \infty$, such that for any combination $\boldsymbol{\eta}$ that achieves bounded loss on the finite-margin set, $\ell^{\boldsymbol{\eta}}(F) \leq m$, the margin $(\mathbf{M}\boldsymbol{\eta})_i$ for any example $i$ in $F$ lies in the bounded interval $[-\ln m, \mu_{\max}]$.*

**Proof** Since the loss $\ell^{\boldsymbol{\eta}}(F)$ is at most $m$, therefore no margin may be less than $-\ln m$. To prove a finite upper bound on the margins, we argue by contradiction. Suppose arbitrarily large margins are producible by bounded loss vectors, that is arbitrarily large elements are present in the set $\{(\mathbf{M}\boldsymbol{\eta})_i : \ell^{\boldsymbol{\eta}}(F) \leq m, 1 \leq i \leq m\}$. Then for some fixed example $x \in F$ there exists a sequence of combinations of weak classifiers, whose $t^{\text{th}}$ element achieves more than margin $t$ on $x$ but has loss at most $m$ on $F$. Applying Lemma 19 we can find a subsequence $\boldsymbol{\lambda}^{(t)}$ whose tail achieves vanishingly small loss on some non-empty subset $S$ of $F$ containing $x$, and bounded margins in $F \setminus S$. Applying Lemma 20 to $\boldsymbol{\lambda}^{(t)}$ we get some convex combination $\boldsymbol{\lambda}^\dagger$ which has positive margins on $S$ and zero margin on $F \setminus S$. Let $\boldsymbol{\eta}^*$ be as in Lemma 21, a finite combination achieving the optimal loss on $F$. Then $\boldsymbol{\eta}^* + \infty \cdot \boldsymbol{\lambda}^\dagger$ achieves the same loss on every example in $F \setminus S$ as the optimal solution $\boldsymbol{\eta}^*$, but zero loss for examples in $S$. This solution is strictly better than $\boldsymbol{\eta}^*$ on $F$, a contradiction to the optimality of $\boldsymbol{\eta}^*$. Therefore our assumption is false, and some finite upper bound $\mu_{\max}$ on the margins $(\mathbf{M}\boldsymbol{\eta})_i$ of vectors satisfying $\ell^{\boldsymbol{\eta}}(F) \leq m$ exists. ■

The proof of the decomposition theorem is complete.

### 4.3 Investigating The Constants

In this section, we try to estimate the constant $C$ in Theorem 14. We show that it can be arbitrarily large for adversarial feature matrices with real entries (corresponding to confidence rated weak hypotheses), but has an upper-bound doubly exponential in the number of examples when the feature

matrix has $\{-1,0,+1\}$ entries only. We also show that this doubly exponential bound cannot be improved without significantly changing the proof in the previous section.

By inspecting the proofs, in particular equations (11) and (14), and seeing that $1/C_3$ is $C$ from Theorem 14, we can bound the constant in Theorem 14 as follows.

**Corollary 23** *The constant $C$ in Theorem 14 that emerges from the proofs is*

$$C \leq \frac{2m^3 N e^{\mu_{\max}}}{\lambda_{\min}^2} \max\left\{1, \frac{8m}{\gamma^2}\right\},$$

*where $m$ is the number of examples, $N$ is the number of hypotheses, $\gamma$ and $\mu_{\max}$ are as given by Items 1 and 3 of the decomposition lemma, and $\lambda_{\min}^2$ is the smallest positive eigenvalue of $\mathbf{M}_F^T \mathbf{M}_F$ ($\mathbf{M}_F$ is the feature matrix restricted to the rows belonging to the finite margin set $F$).*

Our bound on $C$ will be obtained by in turn bounding the quantities $\lambda_{\min}^{-1}, \gamma^{-1}, \mu_{\max}$. These are strongly related to the singular values of the feature matrix $\mathbf{M}$, and in general cannot be easily measured. In fact, when $\mathbf{M}$ has real entries, we have already seen in Section 3.3 that the rate can be arbitrarily large, implying these parameters can have very large values. Even when the matrix $\mathbf{M}$ has integer entries (that is, $-1,0,+1$), the next lemma shows that these quantities can be exponential in the number of examples.

**Lemma 24** *There are examples of feature matrices with $-1,0,+1$ entries and at most $m$ rows or columns (where $m > 10$) for which the quantities $\gamma^{-1}, \lambda^{-1}$ and $\mu_{\max}$ are at least $\Omega(2^m/m)$.*

**Proof** We first show the bounds for $\gamma$ and $\lambda$. Let $\mathbf{M}$ be an $m \times m$ upper triangular matrix with $+1$ on the diagonal, and $-1$ above the diagonal. Let $\vec{y} = (2^{m-1}, 2^{m-2}, \ldots, 1)^T$, and $\mathbf{b} = (1,1,\ldots,1)^T$. Then $\mathbf{M}\vec{y} = \mathbf{b}$, although the $\vec{y}$ has much bigger norm than $\mathbf{b}$: $\|\vec{y}\| \geq 2^{m-1}$, while $\|\mathbf{b}\| = m$. Since $\mathbf{M}$ is invertible, by the definition of $\lambda_{\min}$, we have $\|\mathbf{M}\vec{y}\| \geq \lambda_{\min}\|\vec{y}\|$, so that $\lambda_{\min}^{-1} \geq \|\vec{y}\|/\|\mathbf{M}\vec{y}\| \geq 2^m/m$. Next, note that $\vec{y}$ produces all positive margins $\mathbf{b}$, and hence the zero-loss set consists of all the examples. In particular, if $\boldsymbol{\eta}^\dagger$ is as in Item 1 of the decomposition lemma, then the vector $\gamma^{-1}\boldsymbol{\eta}^\dagger$ achieves margin greater than 1 on each example: $\mathbf{M}(\gamma^{-1}\boldsymbol{\eta}^\dagger) \geq \mathbf{b}$. On the other hand, our matrix is very similar to the one in Theorem 10, and the same arguments in the proof of that theorem can be used to show that if for some $\vec{x}$ we have $(\mathbf{M}\vec{x}) \geq \mathbf{b}$ entry-wise, then $\vec{x} \geq \vec{y}$. This implies that $\gamma^{-1}\|\boldsymbol{\eta}^\dagger\|_1 \geq \|\vec{y}\|_1 = (2^m - 1)$. Since $\boldsymbol{\eta}^\dagger$ has unit $\ell_1$-norm, the bound on $\gamma^{-1}$ follows too.

Next we provide an example showing $\mu_{\max}$ can be $\Omega(2^m/m)$. Consider an $m \times (m-1)$ matrix $\mathbf{M}$. The bottom row of $\mathbf{M}$ is all $+1$. The upper $(m-1)\times(m-1)$ submatrix of $\mathbf{M}$ is a lower triangular matrix with $-1$ on the diagonal and $+1$ below the diagonal. Observe that if $\vec{y}^T = (2^{m-2}, 2^{m-3}, \ldots, 1, 1)$, then $\vec{y}^T\mathbf{M} = \mathbf{0}$. Therefore, for any vector $\vec{x}$, the inner product of the margins $\mathbf{M}\vec{x}$ with $\vec{y}$ is zero: $\vec{y}^T\mathbf{M}\vec{x} = 0$. This implies that achieving positive margin on any example forces some other example to receive negative margin. By Item 1 of the decomposition lemma, the zero loss set in this data set is empty since there cannot be an $\boldsymbol{\eta}^\dagger$ with both positive and zero margins and no negative margins. Thus, all the examples belong to the finite margin set. Next, we choose a combination with at most $m$ loss that nevertheless achieves $\Omega(2^m/m)$ positive margin on some example. Let $\vec{x}^T = (1,2,4,\ldots,2^{m-2})$. Then $(\mathbf{M}\vec{x})^T = (-1,-1,\ldots,-1,2^{m-1}-1)$. Then the margins using $\varepsilon\vec{x}$ are $(-\varepsilon,\ldots,-\varepsilon,\varepsilon(2^{m-1}-1))$ with total loss $(m-1)e^\varepsilon + e^{\varepsilon(1-2^{m-1})}$. Choose $\varepsilon = 1/(2m) \leq 1$, so that the loss on examples corresponding to the first $m-1$ rows is at most $e^\varepsilon \leq 1 + 2\varepsilon = 1 + 1/m$, where the first inequality holds since $\varepsilon \in [0,1]$. For $m > 10$, the choice

of $\varepsilon$ guarantees $1/(2m) = \varepsilon \geq (\ln m)/(2^{m-1}-1)$, so that the loss on the example corresponding to the bottom most row is $e^{-\varepsilon(2^{m-1}-1)} \leq e^{-\ln m} = 1/m$. Therefore the net loss of $\varepsilon\vec{x}$ is at most $(m-1)(1+1/m)+1/m = m$. On the other hand the margin of the example corresponding to the last row is $\varepsilon(2^{m-1}-1) = (2^{m-1}-1)/(2m) = \Omega(2^m/m)$. ■

The above result implies any bound on $C$ derived from Corollary 23 will be at least $2^{\Omega(2^m/m)}$ in the worst case. This does not imply that the best bound one can hope to prove is doubly exponential, only that our techniques in the previous section do not admit anything better. We next show that the bounds in Lemma 24 are nearly the worst possible.

**Lemma 25** *Suppose each entry of* $\mathbf{M}$ *is* $-1, 0$ *or* $+1$*. Then each of the quantities* $\lambda_{\min}^{-1}, \gamma^{-1}$ *and* $\mu_{\max}$ *are at most* $2^{O(m \ln m)}$.

The proof of Lemma 25 is rather technical, and we defer it to the Appendix. Lemma 25 and Corollary 23 together imply a convergence rate of $2^{2^{O(m \ln m)}}/\varepsilon$ to the optimal loss for integer matrices. This bound on $C$ is exponentially worse than the $\Omega(2^m)$ lower bound on $C$ we saw in Section 3.3, a price we pay for obtaining optimal dependence on $\varepsilon$. In the next section we will see how to obtain $\text{poly}(2^{m \ln m}, \varepsilon^{-1})$ bounds, although with a worse dependence on $\varepsilon$. We end this section by showing, just for completeness, how a bound on the norm of $\boldsymbol{\eta}^*$ as defined in Item 2 of the decomposition lemma follows as a quick corollary to Lemma 25.

**Corollary 26** *Suppose* $\boldsymbol{\eta}^*$ *is as given by Item 2 of the decomposition lemma. When the feature matrix has only* $-1, 0, +1$ *entries, we may bound* $\|\boldsymbol{\eta}^*\|_1 \leq 2^{O(m \ln m)}$.

**Proof** Note that every entry of $\mathbf{M}_F \boldsymbol{\eta}^*$ lies in the range $[-\ln m, \mu_{\max} = 2^{O(m \ln m)}]$, and hence $\|\mathbf{M}_F \boldsymbol{\eta}^*\| \leq 2^{O(m \ln m)}$. Next, we may choose $\boldsymbol{\eta}^*$ orthogonal to the null space of $\mathbf{M}_F$; then $\|\boldsymbol{\eta}^*\| \leq \lambda_{\min}^{-1} \|\mathbf{M}_F \boldsymbol{\eta}^*\| \leq 2^{O(m \ln m)}$. Since $\|\boldsymbol{\eta}^*\|_1 \leq \sqrt{N}\|\boldsymbol{\eta}^*\|$, and the number of possible columns $N$ with $\{-1, 0, +1\}$ entries is at most $3^m$, the proof follows. ■

## 5. Improved Estimates

The goal of this section is to show how the ideas introduced in the paper can be applied in ways other than presented so far to produce new and stronger results. By combining techniques from Sections 3 and 4, we obtain both new upper bounds for convergence to the optimal loss, as well as more general lower bounds for convergence to an arbitrary target loss. We also indicate what we believe might be the optimal bounds for either situation.

We first show how the finite rate bound of Theorem 1 along with the decomposition lemma yields a new rate of convergence to the optimal loss. The proof includes choosing a useful target $\boldsymbol{\lambda}^*$ for Theorem 1. Although the dependence on $\varepsilon$ is worse than in Theorem 14, the dependence on $m$ is nearly optimal. We will need the following key application of the decomposition lemma.

**Lemma 27** *When the feature matrix has* $-1, 0, +1$ *entries, for any* $\varepsilon > 0$*, there is some solution with* $\ell_1$*-norm at most* $2^{O(m \ln m)} \ln(1/\varepsilon)$ *that achieves within* $\varepsilon$ *of the optimal loss.*

**Proof** Let $\boldsymbol{\eta}^*, \boldsymbol{\eta}^\dagger, \gamma$ be as given by the decomposition lemma. Let $c = \min_{i \in Z} (\mathbf{M}\boldsymbol{\eta}^*)_i$ be the minimum margin produced by $\boldsymbol{\eta}^*$ on any example in the zero-loss set $Z$. Then $\boldsymbol{\eta}^* - c\boldsymbol{\eta}^\dagger$ produces non-negative margins on $Z$, since $\mathbf{M}\boldsymbol{\eta}^* - c\mathbf{M}\boldsymbol{\eta}^\dagger \geq \mathbf{0}$, and it attains the optimal margins on the finite

margin set $F$, since $\mathbf{M}\boldsymbol{\eta}^\dagger = \mathbf{0}$ on $F$. Therefore, the vector $\boldsymbol{\lambda}^* = \boldsymbol{\eta}^* + \left(\ln(1/\varepsilon)\gamma^{-1} - c\right)\boldsymbol{\eta}^\dagger$ achieves at least $\ln(1/\varepsilon)$ margin on every example in $Z$, and optimal margins on the finite loss set $F$. Hence $L(\boldsymbol{\lambda}^*) \leq \inf_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda}) + \varepsilon$. Using $|c| \leq \|\mathbf{M}\boldsymbol{\eta}^*\| \leq m\|\boldsymbol{\eta}^*\|$,

$$\begin{aligned}\|\boldsymbol{\lambda}^*\|_1 &\leq \|\boldsymbol{\eta}^*\| + \ln(1/\varepsilon)\gamma^{-1}\|\boldsymbol{\eta}^\dagger\| + |c|\|\boldsymbol{\eta}^\dagger\| \\ &\leq \|\boldsymbol{\eta}^*\| + \ln(1/\varepsilon)\gamma^{-1} \cdot 1 + m\|\boldsymbol{\eta}^*\| \cdot 1.\end{aligned}$$

Combining this with the results in Corollary 26 and Lemma 25, we may conclude the vector $\boldsymbol{\lambda}^*$ has $\ell_1$-norm at most $2^{O(m\ln m)}\ln(1/\varepsilon)$. ■

We may now invoke Theorem 1 to obtain a $2^{O(m\ln m)}\ln^6(1/\varepsilon)\varepsilon^{-5}$ rate of convergence to the optimal solution. Rate bounds with similar dependence on $m$ and slightly better dependence on $\varepsilon$ can be obtained by modifying the proof in Section 4 to use first order instead of second order techniques. In that way we may obtain a $\mathrm{poly}(\lambda_{\min}^{-1}, \gamma^{-1}, \mu_{\max})\varepsilon^{-3} = 2^{O(m\ln m)}\varepsilon^{-3}$ rate bound. We omit the rather long but straightforward proof of this fact. Finally, note that if Conjecture 6 is true, then Lemma 27 provides a bound for $B$ in Conjecture 6, implying a $2^{O(m\ln m)}\ln(1/\varepsilon)\varepsilon^{-1}$ rate bound for converging to the optimal loss, which is nearly optimal in both $m$ and $\varepsilon$. We state this as an independent conjecture.

**Conjecture 28** *For feature matrices with $-1, 0, +1$ entries, AdaBoost converges to within $\varepsilon$ of the optimal loss within $2^{O(m\ln m)}\varepsilon^{-(1+o(1))}$ rounds.*

We next focus on lower bounds on the convergence rate to arbitrary target losses discussed in Section 3. We begin by showing the rate dependence on the norm of the solution as given in Lemma 9 holds for much more general data sets.

**Lemma 29** *Suppose a feature matrix has only $\pm 1$ entries, and the finite loss set is non-empty. Consider any coordinate descent procedure, that iteratively chooses a sequence of vectors $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^t, \ldots$, such that successive elements $\boldsymbol{\lambda}^t$ and $\boldsymbol{\lambda}^{t+1}$ of this sequence differ in at most one coordinate, and the loss on this sequence is non-increasing. Then, the number of rounds required by such a procedure to achieve a target loss $\phi^*$ is at least*

$$\inf\{\|\boldsymbol{\lambda}\|_1 : L(\boldsymbol{\lambda}) \leq \phi^*\}/(2 + 2\ln m).$$

**Proof** It suffices to upper-bound the step size $|\alpha_t|$ in any round $t$ by at most $2 + 2\ln m$; as long as the step size is smaller than or equal to $2 + 2\ln m$, it will take at least $\inf\{\|\boldsymbol{\lambda}\|_1 : L(\boldsymbol{\lambda}) \leq \phi^*\}/(2 + 2\ln m)$ steps for $\|\boldsymbol{\lambda}\|_1$ to be at least $\inf\{\|\boldsymbol{\lambda}\|_1 : L(\boldsymbol{\lambda}) \leq \phi^*\}$. To see this, recall that (3) shows that a step $\alpha_t$ causes the loss to change by a factor of $f(\alpha_t)$ given by:

$$f(\alpha_t) = \frac{(1 + r_t)}{2}e^{-\alpha_t} + \frac{(1 - r_t)}{2}e^{\alpha_t},$$

where $r_t$ denotes the correlation in direction $j_t$ in which the step is taken. We find the maximum magnitude of $\alpha_t$ that will still allow $f(\alpha_t)$ to be at most 1. Notice that $f''(\alpha_t) = f(\alpha_t)$ is always positive, since $|r_t| = \delta_t \leq 1$. Therefore $f$ is strictly convex, and $f(\alpha_t) \leq 1$ for $\alpha_t$ lying in some interval. Since $f(\alpha_t) = 1$ at $\alpha_t = 0$ and $\alpha_t = \ln\left((1 + r_t)/(1 - r_t)\right)$, the desired maximum magnitude is the latter value. Therefore,

$$|\alpha_t| \leq \left|\ln\left(\frac{1 + r_t}{1 - r_t}\right)\right| = \ln\left(\frac{1 + |r_t|}{1 - |r_t|}\right) = \ln\left(\frac{1 + \delta_t}{1 - \delta_t}\right).$$

Further, by (5), we have that the optimal step in direction $j_t$ would cause the loss to change as follows: $L(\boldsymbol{\lambda}^t) \leq L(\boldsymbol{\lambda}^{t-1})\sqrt{1-\delta_t^2}$. On the other hand, before the step, the loss is at most 1, $L(\boldsymbol{\lambda}^{t-1}) \leq 1$, and after the step the loss is at least $1/m$, that is, $L(\boldsymbol{\lambda}^t) \geq 1/m$. This last fact comes from having at least one example with at most margin 0. Combining these inequalities we get

$$1/m \leq L(\boldsymbol{\lambda}^t) \leq L(\boldsymbol{\lambda}^{t-1})\sqrt{1-\delta_t^2} \leq \sqrt{1-\delta_t^2},$$

that is, $\sqrt{1-\delta_t^2} \geq 1/m$. Now the step length can be bounded as

$$|\alpha_t| \leq \ln\left(\frac{1+\delta_t}{1-\delta_t}\right) = 2\ln(1+\delta_t) - \ln(1-\delta_t^2) \leq 2\delta_t + 2\ln m \leq 2 + 2\ln m.$$

■

We end by showing a new lower bound for the convergence rate to an arbitrary target loss studied in Section 3. Corollary 11 implies that the rate bound in Theorem 1 has to be at least polynomially large in the norm of the solution. We now show that a polynomial dependence on $\varepsilon^{-1}$ in the rate is unavoidable too. This shows that rates for competing with a finite solution are different from rates on a data set where the optimum loss is achieved by a finite solution, since in the latter we may achieve a $O(\ln(1/\varepsilon))$ rate.

**Corollary 30** *Consider any data set (e.g., the one in Figure 4) for which $\Omega(1/\varepsilon)$ rounds are necessary to get within $\varepsilon$ of the optimal loss. If there are constants $c$ and $\beta$ such that for any $\boldsymbol{\lambda}^*$ and $\varepsilon$, a loss of $L(\boldsymbol{\lambda}^*)+\varepsilon$ can be achieved in at most $O(\|\boldsymbol{\lambda}^*\|_1^c \varepsilon^{-\beta})$ rounds, then $\beta \geq 1$.*

**Proof** The decomposition lemma implies that $\boldsymbol{\lambda}^* = \boldsymbol{\eta}^* + \ln(2/\varepsilon)\boldsymbol{\eta}^\dagger$ with $\ell_1$-norm $O(\ln(1/\varepsilon))$ achieves loss at most $K + \varepsilon/2$ (recall $K$ is the optimal loss). Suppose the corollary fails to hold for constants $c$ and $\beta \leq 1$. Then $L(\boldsymbol{\lambda}^*)+\varepsilon/2 = K+\varepsilon$ loss can be achieved in $O(\varepsilon^{-\beta}/\ln^c(1/\varepsilon)) = o(1/\varepsilon)$ rounds, contradicting the $\Omega(1/\varepsilon)$ lower bound in Lemma 31 in the appendix. ■

## 6. Conclusion

In this paper we studied the convergence rate of AdaBoost with respect to the exponential loss. We showed upper and lower bounds for convergence rates to both an arbitrary target loss achieved by some finite combination of the weak hypotheses, as well as to the infimum loss which may not be realizable. For the first convergence rate, we showed a strong relationship exists between the size of the minimum vector achieving a target loss and the number of rounds of coordinate descent required to achieve that loss. In particular, we showed that a polynomial dependence of the rate on the $\ell_1$-norm $B$ of the minimum size solution is absolutely necessary, and that a $\text{poly}(B, 1/\varepsilon)$ upper bound holds, where $\varepsilon$ is the accuracy parameter. The actual rate we derived has rather large exponents, and we discussed a minor variant of AdaBoost that achieves a much tighter and near optimal rate.

For the second kind of convergence, using entirely separate techniques, we derived a $C/\varepsilon$ upper bound, and showed that this is tight up to constant factors. In the process, we showed a certain decomposition lemma that might be of independent interest. We also studied the constants and showed how they depend on certain intrinsic parameters related to the singular values of the feature matrix. We estimated the worst case values of these parameters, and when considering feature

<table>
<tr><td>Convergence rate with respect to:</td><td>Reference solution (Section 3)</td><td>Optimal solution (Section 4)</td></tr>
<tr><td rowspan="2">Upper bounds:</td><td rowspan="2">$13B^6/\varepsilon^5$<br>[Theorem 1]</td><td>$\mathrm{poly}(e^{\mu_{\max}}, \lambda_{\min}^{-1}, \gamma^{-1})/\varepsilon \leq 2^{2^{O(m \ln m)}}/\varepsilon$<br>[Corollary 23, Lemma 25]</td></tr>
<tr><td>$\mathrm{poly}(\mu_{\max}, \lambda_{\min}^{-1}, \gamma^{-1})/\varepsilon^3 \leq 2^{O(m \ln m)}/\varepsilon^3$<br>[Section 5]</td></tr>
<tr><td rowspan="2">Lower bounds with:<br><br>a) $\{0, \pm 1\}$ entries</td><td>$(B/\varepsilon)^{1-\nu}$ for any $\nu > 0$<br>[Corollaries 11 and 30]</td><td rowspan="2">$\max\left\{\frac{2^m \ln(1/\varepsilon)}{\ln m}, \frac{2}{9\varepsilon}\right\}$<br>[Theorem 1, Lemma 31]</td></tr>
<tr><td>$O(2^m/\ln m)\ln(1/\varepsilon)$<br>[Theorem 10]</td></tr>
<tr><td>b) real entries</td><td colspan="2">Can be arbitrarily large even when $m, N, \varepsilon$ are held fixed [Corollary 13]</td></tr>
<tr><td>Conjectured upper bounds:</td><td>$O(B^2/\varepsilon)$ [Conjecture 6]</td><td>$2^{O(m \ln m)}/\varepsilon^{1+o(1)}$, if entries in $\{0, \pm 1\}$<br>[Conjecture 28]</td></tr>
</table>

Figure 5: Summary of our most important results and conjectures regarding the convergence rate of AdaBoost. Here $m$ refers to the number of training examples, and $\varepsilon$ is the accuracy parameter. The quantity $B$ is the $\ell_1$-norm of the reference solution used in Section 3. The parameters $\lambda_{\min}$, $\gamma$ and $\mu_{\max}$ depend on the data set and are defined and studied in Section 4.

matrices with only $\{-1, 0, +1\}$ entries, this led to a bound on the rate constant $C$ that is doubly exponential in the number of training examples. Since this is rather large, we also included bounds polynomial in both the number of training examples and the accuracy parameter $\varepsilon$, although the dependence on $\varepsilon$ in these bounds is non-optimal.

Finally, for each kind of convergence, we conjectured tighter bounds that are not known to hold presently. A table containing a summary of the results in this paper is included in Figure 5.

## Acknowledgments

This research was funded by the National Science Foundation under grants IIS-1016029 and IIS-1053407. We thank Nikhil Srivastava for informing us of the matrix used in Theorem 10. We also thank Aditya Bhaskara and Matus Telgarsky for many helpful discussions.

## Appendix A. Proofs

We provide proofs for results that have appeared before.

### A.1 Lower Bound For Convergence To Optimal Loss

**Lemma 31** *For any $\varepsilon < 1/3$, to get within $\varepsilon$ of the optimum loss on the data set in Table 4, AdaBoost takes at least $2/(9\varepsilon)$ steps.*

**Proof** Note that the optimal loss is $2/3$, and we are bounding the number of rounds necessary to get within $(2/3)+\varepsilon$ loss for $\varepsilon < 1/3$. We will compute the edge in each round analytically. Let $w_a^t, w_b^t, w_c^t$ denote the normalized-losses (adding up to 1) or weights on examples $a, b, c$ at the beginning of round $t$, $h_t$ the weak hypothesis chosen in round $t$, and $\delta_t$ the edge in round $t$. The values of these parameters are shown below for the first 5 rounds, where we have assumed (without loss of generality) that the hypothesis picked in round 1 is $\hbar_b$:

| Round | $w_a^t$ | $w_b^t$ | $w_c^t$ | $h_t$ | $\delta_t$ |
|---|---|---|---|---|---|
| $t=1:$ | $1/3$ | $1/3$ | $1/3$ | $\hbar_b$ | $1/3$ |
| $t=2:$ | $1/2$ | $1/4$ | $1/4$ | $\hbar_a$ | $1/2$ |
| $t=3:$ | $1/3$ | $1/2$ | $1/6$ | $\hbar_b$ | $1/3$ |
| $t=4:$ | $1/2$ | $3/8$ | $1/8$ | $\hbar_a$ | $1/4$ |
| $t=5:$ | $2/5$ | $1/2$ | $1/10$ | $\hbar_b$ | $1/5.$ |

Based on the patterns above, we first claim that for rounds $t \geq 2$, the edge achieved is $1/t$. In fact we prove the stronger claims, that for rounds $t \geq 2$, the following hold:

1. One of $w_a^t$ and $w_b^t$ is $1/2$.
2. $\delta_{t+1} = \delta_t/(1+\delta_t)$.

Since $\delta_2 = 1/2$, the recurrence on $\delta_t$ would immediately imply $\delta_t = 1/t$ for $t \geq 2$. We prove the stronger claims by induction on the round $t$. The base case for $t = 2$ is shown above and may be verified. Suppose the inductive assumption holds for $t$. Assume without loss of generality that $1/2 = w_a^t > w_b^t > w_c^t$; note this implies $w_b^t = 1-(w_a^t + w_c^t) = 1/2 - w_c^t$. Further, in this round, $\hbar_a$ gets picked, and has edge $\delta_t = w_a^t + w_c^t - w_b^t = 2w_c^t$. Now for any data set, the weights of the examples labeled correctly and incorrectly in a round of AdaBoost are rescaled during the weight update step in a way such that each add up to $1/2$ after the rescaling. Therefore, $w_b^{t+1} = 1/2, w_c^{t+1} = w_c^t\left(\frac{1/2}{w_a^t+w_c^t}\right) = w_c^t/(1+2w_c^t)$. Hence, $\hbar_b$ gets picked in round $t+1$ and, as before, we get edge $\delta_{t+1} = 2w_c^{t+1} = 2w_c^t/(1+2w_c^t) = \delta_t/(1+\delta_t)$. The proof of our claim follows by induction.

Next we find the loss after each iteration. Using $\delta_1 = 1/3$ and $\delta_t = 1/t$ for $t \geq 2$, the loss after $T$ rounds can be written as

$$\prod_{t=1}^{T}\sqrt{1-\delta_t^2} = \sqrt{1-(1/3)^2}\prod_{t=2}^{T}\sqrt{1-1/t^2} = \frac{2\sqrt{2}}{3}\sqrt{\prod_{t=2}^{T}\left(\frac{t-1}{t}\right)\left(\frac{t+1}{t}\right)}.$$

The product can be rewritten as follows:

$$\prod_{t=2}^{T}\left(\frac{t-1}{t}\right)\left(\frac{t+1}{t}\right) = \left(\prod_{t=2}^{T}\frac{t-1}{t}\right)\left(\prod_{t=2}^{T}\frac{t+1}{t}\right) = \left(\prod_{t=2}^{T}\frac{t-1}{t}\right)\left(\prod_{t=3}^{T+1}\frac{t}{t-1}\right).$$

Notice almost all the terms cancel, except for the first term of the first product, and the last term of the second product. Therefore, the loss after $T$ rounds is

$$\frac{2\sqrt{2}}{3}\sqrt{\left(\frac{1}{2}\right)\left(\frac{T+1}{T}\right)} = \frac{2}{3}\sqrt{1+\frac{1}{T}} \geq \frac{2}{3}\left(1+\frac{1}{3T}\right) = \frac{2}{3}+\frac{2}{9T},$$

where the inequality holds for $T \geq 1$. Since the initial error is $1 = (2/3)+1/3$, therefore, for any $\varepsilon < 1/3$, the number of rounds needed to achieve loss $(2/3)+\varepsilon$ is at least $2/(9\varepsilon)$. ■

### A.2 A Useful Technical Result

Here we prove a technical result that was used for proving the various rate upper bounds.

**Lemma 32** *Suppose $u_0, u_1, \ldots,$ are positive numbers satisfying*

$$u_t - u_{t+1} \geq c_0 u_t^{1+p}, \tag{16}$$

*for some non-negative constants $c_0, p$. Then, for any $t$,*

$$\frac{1}{u_t^p} - \frac{1}{u_0^p} \geq p c_0 t.$$

**Proof** By induction on $t$. The base case $t = 0$ is an identity. Assume the statement holds at iteration $t$. Then, by the inductive hypothesis,

$$\frac{1}{u_{t+1}^p} - \frac{1}{u_0^p} = \left(\frac{1}{u_{t+1}^p} - \frac{1}{u_t^p}\right) + \left(\frac{1}{u_t^p} - \frac{1}{u_0^p}\right) \geq \frac{1}{u_{t+1}^p} - \frac{1}{u_t^p} + p c_0 t.$$

Thus it suffices to show $1/u_{t+1}^p - 1/u_t^p \geq p c_0$. Multiplying both sides by $u_t^p$ and adding 1, this is equivalent to showing $(u_t/u_{t+1})^p \geq 1 + p c_0 u_t^p$. We will in fact show the stronger inequality

$$(u_t/u_{t+1})^p \geq \exp\left(c_0 u_t^p\right)^p = \exp\left(p c_0 u_t^p\right). \tag{17}$$

Because of the exponential inequality, $e^x \geq 1 + x$, (17) will imply $(u_t/u_{t+1})^p \geq \exp\left(p c_0 u_t^p\right) \geq 1 + p c_0 u_t^p$, which will complete our proof. To show (17), we first rearrange the condition (16) on $u_t, u_{t+1}$ to obtain

$$u_{t+1} \leq u_t\left(1 - c_0 u_t^p\right) \implies \frac{u_t}{u_{t+1}} \geq \frac{1}{1 - c_0 u_t^p} \geq \frac{1}{\exp\left(-c_0 u_t^p\right)} = \exp\left(c_0 u_t^p\right),$$

where the last inequality again uses the exponential inequality. Notice in dividing by $u_{t+1}$ and $(1 - c_0 u_t^p)$, the inequality does not flip since both terms are positive: $u_t, u_{t+1}$ are positive according to the conditions of the lemma, and $(1 - c_0 u_t^p)$ is positive because of the inequality on the left side of the implication in the above. Since $p \geq 0$, we may raise both sides of the above inequality to the power of $p$ to show (17), finishing our proof. ■

### A.3 Proof Of Lemma 25

In this section we prove Lemma 25, by separately bounding the quantities $\lambda_{\min}^{-1}$, $\gamma^{-1}$ and $\mu_{\max}$, through a sequence of lemmas. We will use the next result repeatedly.

**Lemma 33** *If $\mathbf{A}$ is an $n \times n$ invertible matrix with $-1, 0, +1$ entries, then $\min_{\vec{x}:\|\vec{x}\|=1} \|\mathbf{A}\vec{x}\|$ is at least $1/n! = 2^{-O(n \ln n)}$.*

**Proof** It suffices to show that $\|\mathbf{A}^{-1}\vec{x}\| \leq n!$ for any $\vec{x}$ with unit norm. Now $\mathbf{A}^{-1} = \mathrm{adj}(\mathbf{A})/\det(\mathbf{A})$ where $\mathrm{adj}(\mathbf{A})$ is the adjoint of $\mathbf{A}$, whose $i,j$-th entry is the $i,j$th cofactor of $\mathbf{A}$ (given by $(-1)^{i+j}$ times the determinant of the $n-1 \times n-1$ matrix obtained by removing the $i$th row and $j$th column of $\mathbf{A}$), and $\det(\mathbf{A})$ is the determinant of $\mathbf{A}$. The determinant of any $k \times k$ matrix $G$ can be

written as $\sum_\sigma \mathrm{sgn}(\sigma)\prod_{i=1}^k G_{i,\sigma(i)}$, where $\sigma$ ranges over all the permutations of $1,\ldots,k$. Therefore each entry of $\mathrm{adj}(\mathbf{A})$ is at most $(n-1)!$, and $\det(\mathbf{A})$ is a non-zero integer. Therefore $\|\mathbf{A}^{-1}\vec{x}\| = \|\mathrm{adj}(\mathbf{A})\vec{x}\|/\det(\mathbf{A}) \le n!\|\vec{x}\|$, and the proof is complete. ■

We first show our bound holds for $\lambda_{\min}$.

**Lemma 34** *Suppose* $\mathbf{M}$ *has* $-1,0,+1$ *entries, and let* $\mathbf{M}_F, \lambda_{\min}$ *be as in Corollary 23. Then* $\lambda_{\min} \ge 1/m!$.

**Proof** Let $\mathbf{A}$ denote the matrix $\mathbf{M}_F$. It suffices to show that $\mathbf{A}$ does not squeeze too much the norm of any vector orthogonal to the null-space $\ker\mathbf{A} \triangleq \{\boldsymbol{\eta} : \mathbf{A}\boldsymbol{\eta} = \mathbf{0}\}$ of $\mathbf{A}$, that is, $\|\mathbf{A}\boldsymbol{\lambda}\| \ge (1/m!)\|\boldsymbol{\lambda}\|$ for any $\boldsymbol{\lambda} \in \ker\mathbf{A}^\perp$. We first characterize $\ker\mathbf{A}^\perp$ and then study how $\mathbf{A}$ acts on this subspace.

Let the rank of $\mathbf{A}$ be $k \le m$ (notice $\mathbf{A} = \mathbf{M}_F$ has $N$ columns and fewer than $m$ rows). Without loss of generality, assume the first $k$ columns of $\mathbf{A}$ are independent. Then every column of $\mathbf{A}$ can be written as a linear combination of the first $k$ columns of $\mathbf{A}$, and we have $\mathbf{A} = \mathbf{A}'[\mathbf{I}|\mathbf{B}]$ (that is, the matrix $\mathbf{A}$ is the product of matrices $\mathbf{A}'$ and $[\mathbf{I}|\mathbf{B}]$), where $\mathbf{A}'$ is the submatrix consisting of the first $k$ columns of $\mathbf{A}$, $\mathbf{I}$ is the $k\times k$ identity matrix, and $\mathbf{B}$ is some $k\times(N-k)$ matrix of linear combinations (here $|$ denotes concatenation). The null-space of $\mathbf{A}$ consists of $\vec{x}$ such that $\mathbf{0} = \mathbf{A}\vec{x} = \mathbf{A}'[\mathbf{I}|\mathbf{B}]\vec{x} = \mathbf{A}'(\vec{x}_k + \mathbf{B}\vec{x}_{-k})$, where $\vec{x}_k$ is the first $k$ coordinates of $\vec{x}$, and $\vec{x}_{-k}$ the remaining $N-k$ coordinates. Since the columns of $\mathbf{A}'$ are independent, this happens if and only if $\vec{x}_k = -\mathbf{B}\vec{x}_{-k}$. Therefore $\ker\mathbf{A} = \left\{(-\mathbf{B}\vec{z},\vec{z}) : \vec{z}\in\mathbb{R}^{N-k}\right\}$. Since a vector $\vec{x}$ lies in the orthogonal subspace of $\ker\mathbf{A}$ if it is orthogonal to every vector in the latter, we have

$$\ker\mathbf{A}^\perp = \left\{(\vec{x}_k,\vec{x}_{-k}) : \langle\vec{x}_k, \mathbf{B}\vec{z}\rangle = \langle\vec{x}_{-k},\vec{z}\rangle, \forall\vec{z}\in\mathbb{R}^{N-K}\right\}.$$

We next see how $\mathbf{A}$ acts on this subspace. Recall $\mathbf{A} = \mathbf{A}'[\mathbf{I}|\mathbf{B}]$ where $\mathbf{A}'$ has $k$ independent columns. By basic linear algebra, the row rank of $\mathbf{A}'$ is also $k$, and assume without loss of generality that the first $k$ rows of $\mathbf{A}'$ are independent. Denote by $\mathbf{A}_k$ the $k\times k$ submatrix of $\mathbf{A}'$ formed by these $k$ rows. Then for any vector $\vec{x}$,

$$\|\mathbf{A}\vec{x}\| = \|\mathbf{A}'[\mathbf{I}|\mathbf{B}]\vec{x}\| = \|\mathbf{A}'(\vec{x}_k + \mathbf{B}\vec{x}_{-k})\| \ge \|\mathbf{A}_k(\vec{x}_k + \mathbf{B}\vec{x}_{-k})\| \ge \frac{1}{k!}\|\vec{x}_k + \mathbf{B}\vec{x}_{-k}\|,$$

where the last inequality follows from Lemma 33. To finish the proof, it suffices to show that $\|\vec{x}_k + \mathbf{B}\vec{x}_{-k}\| \ge \|\vec{x}\|$ for $\vec{x}\in\ker\mathbf{A}^\perp$. Indeed, by expanding out $\|\vec{x}_k + \mathbf{B}\vec{x}_{-k}\|^2$ as inner product with itself, we have

$$\|\vec{x}_k + \mathbf{B}\vec{x}_{-k}\|^2 = \|\vec{x}_k\|^2 + \|\mathbf{B}\vec{x}_{-k}\|^2 + 2\langle\vec{x}_k, \mathbf{B}\vec{x}_{-k}\rangle \ge \|\vec{x}_k\|^2 + 2\|\vec{x}_{-k}\|^2 \ge \|\vec{x}\|^2,$$

where the first inequality follows since $\vec{x}\in\ker\mathbf{A}^\perp$ implies $\langle\vec{x}_k,\mathbf{B}\vec{x}_{-k}\rangle = \langle\vec{x}_{-k},\vec{x}_{-k}\rangle$. ■

To show the bounds on $\gamma^{-1}$ and $\mu_{\max}$, we will need an intermediate result.

**Lemma 35** *Suppose* $\mathbf{A}$ *is a matrix, and* $\mathbf{b}$ *a vector, both with* $-1,0,1$ *entries. If* $\mathbf{A}\vec{x} = \mathbf{b}, \vec{x}\ge\mathbf{0}$ *is solvable, then there is a solution satisfying* $\|\vec{x}\| \le k\cdot k!$, *where* $k = \mathrm{rank}(\mathbf{A})$.

**Proof** Pick a solution $\vec{x}$ with maximum number of zeroes. Let $J$ be the set of coordinates for which $x_i$ is zero. We first claim that there is no other solution $\vec{x}'$ which is also zero on the set $J$. Suppose there were such an $\vec{x}'$. Note any point $\vec{p}$ on the infinite line joining $\vec{x},\vec{x}'$ satisfies $\mathbf{A}\vec{p} = \mathbf{b}$, and $\vec{p}_J = \mathbf{0}$ (that is, $p_{i'} = 0$ for $i'\in J$). If $i$ is any coordinate not in $J$ such that $x_i \ne x'_i$, then for some point $\vec{p}^i$

along the line, we have $\vec{p}^i_{J\cup\{i\}} = \mathbf{0}$. Choose $i$ so that $\vec{p}^i$ is as close to $\vec{x}$ as possible. Since $\vec{x} \geq \mathbf{0}$, by continuity this would also imply that $\vec{p}^i \geq \mathbf{0}$. But then $\vec{p}^i$ is a solution with more zeroes than $\vec{x}$, a contradiction.

The claim implies that the reduced problem $\mathbf{A}'\tilde{\vec{x}} = \mathbf{b}, \tilde{\vec{x}} \geq \mathbf{0}$, obtained by substituting $\vec{x}_J = \mathbf{0}$, has a unique solution. Let $k = \text{rank}(\mathbf{A}')$, $\mathbf{A}_k$ be a $k \times k$ submatrix of $\mathbf{A}'$ with full rank, and $\mathbf{b}_k$ be the restriction of $\mathbf{b}$ to the rows corresponding to those of $\mathbf{A}_k$ (note that $\mathbf{A}'$, and hence $\mathbf{A}_k$, contain only $-1, 0, +1$ entries). Then, $\mathbf{A}_k\tilde{\vec{x}} = \mathbf{b}_k, \tilde{\vec{x}} \geq \mathbf{0}$ is equivalent to the reduced problem. In particular, by uniqueness, solving $\mathbf{A}_k\tilde{\vec{x}} = \mathbf{b}_k$ automatically ensures the obtained $\vec{x} = (\tilde{\vec{x}}, \mathbf{0}_J)$ is a non-negative solution to the original problem, and satisfies $\|\vec{x}\| = \|\tilde{\vec{x}}\|$. But, by Lemma 33,

$$\|\tilde{\vec{x}}\| \leq k! \|\mathbf{A}_k\tilde{\vec{x}}\| = k!\|\mathbf{b}_k\| \leq k \cdot k!.$$

■

The bound on $\gamma^{-1}$ follows easily.

**Lemma 36** *Let $\gamma, \boldsymbol{\eta}^\dagger$ be as in Item 1 of Lemma 15. Then $\boldsymbol{\eta}^\dagger$ can be chosen such that $\gamma \geq 1/\left(\sqrt{N}m \cdot m!\right) \geq 2^{-O(m \ln m)}$.*

**Proof** We know that $\mathbf{M}(\boldsymbol{\eta}^\dagger/\gamma) = \mathbf{b}$, where $\mathbf{b}$ is zero on the set $F$ and at least 1 for every example in the zero loss set $Z$ (as given by Item 1 of Lemma 15). Since $\mathbf{M}$ is closed under complementing columns, we may assume in addition that $\boldsymbol{\eta}^\dagger \geq \mathbf{0}$. Introduce slack variables $z_i$ for $i \in Z$, and let $\tilde{\mathbf{M}}$ be $\mathbf{M}$ augmented with the columns $-\vec{e}_i$ for $i \in Z$, where $\vec{e}_i$ is the standard basis vector with 1 on the $i$th coordinate and zero everywhere else. Then, by setting $\vec{z} = \mathbf{M}(\boldsymbol{\eta}^\dagger/\gamma) - \mathbf{b}$, we have a solution $(\boldsymbol{\eta}^\dagger/\gamma, \vec{z})$ to the system $\tilde{\mathbf{M}}\vec{x} = \mathbf{b}, \vec{x} \geq \mathbf{0}$. Applying Lemma 35, we know there exists some solution $(\vec{y}, \vec{z}')$ with norm at most $m \cdot m!$ (here $\vec{z}'$ corresponds to the slack variables). Observe that $\vec{y}/\|\vec{y}\|_1$ is a valid choice for $\boldsymbol{\eta}^\dagger$ yielding a $\gamma$ of $1/\|\vec{y}\|_1 \geq 1/(\sqrt{N}m \cdot m!)$. ■

To show the bound for $\mu_{\max}$ we will need a version of Lemma 35 with strict inequality.

**Corollary 37** *Suppose $\mathbf{A}$ is a matrix, and $\mathbf{b}$ a vector, both with $-1, 0, 1$ entries. If $\mathbf{A}\vec{x} = \mathbf{b}, \vec{x} > \mathbf{0}$ is solvable, then there is a solution satisfying $\|\vec{x}\| \leq 1 + k \cdot k!$, where $k = \text{rank}(\mathbf{A})$.*

**Proof** Using Lemma 35, pick a solution to $\mathbf{A}\vec{x} = \mathbf{b}, \vec{x} \geq \mathbf{0}$ with norm at most $k \cdot k!$. If $\vec{x} > \mathbf{0}$, then we are done. Otherwise let $\vec{y} > \mathbf{0}$ satisfy $\mathbf{A}\vec{x} = \mathbf{b}$, and consider the segment joining $\vec{x}$ and $\vec{y}$. Every point $\vec{p}$ on the segment satisfies $\mathbf{A}\vec{p} = b$. Further any coordinate becomes zero at most once on the segment. Therefore, there are points arbitrarily close to $\vec{x}$ on the segment with positive coordinates that satisfy the equation, and these have norms approaching that of $\vec{x}$. ■

We next characterize the feature matrix $\mathbf{M}_F$ restricted to the finite-loss examples, which might be of independent interest.

**Lemma 38** *If $\mathbf{M}_F$ is the feature matrix restricted to the finite-loss examples $F$ (as given by Item 2 of Lemma 15), then there exists a positive linear combination $\vec{y} > \mathbf{0}$ such that $\mathbf{M}_F^T\vec{y} = \mathbf{0}$.*

**Proof** Item 3 of the decomposition lemma states that whenever the loss $\ell^{\vec{x}}(F)$ of a vector is bounded by $m$, then the largest margin $\max_{i \in F}(\mathbf{M}_F\vec{x})_i$ is at most $\mu_{\max}$. This implies that there is no vector $\vec{x}$ such that $\mathbf{M}_F\vec{x} \geq \mathbf{0}$ and at least one of the margins $(\mathbf{M}_F\vec{x})_i$ is positive; otherwise, an arbitrarily large multiple of $\vec{x}$ would still have loss at most $m$, but margin exceeding the constant $\mu_{\max}$. In other words,

$\mathbf{M}_F\vec{x} \geq \mathbf{0}$ implies $\mathbf{M}_F\vec{x} = \mathbf{0}$. In particular, the subspace of possible margin vectors $\left\{\mathbf{M}_F\vec{x} : \vec{x} \in \mathbb{R}^N\right\}$ is disjoint from the convex set $\Delta_F$ of distributions over examples in $F$, which consists of points in $\mathbb{R}^{|F|}$ with all non-negative and at least one positive coordinates. By the Hahn-Banach Separation theorem, there exists a hyperplane separating these two bodies, that is, there is a $\vec{y} \in \mathbb{R}^{|F|}$, such that for any $\vec{x} \in \mathbb{R}^N$ and $\vec{p} \in \Delta_F$, we have $\langle \vec{y}, \mathbf{M}_F\vec{x}\rangle \leq 0 < \langle \vec{y}, \vec{p}\rangle$. By choosing $\vec{p} = \vec{e}_i$ for various $i \in F$, the second inequality yields $\vec{y} > \mathbf{0}$. Since $\mathbf{M}_F\vec{x} = -\mathbf{M}_F(-\vec{x})$, the first inequality implies that equality holds for all $\vec{x}$, that is, $\vec{y}^T\mathbf{M}_F = \mathbf{0}^T$. ■

We can finally upper-bound $\mu_{\max}$.

**Lemma 39** *Let $F, \mu_{\max}$ be as in Items 2,3 of the decomposition lemma. Then $\mu_{\max} \leq \ln m \cdot |F|^{1.5} \cdot |F|! \leq 2^{O(m \ln m)}$.*

**Proof** Pick any example $i \in F$ and any combination $\boldsymbol{\lambda}$ whose loss on $F$, $\sum_{i \in F} e^{-(\mathbf{M}\boldsymbol{\lambda})_i}$, is at most $m$. Let $\mathbf{b}$ be the $i^{\text{th}}$ row of $\mathbf{M}$, and let $\mathbf{A}^T$ be the matrix $\mathbf{M}_F$ without the $i$th row. Then Lemma 38 says that $\mathbf{A}\vec{y} = -\mathbf{b}$ for some positive vector $\vec{y} > \mathbf{0}$. This implies the margin of $\boldsymbol{\lambda}$ on example $i$ is $(\mathbf{M}\boldsymbol{\lambda})_i = -\vec{y}^T\mathbf{A}^T\boldsymbol{\lambda}$. Since the loss of $\boldsymbol{\lambda}$ on $F$ is at most $m$, each margin on $F$ is at least $-\ln m$, and therefore $\max_{i \in F}\left(-\mathbf{A}^T\boldsymbol{\lambda}\right)_i \leq \ln m$. Hence, the margin of example $i$ can be bounded as $(\mathbf{M}\boldsymbol{\lambda})_i = \left\langle \vec{y}^T, -\mathbf{A}^T\boldsymbol{\lambda}\right\rangle \leq \ln m \|\vec{y}\|_1$. Using Corollary 37, we can find $\vec{y}$ with bounded norm, $\|\vec{y}\|_1 \leq \sqrt{|F|}\|\vec{y}\| \leq \sqrt{|F|}(1 + k \cdot k!)$ , where $k = \text{rank}(\mathbf{A}) \leq \text{rank}(\mathbf{M}_F) \leq |F|$. The proof follows. ■

## References


P. L. Bartlett and M. Traskin. AdaBoost is consistent. *Journal of Machine Learning Research*, 8: 2347–2368, 2007.

P. J. Bickel, Y. Ritov, and A. Zakai. Some theory for generalized boosting algorithms. *Journal of Machine Learning Research*, 7:705–732, 2006.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

L. Breiman. Prediction games and arcing classifiers. *Neural Computation*, 11(7):1493–1517, 1999.

M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48(1/2/3), 2002.

M. Frean and T. Downs. A simple cost function for boosting. Technical report, Department of Computer Science and Electrical Engineering, University of Queensland, 1998.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, Aug. 1997.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2):337–374, Apr. 2000.

J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5), Oct. 2001.

A. J. Grove and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1998.

D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. Springer, third edition, 2008.

Z. Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, Jan. 1992.

L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. In *Advances in Neural Information Processing Systems 12*, 2000.

T. Onoda, G. Rätsch, and K.-R. Müller. An asymptotic analysis of AdaBoost in the binary classification case. In *Proceedings of the 8th International Conference on Artificial Neural Networks*, pages 195–200, 1998.

G. Rätsch and M. K. Warmuth. Efficient margin maximizing with boosting. *Journal of Machine Learning Research*, 6:2131–2152, 2005.

G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3): 287–320, 2001.

G. Rätsch, S. Mika, and M. K. Warmuth. On the convergence of leveraging. In *Advances in Neural Information Processing Systems 14*, 2002.

C. Rudin, I. Daubechies, and R. E. Schapire. The dynamics of AdaBoost: Cyclic behavior and convergence of margins. *Journal of Machine Learning Research*, 5:1557–1595, 2004.

C. Rudin, R. E. Schapire, and I. Daubechies. Analysis of boosting algorithms using the smooth margin function. *Annals of Statistics*, 35(6):2723–2768, 2007.

R. E. Schapire. The convergence rate of AdaBoost. In *The 23rd Conference on Learning Theory*, 2010. open problem.

R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, Dec. 1999.

R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5):1651–1686, October 1998.

S. Shalev-Shwartz and Y. Singer. On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms. In *21st Annual Conference on Learning Theory*, 2008.

M. Telgarsky. The convergence rate of AdaBoost and friends. http://arxiv.org/abs/1101.4752, January 2011.

T. Zhang and B. Yu. Boosting with early stopping: Convergence and consistency. *Annals of Statistics*, 33(4):1538–1579, 2005.

Journal of Machine Learning Research 14 (2013) 2349-2353 Submitted 3/13; Published 8/13

# Orange: Data Mining Toolbox in Python

**Janez Demšar** JANEZ.DEMSAR@FRI.UNI-LJ.SI
**Tomaž Curk** TOMAZ.CURK@FRI.UNI-LJ.SI
**Aleš Erjavec** ALES.ERJAVE@FRI.UNI-LJ.SI
**Črt Gorup** CRT.GORUP@FRI.UNI-LJ.SI
**Tomaž Hočevar** TOMAZ.HOCEVAR@FRI.UNI-LJ.SI
**Mitar Milutinovič** MITAR.MILUTINOVIC@FRI.UNI-LJ.SI
**Martin Možina** MARTIN.MOZINA@FRI.UNI-LJ.SI
**Matija Polajnar** MATIJA.POLAJNAR@FRI.UNI-LJ.SI
**Marko Toplak** MARKO.TOPLAK@FRI.UNI-LJ.SI
**Anže Starič** ANZE.STARIC@FRI.UNI-LJ.SI
**Miha Štajdohar** MIHA.STAJDOHAR@FRI.UNI-LJ.SI
**Lan Umek** LAN.UMEK@FRI.UNI-LJ.SI
**Lan Žagar** LAN.ZAGAR@FRI.UNI-LJ.SI
**Jure Žbontar** JURE.ZBONTAR@FRI.UNI-LJ.SI
**Marinka Žitnik** MARINKA.ZITNIK@FRI.UNI-LJ.SI
**Blaž Zupan** BLAZ.ZUPAN@FRI.UNI-LJ.SI
*Faculty of Computer and Information Science*
*University of Ljubljana*
*Tržaška 25, SI-1000 Ljubljana, Slovenia*

## Abstract

Orange is a machine learning and data mining suite for data analysis through Python scripting and visual programming. Here we report on the scripting part, which features interactive data analysis and component-based assembly of data mining procedures. In the selection and design of components, we focus on the flexibility of their reuse: our principal intention is to let the user write simple and clear scripts in Python, which build upon C++ implementations of computationally-intensive tasks. Orange is intended both for experienced users and programmers, as well as for students of data mining.

**Keywords:** Python, data mining, machine learning, toolbox, scripting

## 1. Introduction

Scripting languages have recently risen in popularity in all fields of computer science. Within the context of explorative data analysis, they offer advantages like interactivity and fast prototyping by gluing together existing components or adapting them for new tasks. Python is a scripting language with clear and simple syntax, which also made it popular in education. Its relatively slow execution can be circumvented by using libraries that implement the computationally intensive tasks in low-level languages.

Python offers a huge number of extension libraries. Many are related to machine learning, including several general packages like scikit-learn (Pedregosa et al., 2011), PyBrain (Schaul et al., 2010) and mlpy (Albanese et al., 2012). Orange was conceived in late 1990s and is among the oldest of such tools. It focuses on simplicity, interactivity through scripting, and component-based design.

©2013 Janez Demšar, Tomaž Curk, Aleš Erjavec, Črt Gorup, Tomaž Hočevar, Mitar Milutinovič, Martin Možina, Matija Polajnar, Marko Toplak, Anže Starič, Miha Štajdohar, Lan Umek, Lan Žagar, Jure Žbontar, Marinka Žitnik and Blaž Zupan

## 2. Toolbox Overview

Orange library is a hierarchically-organized toolbox of data mining components. The low-level procedures at the bottom of the hierarchy, like data filtering, probability assessment and feature scoring, are assembled into higher-level algorithms, such as classification tree learning. This allows developers to easily add new functionality at any level and fuse it with the existing code. The main branches of the component hierarchy are:

**data management and preprocessing** for data input and output, data filtering and sampling, imputation, feature manipulation (discretization, continuization, normalization, scaling and scoring), and feature selection,

**classification** with implementations of various supervised machine learning algorithms (trees, forests, instance-based and Bayesian approaches, rule induction), borrowing from some well-known external libraries such as LIBSVM (Chang and Lin, 2011),

**regression** including linear and lasso regression, partial least square regression, regression trees and forests, and multivariate regression splines,

**association** for association rules and frequent itemsets mining,

**ensembles** implemented as wrappers for bagging, boosting, forest trees, and stacking,

**clustering,** which includes $k$-means and hierarchical clustering approaches,

**evaluation** with cross-validation and other sampling-based procedures, functions for scoring the quality of prediction methods, and procedures for reliability estimation,

**projections** with implementations of principal component analysis, multi-dimensional scaling and self-organizing maps.

The library is designed to simplify the assembly of data analysis workflows and crafting of data mining approaches from a combination of existing components. Besides broader range of features, Orange differs from most other Python-based machine learning libraries by its maturity (over 15 years of active development and use), a large user community supported through an active forum, and extensive documentation that includes tutorials, scripting examples, data set repository, and documentation for developers. Orange scripting library is also a foundation for its visual programming platform with graphical user interface components for interactive data visualization.

The two major packages that are similar to Orange and are still actively developed are scikit-learn (Pedregosa et al., 2011) and mlpy (Albanese et al., 2012). Both are more tightly integrated with numpy and at present better blend into Python's numerical computing habitat. Orange was on the other hand inspired by classical machine learning that focuses on symbolic methods. Rather than supporting only numerical arrays, Orange data structures combine symbolic, string and numerical attributes and meta data information. User can for instance refer to variables and values by their names. Variables store mapping functions, a mechanism which for instance allows classifiers to define transformations on training data that are then automatically applied when making predictions. These features also make Orange more suitable for interactive, explorative data analysis.

## 3. Scripting Examples

Let us illustrate the utility of Orange through an example of data analysis in Python shell:

```
>>> import Orange
>>> data = Orange.data.Table("titanic")
>>> len(data)
2201
>>> nbc = Orange.classification.bayes.NaiveLearner()
>>> svm = Orange.classification.svm.SVMLearner()
>>> stack = Orange.ensemble.stacking.StackedClassificationLearner([nbc,svm])
>>> res = Orange.evaluation.testing.cross_validation([nbc, svm, stack], data)
>>> Orange.evaluation.scoring.AUC(res)
[0.7148500435874006, 0.731873352343742, 0.7635593576372478]
```

We first read the data on survival of 2,201 passengers from HMS Titanic and construct a set of learning algorithms: a naive Bayesian and SVM learner, and a stacked combination of the two (Wolpert, 1992). We then cross-validate the learners and report the area under ROC curves.

Running stacking on the subset of about 470 female passengers improves AUC score:

```
>>> females = Orange.data.Table([d for d in data if d["sex"]=="female"])
>>> len(females)
470
>>> res = Orange.evaluation.testing.cross_validation([stack], females)
>>> Orange.evaluation.scoring.AUC(res)
[0.8124014221073045]
```

We can use existing machine learning components to craft new ones. For instance, learning algorithms must implement a `__call__` operator that accepts the training data and, optionally, data instance weights, and has to return a model. The following example defines a new learner that encloses another learner into a feature selection wrapper: it sorts the features by their information gain (as implemented in `Orange.feature.scoring.InfoGain`), constructs a new data set with only the `m` best features and calls the `base_learner`.

```
class FSSLearner(Orange.classification.PyLearner):
    def __init__(self, base_learner, m=5):
        self.m = m
        self.base_learner = base_learner

    def __call__(self, data, weights=None):
        gain = Orange.feature.scoring.InfoGain()
        best = sorted(data.domain.features, key=lambda x: -gain(x, data))[:self.m]
        domain = Orange.data.Domain(best + [data.domain.class_var])
        new_data = Orange.data.Table(domain, data)
        model = self.base_learner(new_data, weights)
        return Orange.classification.PyClassifier(classifier=model)
```

Below we compare the original and wrapped naive Bayesian classifier on a data set with 106 instances and 57 features:

```
>>> data = Orange.data.Table("promoters")
>>> len(data), len(data.domain.features)
(106, 57)
```

```
>>> bayes = Orange.classification.bayes.NaiveLearner()
>>> res = Orange.evaluation.testing.cross_validation([bayes, FSSLearner(bayes)], data)
>>> Orange.evaluation.scoring.AUC(res)
[0.9329999999999998, 0.945]
```

## 4. Code Design

Orange's core is a collection of nearly 200 C++ classes that cover the basic data structures and majority of preprocessing and modeling algorithms. The C++ part is self-contained, without any calls to Python that would induce unnecessary overhead. The core includes several open source libraries, including LIBSVM (Chang and Lin, 2011), LIBLINEAR (Fan et al., 2008), Earth (see `http://www.milbo.users.sonic.net/earth`), QHull (Barber et al., 1996) and a subset of BLAS (Blackford et al., 2002). The Python layer also uses popular Python libraries numpy for linear algebra, networkx (Hagberg et al., 2008) for working with networks and matplotlib (Hunter, 2007) for basic visualization.

The upper layer of Orange is written in Python and includes procedures that are not time-critical. This is also the place at which users outside the core development group most easily contribute to the project.

Automated testing of the system relies on over 1,500 regression tests that are mostly based on code snippets from extensive documentation. A part of the code is also covered with stricter unit tests.

## 5. Availability, Requirements and Plans for the Future

Orange is free software released under GPL. The code is hosted on Bitbucket repository (`https://bitbucket.org/biolab/orange`). Orange runs on Windows, Mac OS X and Linux, and can also be installed from the Python Package Index repository (`pip install Orange`). Binary installer for Windows and application bundle for Mac OS X are available on project's web site (`http://orange.biolab.si`).

Orange currently runs on Python 2.6 and 2.7. A version for Python 3 and higher is under development. There, we will switch to numpy-based data structures and scrap the C++ core in favor of using routines from numpy and scipy (Jones et al., 2001–), scikit-learn (Pedregosa et al., 2011) and similar libraries that did not exist when Orange was first conceived. Despite planned changes in the core, we will maintain backward compatibility. For existing users, the changes of the Python interface will be minor.

## Acknowledgments

We would like to acknowledge support for this project from the Slovenian Research Agency (P2-0209, J2-9699, L2-1112), National Institute of Health (P01-HD39691), and Astra Zeneca. We thank the anonymous reviewers for their constructive comments.

## References

D. Albanese, R. Visintainer, S. Merler, S. Riccadonna, G. Jurman, and C. Furlanello. mlpy: Machine learning Python. *CoRR*, abs/1202.6548, 2012.

C. B. Barber, D. P. Dobkin, and H. T. Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Trans. on Mathematical Software*, 22(4), 1996.

L. S. Blackford, A. Petitet, R. Pozo, K. Remington, R. C. Whaley, J. Demmel, J. Dongarra, I. Duff, S. Hammarling, and G. Henry. An updated set of basic linear algebra subprograms (BLAS). *ACM Transactions on Mathematical Software*, 28(2):135–151, 2002.

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pages 11–15, Pasadena, CA USA, 2008.

J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering*, 9(3): 90–95, 2007.

E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL `http://www.scipy.org/`.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg. scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber. PyBrain. *Journal of Machine Learning Research*, 11:743–746, 2010.

D. H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

Journal of Machine Learning Research 14 (2013) 2355-2359 Submitted 2/12; Revised 5/13; Published 8/13

# *Tapkee*: An Efficient Dimension Reduction Library

**Sergey Lisitsyn** LISITSYN.S.O@GMAIL.COM
*Samara State Aerospace University*
*34 Moskovskoye shosse*
*443086 Samara, Russia*

**Christian Widmer*** CWIDMER@CBIO.MSKCC.ORG
*Computational Biology Center*
*Memorial Sloan-Kettering Cancer Center*
*415 E 68th street*
*New York City 10065 NY, USA*

**Fernando J. Iglesias Garcia** FJIG@KTH.SE
*KTH Royal Institute of Technology*
*79 Valhallavägen*
*10044 Stockholm, Sweden*

**Editor:** Cheng Soon Ong

## Abstract

We present *Tapkee*, a C++ template library that provides efficient implementations of more than 20 widely used dimensionality reduction techniques ranging from *Locally Linear Embedding* (Roweis and Saul, 2000) and *Isomap* (de Silva and Tenenbaum, 2002) to the recently introduced *Barnes-Hut-SNE* (van der Maaten, 2013). Our library was designed with a focus on performance and flexibility. For performance, we combine efficient multi-core algorithms, modern data structures and state-of-the-art low-level libraries. To achieve flexibility, we designed a clean interface for applying methods to user data and provide a callback API that facilitates integration with the library. The library is freely available as open-source software and is distributed under the permissive *BSD 3-clause* license. We encourage the integration of *Tapkee* into other open-source toolboxes and libraries. For example, *Tapkee* has been integrated into the codebase of the *Shogun* toolbox (Sonnenburg et al., 2010), giving us access to a rich set of kernels, distance measures and bindings to common programming languages including Python, Octave, Matlab, R, Java, C#, Ruby, Perl and Lua. Source code, examples and documentation are available at `http://tapkee.lisitsyn.me`.

**Keywords:** dimensionality reduction, machine learning, C++, open source software

## 1. Introduction

The aim of dimension reduction is to find low-dimensional representations of data to facilitate data visualization, interpretation and preprocessing for further analysis. It has applications in diverse fields including bioinformatics, physics and computer vision. Our library, *Tapkee*, provides numerous efficient implementations of both linear and non-linear dimension reduction algorithms, ranging from established methods to more recent developments in the field.

Implementations of modern dimensionality reduction algorithms are available in several other open-source toolboxes, such as *Scikit-learn* (Pedregosa et al., 2011), *Waffles* (Gashler, 2011) and the

∗. Also with Machine Learning Group, Technische Universität Berlin, Franklinstr. 28/29, 10587 Berlin, Germany.

©2013 Sergey Lisitsyn, Christian Widmer and Fernando J. Iglesias Garcia.

*Matlab Toolbox for Dimensionality Reduction* by Laurens van der Maaten. In contrast to existing toolkits, our aim is to provide a generic C++ library in the spirit of the *Standard Template Library* to allow for greater flexibility. In particular, we designed *Tapkee* to be callback-centric (i.e., externally defined functions may be passed to the library as arguments), which naturally enables the user to combine our library with custom distance or similarity measures, which are at the core of the most dimensionality reduction algorithms. Our second central design goal is efficiency. For this, we combine advanced data structures (such as cover tree by Beygelzimer et al., 2006), with efficient libraries (as ARPACK) and carefully engineered, well-tested C++ code. Furthermore, we provide parallel implementations of many algorithms using *OpenMP* and preview preliminary support for *GPU* computing. Finally, we want *Tapkee* to be easy to use. Therefore, we provide a simple, well-documented API and usage examples for each method. To support the user in the choice of algorithms, we provide mathematical background for each method on the project website.

To facilitate the use of *Tapkee* for the end-user, we distribute our library as part of the *Shogun* toolbox, providing access to its language bindings to such languages as Python, MATLAB, Java and a rich set of kernel and distance functions. However, we emphasize that *Tapkee* is a stand-alone library. The integration of *Tapkee* into C++ projects or other toolkits is therefore highly encouraged by its software design (a template library with callbacks and only few dependencies) as well as the choice of a permissive software license (*BSD*). Source code, documentation and various graphical and code examples can be found on the project website: `http://tapkee.lisitsyn.me`.

## 2. Implemented Algorithms

Currently, our toolkit provides implementations of the following algorithms:

1. Local neighborhood methods, such as *Locally Linear Embedding* (Roweis and Saul, 2000) and *Neighborhood Preserving Embedding* (He et al., 2005). This group also includes *Hessian Locally Linear Embedding* (Donoho and Grimes, 2003), *Local Tangent Space Alignment* (Zhang and Zha, 2004) and *Linear Local Tangent Space Alignment* (Zhang et al., 2007);

2. Classic distance-based methods such as *Multidimensional Scaling* and *Isomap* along with its landmark approximations such as *Landmark Multidimensional Scaling* and *Landmark Isomap* (de Silva and Tenenbaum, 2002);

3. Graph-based methods, such as *Laplacian Eigenmaps* (Belkin and Niyogi, 2002), *Locality Preserving Projections* (He and Niyogi, 2003) and *Diffusion Maps* (Coifman and Lafon, 2006);

4. Iterative methods such as *Stochastic Proximity Embedding* (Agrafiotis, 2003), *Factor Analysis*, *t-SNE* and *Barnes-Hut-SNE* (van der Maaten, 2013);

5. Widely known *PCA* and *kernel PCA*, *Randomized PCA* (Halko et al., 2011) algorithms.

## 3. Software Design

*Tapkee* is a C++ pure template library with a flexible, modular structure. In practice, this allows easier integration of the library code into existing projects without the need for linking. Furthermore, the library code is specialized during compilation with capabilities of compile-time optimization and customization (e.g., custom floating point precision or linear algebra solvers). As a major design


Figure 1: Swissroll 3D data and its embedding constructed with *Isomap* (on the left); embedding of the faces data set computed with *t-SNE* (in the middle); embedding of English words computed with *kernel LLE* using *Ratcliff-Obershelp* as similarity measure (on the right).

principle, our library is engineered to be *callback-based* (with externally defined functions passed to the library as arguments). The algorithms in our library are implemented as generically as possible (formulated in terms of callback functions) with callbacks providing the interface between the user data and algorithm. For example, the user may want to embed biological sequences using a string kernel from a third party library, which—using callbacks—can be passed to *Tapkee* as a custom similarity measure. A major benefit of our callback-centric design is the improved re-usability of our software, as user-defined callback functions may contain complex custom code or operate on custom data structures without any changes to our library code.

Most dimensionality reduction algorithms rely on linear algebra operations, nearest neighbor finding and the solving of eigenproblems. To address the need for high performance linear algebra we leverage the *Eigen3* template library. This makes our implementations safe, easy to read and fast. Next, we approach the nearest neighbor problem with the *vantage point tree* and the *cover tree* data structures (Beygelzimer et al., 2006) which can be used with arbitrary metric spaces. Finally, to solve eigenproblems arising in most of the implemented algorithms we employ three methods: QR decomposition, Lanczos method from the *ARPACK* library and a randomized method described in Halko et al. (2011). The library supports all major platforms (*Linux*, *Mac OS X* and *Windows*). To ensure quality we combine unit-testing and continuous integration with *Travis*.

## 4. Applications

We have successfully applied *Tapkee* to compute low-dimensional representations of various data sets including images and biological sequences. Images were taken from the widely-known *MNIST* data set. To demonstrate an embedding of string objects, we computed a 2*D* embedding of English words and biological sequences (gene starts) for several organisms. See Figure 1 for examples of 2*D* representations computed with *Tapkee* and the website for more graphical examples. At the time of writing, we are aware of several successful applications of *Tapkee* to problems in hydrodynamics, bioinformatics and exploratory analysis of energy landscapes, from user feedback.

## 5. Comparison To Related Work

We have compared our implementations with their counterparts in *Scikit-learn* 0.13.1, *Matlab toolbox for Dimensionality Reduction* 0.8 and *Waffles* 2013-04-06. We present a performance com-

| | *Tapkee* | | *Scikit-learn* | | *Waffles* | | *MTfDR* | |
|---|---|---|---|---|---|---|---|---|
| | LLE | ISOMAP | LLE | ISOMAP | LLE | ISOMAP | LLE | ISOMAP |
| *Swissroll,* $k = 15$ | **0.45** (0.68) | **3.84** (14.60) | 0.89 | 20.71 | – | 227.71 | 11.88 | 2433.31 |
| *MIT-CBCL,* $k = 20$ | **1.15** (2.57) | **1.20** (3.06) | 12.77 | 5.94 | 64.76 | 20.05 | 2.75 | 5.56 |
| *MNIST,* $k = 20$ | **2.55** (3.63) | **1.93** (4.67) | 8.40 | 9.12 | – | 29.92 | 2.80 | 253.55 |
| *AVIRIS,* $k = 80$ | **5.37** (6.27) | **7.44** (19.85) | 10.76 | 21.53 | – | 62.25 | 25.42 | 763.55 |

Table 1: Performance comparison for *Tapkee*, *Scikit-learn* (Pedregosa et al., 2011), *Waffles* (Gashler, 2011) and *Matlab Toolbox for Dimensionality Reduction* (*MTfDR*). Time is given in seconds (− means no convergence in 3 hours). For *Tapkee*, results for 4 threads and for a single thread (in parentheses) are shown.

parison[1] for implementations of two widely-used methods, which were present in all of the above libraries: Locally Linear Embedding and Isomap. For this comparison we used four data sets: a subset of *MNIST* (2000 vectors of size 784), swissroll with 5000 3D vectors, images from *MIT-CBCL* face recognition data set (384 vectors of size 40,000) and subsampled hyperspectral image obtained from the *AVIRIS* project (2520 vectors of size 224). We report convergence times for each implementation, method and data set in Table 1. We note that when using a single thread *Tapkee* outperforms the other implementations in all but one case. When using 4 cores, *Tapkee* is considerably faster and outperforms the other libraries in all experiments. *Scikit-learn* has the second best overall performance, which hints at the high quality of code. Furthermore, *MTfDR* has good performance for high-dimensional data sets, but shows slow convergence for problems with a large number of vectors. Finally, *Waffles*' LLE implementation sometimes fails to converge - this may be caused by the use of the power iteration method.

In summary, we find that the choice of algorithms and low-level libraries enables *Tapkee* to outperform other implementations under various conditions. For the code to reproduce the above experiments (and additional benchmarks) and an overview of available methods for each toolkit, please see our supplementary webpage `http://iglesias.github.io/tapkee_benchmarks/`.

## 6. Conclusion

We have implemented an efficient and flexible library for dimension reduction with all the methods implemented using state-of-the-art algorithms and data structures. Our library readily handles big data sets and facilitates interaction with custom code using a callback-centric API. To compare our toolkit to existing software, we provide a speed comparison on a range of data sets showing considerable speed-up. We believe the infrastructure provided by *Tapkee* (such as API, modular design, bindings to libraries) can serve as a platform for further dimension reduction development. Finally, we hope that our work proves to be valuable to researchers, students and practitioners.

## Acknowledgments

We would like to thank: L. van der Maaten, J. Langford, V. Gorbatiuk, E. Andreev, D. Harmon and D. Okanohara for their code; G. Rätsch, K.-R. Müller, S. Sonnenburg and D. Kuo for helpful comments. The work was supported by Google (GSoC 2011), the Russian Foundation for Basic Research (13-07-12030, 12-07-00581-a), the Max Planck Society and DFG grant RA1894/1.

1. We used a machine with 2 Intel® Xeon® CPUs and 16Gb of RAM running Ubuntu 12.04.2 and Matlab® 2011a.

## References

D. K. Agrafiotis. Stochastic proximity embedding. *Journal of Computational Chemistry*, 24(10): 1215–1221, 2003.

M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Science*, 14:585–591, 2002. ISSN 10495258.

A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. *Proceedings of the 23rd International Conference on Machine Learning ICML 06*, 1:97–104, 2006.

R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1): 5–30, 2006.

V. de Silva and J. B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. *Advances in Neural Information Processing Systems*, 15:705–712, 2002.

D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences of the United States of America*, 100(10):5591–5596, 2003.

M. S. Gashler. Waffles: A machine learning toolkit. *Journal of Machine Learning Research*, 12 (July):2383–2387, 2011.

N. Halko, P.-G. Martinsson, Y. Shkolnisky, and M. Tygert. An algorithm for the principal component analysis of large data sets. *SIAM Journal on Scientific Computing*, 33(5):2580–2594, 2011.

X. He and P. Niyogi. Locality preserving projections. *Matrix*, 16(December):153–160, 2003.

X. He, D. Cai, S. Yan, and H.-J. Zhang. Neighborhood preserving embedding. *Tenth IEEE International Conference on Computer Vision ICCV05 Volume 1*, 2:1208–1213, 2005.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

S. Sonnenburg, G. Raetsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. De Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 11(x):1799–1802, 2010.

L. van der Maaten. Barnes-Hut-SNE. *arXiv preprint arXiv:1301.3342*, 2013.

T. Zhang, J. Yang, D. Zhao, and X. Ge. Linear local tangent space alignment and application to face recognition. *Neurocomputing*, 70(7-9):1547–1553, 2007. ISSN 09252312.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal on Scientific Computing*, 26(1):313–338, 2004.

Journal of Machine Learning Research 14 (2013) 2361-2376 Submitted 3/13; Revised 6/13; Published 8/13

# On the Mutual Nearest Neighbors Estimate in Regression

**Arnaud Guyader** ARNAUD.GUYADER@UHB.FR
*Université Rennes 2*
*IRMAR & INRIA Rennes*
*Campus de Villejean*
*Place du Recteur Henri Le Moal, CS 24307*
*35043 Rennes Cedex, France*

**Nick Hengartner** NICKH@LANL.GOV
*Information Sciences Group*
*Los Alamos National Laboratory*
*TA-3, Bldg 508, Room 133, Mail Stop B-256*
*Los Alamos, NM 86545, USA*

**Editor:** Ulrike von Luxburg

## Abstract

Motivated by promising experimental results, this paper investigates the theoretical properties of a recently proposed nonparametric estimator, called the Mutual Nearest Neighbors rule, which estimates the regression function $m(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ as follows: first identify the $k$ nearest neighbors of $\mathbf{x}$ in the sample $\mathcal{D}_n$, then keep only those for which $\mathbf{x}$ is itself one of the $k$ nearest neighbors, and finally take the average over the corresponding response variables. We prove that this estimator is consistent and that its rate of convergence is optimal. Since the estimate with the optimal rate of convergence depends on the unknown distribution of the observations, we also present adaptation results by data-splitting.

**Keywords:** nonparametric estimation, nearest neighbor methods, mathematical statistics

## 1. Introduction

Let $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ be a sample of independent and identically distributed (i.i.d.) copies of an $\mathbb{R}^d \times \mathbb{R}$-valued random pair $(\mathbf{X}, Y)$ satisfying $\mathbb{E}Y^2 < \infty$. For fixed $\mathbf{x}$ in $\mathbb{R}^d$, our goal is to estimate the regression function $m(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ using the data $\mathcal{D}_n$. A regression function estimate $m_n(\mathbf{x})$ is said to be weakly consistent if the mean integrated squared error $\mathbb{E}[m_n(\mathbf{X}) - m(\mathbf{X})]^2$ tends to 0 as the sample size $n$ goes to infinity, and is said to be universally weakly consistent if this property holds for all distributions of $(\mathbf{X}, Y)$ with $\mathbb{E}Y^2 < \infty$.

Equip the space $\mathbb{R}^d$ with the standard Euclidean metric. Then, for $\mathbf{x}$ in $\mathbb{R}^d$, the $k$ Nearest Neighbors ($k$NN) estimate for the regression function $m$ is defined by

$$m_n^{k\mathrm{NN}}(\mathbf{x}) = \frac{1}{k} \sum_{i=1}^{k} Y_{(i,n)}(\mathbf{x}),$$

where $(\mathbf{X}_{(1,n)}(\mathbf{x}), Y_{(1,n)}(\mathbf{x})), \ldots, (\mathbf{X}_{(n,n)}(\mathbf{x}), Y_{(n,n)}(\mathbf{x}))$ denotes a reordering of the data according to the increasing values of $d_i = d_i(\mathbf{x}) = \|\mathbf{X}_i - \mathbf{x}\|$ (ties are broken in favor of smallest indices). This procedure is one of the oldest approaches to regression analysis, dating back to Fix and Hodges

©2013 Arnaud Guyader and Nick Hengartner.

(1951, 1952), and is among the most popular nonparametric methods. We refer the reader to Devroye et al. (1996) for results and details in the classification context, and to Györfi et al. (2002) for the regression framework considered in the present paper. Accordingly, we adhere as much as possible to their notations.

Let us denote $\mathcal{N}_k(\mathbf{x})$ the set of the $k$ nearest neighbors of $\mathbf{x}$ in $\mathcal{D}_n$, $\mathcal{N}'_k(\mathbf{X}_i)$ the set of the $k$ nearest neighbors of $\mathbf{X}_i$ in $(\mathcal{D}_n \setminus \{\mathbf{X}_i\}) \cup \{\mathbf{x}\}$, and

$$\mathcal{M}_k(\mathbf{x}) = \left\{\mathbf{X}_i \in \mathcal{N}_k(\mathbf{x}) : \ \mathbf{x} \in \mathcal{N}'_k(\mathbf{X}_i)\right\},$$

the set of the Mutual Nearest Neighbors (MNN) of $\mathbf{x}$. Denoting $M_k(\mathbf{x}) = |\mathcal{M}_k(\mathbf{x})|$ the number of mutual nearest neighbors of $\mathbf{x}$, $M_k(\mathbf{x})$ is a random variable taking values between 0 and $k$. The mutual nearest neighbors regression estimate is then defined as follows

$$m_n(\mathbf{x}) = \frac{1}{M_k(\mathbf{x})} \sum_{i:X_i \in \mathcal{M}_k(\mathbf{x})} Y_i,$$

with the convention that $0/0 = 0$. Two remarks are in order. First, contrarily to the $k$-NN estimate, the MNN estimate is symmetric. This means that, when averaging over the neighbors of $\mathbf{x}$ in the sample $\mathcal{D}_n$, we only consider the points for which $\mathbf{x}$ is itself one of the $k$ nearest neighbors.

The second remark is that, compared to the standard $k$NN rule, there might be an additional computational cost for applying the MNN procedure. Specifically, we might consider two different situations. In the first one, it is possible to precompute and sort the distances between all couples of points $(\mathbf{X}_i, \mathbf{X}_j)$ in the sample $\mathcal{D}_n$. Since the cost of computing the distance between a pair of $d$-dimensional vectors is $O(d)$, and that there are $n(n-1)/2$ such pairs in $\mathcal{D}_n$, and considering that the (quick)sorting of a vector of size $n$ is $O(n \log n)$, the cost of this precomputation is $O((d + \log n)n^2)$. In this case, after computing and sorting the pairwise distances, the computational burden of MNN and $k$NN are of the same order. Indeed, for a new point $\mathbf{x}$, computing the distances to the $\mathbf{X}_i$'s and finding the $k$ nearest neighbors has a cost in $O((d + \log k)n)$. For the mutual nearest neighbors, for each of these $k$ nearest neighbors, one has also to see if $\mathbf{x}$ is one of its $k$ nearest neighbors, hence an additive cost in $O(k)$. In the second situation, the cost of precomputation is prohibitively expensive, typically due to large sample size $n$ and high dimension $d$ of the covariates. In this case, the algorithmic cost for the $k$NN rule is of course the same as before, that is in $O((d + \log k)n)$, while the cost for the MNN rule is $O((k+1)(d + \log k)n) = O(k(d + \log k)n)$.

The term of *mutual nearest neighbors* seems to date back to Chidananda Gowda and Krishna (1978, 1979) in the context of clustering. In the past few years, it has raised an increasing interest in image analysis for object retrieval (see for example Jégou et al. (2010) and Qin et al. (2011)) as well as for classification purposes (see Liu et al., 2010). Interestingly, the latter reports that experimental results show that, on standard data sets, the MNN estimates have better performances than standard nearest neighbors estimates as well as other widely used classification rules.

Without claiming that MNN estimates always outperform standard nearest neighbors estimates, a heuristic explanation for this better behavior in some situations is related to the existence of hubs in high dimensional data. Specifically, a hub is a point which appears in many more $k$NN lists than the others, making it very influential in $k$NN estimates. As explained in Radovanović et al. (2010), hubness is an aspect of the curse of dimensionality as increasing the dimensionality results in the emergence of hubs under widely applicable conditions. These authors have also conducted several simulations to show how the existence of "bad" hubs negatively affects the $k$NN classifier

(see Section 7.1.2 in Radovanović et al. 2010). In our context, the existence of hubs might not affect the performance of MNN estimates and one could even consider the MNN rule as a variant of the $k$NN rule which allows to automatically reduce the role of these hubs.

However, to the best of our knowledge, little if nothing is known about the theoretical properties of the mutual nearest neighbors estimator. Our goal in this paper is to investigate its statistical properties, focusing our attention on the regression viewpoint. In Section 2, we present strong and weak consistency results. In Section 3, we go one step further and show that the rate of convergence of this estimate is, in fact, optimal when $d \geq 2$. Since the parameter $k = k_n$ of the estimate with the optimal rate of convergence depends on the unknown distribution of $(\mathbf{X}, Y)$, especially on the smoothness of the regression function, we also present adaptive (i.e., data-dependent) choices for $k_n$ that preserve the minimax optimality of the estimate.

## 2. Consistency

To prove the consistency of the MNN estimator, we write

$$m_n(\mathbf{x}) = \sum_{i=1}^{n} W_i(\mathbf{x}, \mathbf{X}_1, \ldots, \mathbf{X}_n) Y_i = \sum_{i=1}^{n} W_i Y_i,$$

where the weights $W_i$ are non negative random variables defined by

$$W_i = \begin{cases} \frac{1}{M_k(\mathbf{x})} & \text{if } M_k(\mathbf{x}) > 0 \text{ and } \mathbf{X}_i \in \mathcal{M}_k(\mathbf{x}), \\ 0 & \text{otherwise.} \end{cases}$$

This representation brings the MNN estimator into the general framework of weighted nearest neighbors, as studied for example in Stone (1977). But, contrarily to the standard $k$NN estimator for which the weights are deterministically linked to the order statistics $\mathbf{X}_{(1,n)}(\mathbf{x}), \ldots, \mathbf{X}_{(n,n)}(\mathbf{x})$, notice that this is not the case in our situation.

Nonetheless, in order to control the random weights $W_i$, we will exploit the following observation: for all $\mathbf{X}_i$ in $\mathcal{N}_k'(\mathbf{x})$, we have the following assertion

$$\|\mathbf{X}_i - \mathbf{x}\| < \frac{d_{(k+1)}}{2} = \frac{\|\mathbf{X}_{(k+1,n)}(\mathbf{x}) - \mathbf{x}\|}{2} \quad \Rightarrow \quad \mathbf{X}_i \in \mathcal{M}_k(\mathbf{x}). \tag{1}$$

Indeed, if not, there would exist $k$ points $\tilde{\mathbf{X}}_1, \ldots, \tilde{\mathbf{X}}_k$, different from $\mathbf{X}_i$, and such that for all $j = 1, \ldots, k$,

$$\|\tilde{\mathbf{X}}_j - \mathbf{X}_i\| < \|\mathbf{X}_i - \mathbf{x}\| < \frac{d_{(k+1)}}{2}.$$

By the triangle inequality,

$$\|\tilde{\mathbf{X}}_j - \mathbf{x}\| < \|\mathbf{X}_i - \mathbf{x}\| + \|\tilde{\mathbf{X}}_j - \mathbf{X}_i\| < d_{(k+1)},$$

which implies that there are at least $(k+1)$ points in the open ball $S_{\mathbf{x}, d_{(k+1)}}$ centered at $\mathbf{x}$ of radius $d_{(k+1)}$, which contradicts the definition of $d_{(k+1)}$.

Accordingly, let us define the random variable $B$ as the number of nearest neighbors $\mathbf{X}_i$'s which belong to $S_{\mathbf{x}, d_{(k+1)}/2}$. Given $d_{(k+1)}$ and defining $p_k$ as

$$p_k = \frac{\mu(S_{\mathbf{x}, d_{(k+1)}/2})}{\mu(S_{\mathbf{x}, d_{(k+1)}})},$$

where $\mu$ stands for the law of $\mathbf{X}$, we will justify in the proof of Theorem 1 that $B$ has a binomial distribution with parameters $k$ and $p_k$. As a consequence, Assertion (1) reads as an inequality between random variables

$$M_k(\mathbf{x}) \geq B.$$

This latter remark is of crucial importance for showing the following consistency results as well as for establishing the rates of convergence of Section 3. We begin with a strong consistency result.

**Theorem 1** *Suppose that the distribution $\mu$ of $\mathbf{X}$ is absolutely continuous on $\mathbb{R}^d$, that $Y$ is bounded and that the regression function $m$ is $\mu$ almost everywhere continuous. If $k \to \infty$, $k/n \to 0$, and $k/\log n \to \infty$, then $m_n$ is strongly consistent, that is*

$$m_n(\mathbf{X}) - m(\mathbf{X}) \to 0,$$

*with probability one.*

The proof of Theorem 1 reveals that local convergence in probability holds without the assumption that $k/\log n \to \infty$. Indeed, for $\mu$ almost every $\mathbf{x}$ and for every $\varepsilon > 0$,

$$\mathbb{P}(|m_n(\mathbf{x}) - m(\mathbf{x})| > \varepsilon) \to 0,$$

when $n$ goes to infinity, provided that $k \to \infty$ and $k/n \to 0$. Since $Y$ is bounded, the weak (i.e., $L_2$) consistency of Theorem 2 below is just a straightforward consequence of the dominated convergence theorem.

Notice that a standard way to prove the weak consistency of weighted nearest neighbors rules is to check the five conditions of Stone's universal consistency theorem (see Stone, 1977, Theorem 1). As is often the case, one of them is in fact particularly hard to verify in our situation, namely that there exists $C \geq 1$ such that for any nonnegative Borel function $f$ on $\mathbb{R}^d$,

$$\mathbb{E}\left[\sum_{i=1}^{n} W_i f(\mathbf{X}_i)\right] \leq C\, \mathbb{E}\left[f(\mathbf{X})\right].$$

The additional constraints in Theorem 2 are sufficient and are in fact the same as for the layered nearest neighbor estimate studied in Biau and Devroye (2010), as well as for the affine invariant nearest neighbor estimate investigated in Biau et al. (2012).

**Theorem 2** *Suppose that the distribution $\mu$ of $\mathbf{X}$ is absolutely continuous on $\mathbb{R}^d$, that $Y$ is bounded and that the regression function $m$ is $\mu$ almost everywhere continuous. If $k \to \infty$ and $k/n \to 0$, then $m_n$ is weakly consistent, that is*

$$\mathbb{E}[(m_n(\mathbf{X}) - m(\mathbf{X}))^2] \to 0.$$

We may lighten the assumption that $\mathbf{X}$ has a density. Indeed, an inspection of the proof of Theorem 1 indicates that consistency holds as long as for $\mu$ almost every $\mathbf{x}$,

$$\liminf_{h \to 0} \frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} > 0.$$

Interestingly, this condition is linked to the notion of *doubling measure* in geometric measure theory. We refer the interested reader to the monographs of Ambrosio and Tilli (2004), Heinonen (2001), and to the paper of Ambrosio et al. (2004).

Recall that the support $\mathcal{S}(\mu)$ is defined as the collection of all $\mathbf{x}$ with $\mu(S_{\mathbf{x},h}) > 0$ for all $h > 0$. In our context, a probability measure $\mu$ is said to be doubling on its support $\mathcal{S}(\mu)$ equipped with the Euclidean norm if there exists a constant $c > 0$ such that, for every $\mathbf{x}$ in $\mathcal{S}(\mu)$,

$$\frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} > c, \tag{2}$$

and $\mu$ is said to be asymptotically doubling if, for every $\mathbf{x}$ in $\mathcal{S}(\mu)$,

$$\liminf_{h\to 0} \frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} > 0.$$

Thus, we can relax the condition of Theorems 1 and 2 to only requiring that the probability measure $\mu$ is asymptotically doubling almost surely. To see that this condition is weaker than requiring a density, note that if $\mu$ admits the density $f$, then a consequence of Lebesgue's differentiation Theorem (see for example Theorem A.10 in Devroye et al. (1996)) is that for $\mu$-almost every $\mathbf{x}$ in $\mathcal{S}(\mu)$,

$$\frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} = \frac{\int_{S_{\mathbf{x},h/2}} f(\mathbf{u})\mathrm{d}\mathbf{u}}{\int_{S_{\mathbf{x},h}} f(\mathbf{u})\mathrm{d}\mathbf{u}} \to \frac{1}{2^d},$$

when $h$ tends to 0. Hence $\mu$ is asymptotically doubling almost surely.

It is also readily seen that any discrete probability measure is asymptotically doubling almost surely. Singular continuous probability measures can also be asymptotically doubling as is seen on the following example. Consider the uniform distribution on the standard Cantor ternary set $\mathcal{C}$. Recall that the uniform probability measure $\mu$ on $\mathcal{C}$ is the weak limit of the uniform probability measures $\mu_N$ on the sets $\mathcal{C}_N$ defined for every integer $N$ as the union of $2^N$ disjoint intervals with common length $3^{-N}$. It is easy to see that for every integer $N$ and for every $\mathbf{x}$ in $\mathcal{C}_N$,

$$\frac{1}{2} \leq \frac{\mu_N(S_{\mathbf{x},h/2})}{\mu_N(S_{\mathbf{x},h})} \leq 1.$$

Hence, for every $\mathbf{x}$ in $\mathcal{C}$,

$$\liminf_{h\to 0} \frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} \geq \frac{1}{2},$$

and $\mu$ is asymptotically doubling almost surely.

Next, we give an example of a singular continuous distribution that is non-asymptotically doubling with probability one. Given a sequence $(U_N)$ of independent Bernoulli distributed random variables with respective parameters $N/(N+1)$, which means that for all $N \geq 1$, $\mathbb{P}(U_N = 1) = N/(N+1)$, define the random variable

$$\mathbf{X} = \sum_{N=1}^{\infty} \frac{2U_N}{3^N}.$$

Note that $\mathbf{X}$ takes values in the standard Cantor ternary set $\mathcal{C}$, but that the law $\mu$ of $\mathbf{X}$ is not the uniform law on it: obviously, in the triadic expansion of $\mathbf{X}$, the 2's are much more likely than the

0's. Nevertheless, a direct application of Borel-Cantelli Lemma ensures that $\mu$ almost surely, there is an infinite number of 0's in the triadic expansion of $\mathbf{X}$. For such an $\mathbf{x} = \sum_{N=1}^{\infty} 2u_N/3^N$, consider the infinite set of indices

$$I_{\mathbf{x}} = \{N \geq 1 : \ u_N = 0\},$$

and denote $\mu_N$ the restriction of $\mu$ to the set $C_N$ defined as above, that is, the union of $2^N$ disjoint intervals with common length $3^{-N}$. Then, by construction, for each $N$ in $I_{\mathbf{x}}$, there exists an $h = h_N(\mathbf{x}) \in [1/3^N, 2/3^N]$ such that

$$\frac{\mu_N(S_{\mathbf{x},h/2})}{\mu_N(S_{\mathbf{x},h})} = \frac{1}{N} \ \Rightarrow \ \liminf_{h\to 0} \frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} = 0.$$

Consequently, $\mu$ is almost surely not asymptotically doubling. However, even on this pathological probability space, it is not obvious that we can define a regression function $m$ and a distribution for $Y$ such that the mutual nearest neighbors rule would fail to be consistent.

To conclude this section, let us finally notice that, in the context of adaptation to local intrinsic dimension of $k$NN regression, similar ideas related to the doubling property also appear in a recent paper by Kpotufe (2011).

## 3. Rates of Convergence

In this section, we are interested in rate of convergence results for the class $\mathcal{F}$ of $(1, C, \rho, \sigma^2)$-smooth distributions $(\mathbf{X}, Y)$ such that $\mathbf{X}$ has compact support with diameter $2\rho$, the regression function $m$ is Lipschitz with constant $C$ and, for all $\mathbf{x} \in \mathbb{R}^d$, $\mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2 < \infty$ (the symbol $\mathbb{V}$ denotes variance).

It is known (see, for example, Ibragimov and Khasminskii 1980, 1981, 1982, Stone 1980, 1982, or Györfi et al. 2002) that for the class $\mathcal{F}$, the optimal minimax rate of convergence is $n^{-2/(d+2)}$. In particular, one has that

$$\liminf_{n\to\infty} \inf_{\hat{m}_n} \sup_{(\mathbf{X},Y)\in\mathcal{F}} \frac{\mathbb{E}[\hat{m}_n(\mathbf{X}) - m(\mathbf{X})]^2}{\left((\rho C)^d \sigma^2\right)^{\frac{2}{d+2}} n^{-\frac{2}{d+2}}} \geq \Delta,$$

for some positive constant $\Delta$ independent of $C$, $\rho$ and $\sigma^2$. Here the infimum is taken over all estimates $\hat{m}_n$, that is, over all measurable functions of the data.

It turns out that, for $d \geq 2$ and a suitable choice of the sequence $(k_n)$, the MNN estimate $m_n$ achieves the optimum rate for the class $\mathcal{F}$, that is

$$\limsup_{n\to\infty} \sup_{(\mathbf{X},Y)\in\mathcal{F}} \frac{\mathbb{E}[m_n(\mathbf{X}) - m(\mathbf{X})]^2}{\left((\rho C)^d \sigma^2\right)^{\frac{2}{d+2}} n^{-\frac{2}{d+2}}} \leq \Lambda,$$

for some positive $\Lambda$ independent of $C$, $\rho$ and $\sigma^2$.

Before precisely stating this result, we need an additional notation. Let $\mu$ be a probability measure on $\mathbb{R}^d$ with compact support $\mathcal{S}(\mu)$ with diameter $2\rho$. We will assume that $\mu$ is doubling, as defined in (2), and let

$$p = \inf_{(\mathbf{x},h)\in\mathcal{S}(\mu)\times(0,2\rho]} \frac{\mu(S_{\mathbf{x},h/2})}{\mu(S_{\mathbf{x},h})} > 0. \tag{3}$$

It is readily seen that if $\mu$ is absolutely continuous with density $f$, a sufficient condition is that there exist two strictly positive real numbers $a$ and $A$ such that for almost every $\mathbf{x}$ in $\mathcal{S}(\mu)$, we have $a \leq f(\mathbf{x}) \leq A$.

**Theorem 3** *Assume that ties occur with probability 0. Suppose that the law $\mu$ of $\mathbf{X}$ has a compact support $\mathcal{S}(\mu)$ with diameter $2\rho$, and that $\mu$ is doubling, with $p$ defined as in (3). Suppose in addition that, for all $\mathbf{x}$ and $\mathbf{x}' \in \mathbb{R}^d$,*

$$\sigma^2(\mathbf{x}) = \mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2,$$

*and*

$$\left|m(\mathbf{x}) - m(\mathbf{x}')\right| \leq C\|\mathbf{x} - \mathbf{x}'\|,$$

*for some positive constants $\sigma^2$ and $C$. Denote by $L_m$ an upper-bound of the continuous mapping $m$ on the compact $\mathcal{S}(\mu)$. Then*

(*i*) *If $d = 1$,*

$$\mathbb{E}\left[m_n(\mathbf{X}) - m(\mathbf{X})\right]^2 \leq \frac{2\sigma^2}{kp} + \frac{16\rho^2 C^2 k}{n} + L_m^2(1-p)^k.$$

(*ii*) *If $d = 2$,*

$$\mathbb{E}\left[m_n(\mathbf{X}) - m(\mathbf{X})\right]^2 \leq \frac{2\sigma^2}{kp} + \frac{32\rho^2 C^2}{n} + L_m^2(1-p)^k.$$

(*iii*) *If $d \geq 3$,*

$$\mathbb{E}\left[m_n(\mathbf{X}) - m(\mathbf{X})\right]^2 \leq \frac{2\sigma^2}{kp} + \frac{8\rho^2 C^2 \lfloor n/k \rfloor^{-2/d}}{1 - 2/d} + L_m^2(1-p)^k.$$

By balancing the terms in Theorem 3, we are led to the following corollary:

**Corollary 1** *Under the assumptions of Theorem 3,*

(*i*) *If $d = 1$, there exists a sequence $(k_n)$ with $k_n \propto \sqrt{n}$ such that*

$$\mathbb{E}\left[m_n(\mathbf{X}) - m(\mathbf{X})\right]^2 \leq (\Lambda + o(1))\frac{\rho C \sigma}{\sqrt{n}},$$

*for some positive constant $\Lambda$ independent of $\rho$, $C$ and $\sigma^2$.*

(*ii*) *If $d \geq 2$, there exists a sequence $(k_n)$ with $k_n \propto n^{\frac{2}{d+2}}$ such that*

$$\mathbb{E}\left[m_n(\mathbf{X}) - m(\mathbf{X})\right]^2 \leq (\Lambda + o(1))\left(\frac{(\rho C)^d \sigma^2}{n}\right)^{\frac{2}{d+2}},$$

*for some positive constant $\Lambda$ independent of $\rho$, $C$ and $\sigma^2$.*

Two remarks are in order.

1. We note that, for $d \geq 2$ and a suitable choice of $k_n$, the MNN estimate achieves both the minimax $n^{-2/(d+2)}$ rate and the optimal order of magnitude $((\rho C)^d \sigma^2)^{2/(d+2)}$ in the constant, for the class $\mathcal{F}$ of $(1, C, \rho, \sigma^2)$-smooth distributions $(\mathbf{X}, Y)$ such that $\mathbf{X}$ has compact support with covering radius $\rho$, the regression function $m$ is Lipschitz with constant $C$ and, for all $\mathbf{x} \in \mathbb{R}^d$, $\mathbb{V}[Y \,|\, \mathbf{X} = \mathbf{x}] \leq \sigma^2$.

2. For $d = 1$, the obtained rate is not optimal. This low-dimensional phenomenon is also known to hold for the traditional $k$NN regression estimate, which does not achieve the optimal rate in dimension 1 (see Problem 6.1 in Györfi et al. 2002).

In Corollary 1, the parameter $k_n$ of the estimate with the optimal rate of convergence for the class $\mathcal{F}$ depends on the unknown distribution of $(\mathbf{X},Y)$, especially on the smoothness of the regression function as measured by the Lipschitz constant $C$. To conclude this section, we present a data-dependent way for choosing the resampling size $k_n$ and show that, for bounded $Y$, the estimate with parameter chosen in such an adaptive way achieves the optimal rate of convergence.

To this end, we split the sample $\mathcal{D}_n = \{(\mathbf{X}_1,Y_1),\ldots,(\mathbf{X}_n,Y_n)\}$ in two parts, denoted by $\mathcal{D}_n^\ell$ (learning set) and $\mathcal{D}_n^t$ (testing set), of size $\lfloor n/2 \rfloor$ and $n - \lfloor n/2 \rfloor$, respectively. The first half is used to construct the MNN estimate

$$m_{\lfloor n/2 \rfloor}(\mathbf{x}, \mathcal{D}_n^\ell) = m_{k,\lfloor n/2 \rfloor}(\mathbf{x}, \mathcal{D}_n^\ell).$$

The second half is used to choose $k$ by picking $\hat{k}_n \in \mathcal{K} = \{1,\ldots,\lfloor n/2 \rfloor\}$ to minimize the empirical risk

$$\frac{1}{n - \lfloor n/2 \rfloor} \sum_{i=\lfloor n/2 \rfloor + 1}^{n} \left( Y_i - m_{k,\lfloor n/2 \rfloor}(\mathbf{X}_i, \mathcal{D}_n^\ell) \right)^2.$$

Define the estimate

$$m_n(\mathbf{x}) = m_{\hat{k}_n,\lfloor n/2 \rfloor}(\mathbf{x}, \mathcal{D}_n^\ell),$$

and note that $m_n$ depends on the entire data $\mathcal{D}_n$. If $|Y| \leq L < \infty$ almost surely, a straightforward adaptation of Theorem 7.1 in Györfi et al. (2002) shows that, for any $\delta > 0$,

$$\mathbb{E}[m_n(\mathbf{X}) - m(\mathbf{X})]^2 \leq (1+\delta) \inf_{k \in \mathcal{K}} \mathbb{E}[m_{k,\lfloor n/2 \rfloor}(\mathbf{X}, \mathcal{D}_n^\ell) - m(\mathbf{X})]^2 + \Xi \frac{\ln n}{n},$$

for some positive constant $\Xi$ depending only on $L$, $d$ and $\delta$. Immediately from Corollary 1, we can conclude:

**Theorem 4** *Suppose that $|Y| \leq L$ almost surely, and let $m_n$ be the MNN estimate with $k \in \mathcal{K} = \{1,\ldots,\lfloor n/2 \rfloor\}$ chosen by data-splitting. Then the condition $(\ln n)^{(d+2)/(2d)} n^{-1/2} \leq \rho C$ together with $d \geq 2$ implies*

$$\mathbb{E}[m_n(\mathbf{X}) - m(\mathbf{X})]^2 \leq (\Lambda + o(1)) \left( \frac{(\rho C)^d}{n} \right)^{\frac{2}{d+2}},$$

*for some positive constant $\Lambda$ which depends only on $L$ and $d$.*

Thus, the expected error of the estimate obtained via data-splitting is bounded from above up to a constant by the corresponding minimax lower bound for the class $\mathcal{F}$ of regression functions, with the optimal dependence in $C$ and $\rho$.

## 4. Proofs

Proofs of the main results are gathered in this section.

### 4.1 Proof of Theorem 1

Let us fix $\varepsilon > 0$ and $\mathbf{x}$ in $\mathcal{S}(\mu)$ such that $m$ is continuous at $\mathbf{x}$. Setting

$$\tilde{m}_n(\mathbf{x}) = \sum_{i=1}^{n} W_i \, m(\mathbf{X}_i),$$

we have

$$\begin{aligned}\mathbb{P}\left(|m_n(\mathbf{x}) - m(\mathbf{x})| > 2\varepsilon\right) \leq\ & \mathbb{P}\left(M_k(x) < \frac{k}{2^{d+1}}\right) \\ & + \mathbb{P}\left(|m_n(\mathbf{x}) - \tilde{m}_n(\mathbf{x})| > \varepsilon, M_k(x) \geq \frac{k}{2^{d+1}}\right) \\ & + \mathbb{P}\left(|\tilde{m}_n(\mathbf{x}) - m(\mathbf{x})| > \varepsilon, M_k(x) \geq \frac{k}{2^{d+1}}\right). \end{aligned} \tag{4}$$

First, remark that rearranging the $k$ (ordered) statistics $\mathbf{X}_{(1,n)}, \ldots, \mathbf{X}_{(k,n)}$ in the original order of their outcome, one obtains the $k$ (non-ordered) random variables $\mathbf{X}_1^\star, \ldots, \mathbf{X}_k^\star$. Let $\tilde{\mathbf{X}}_1, \ldots, \tilde{\mathbf{X}}_k$ be i.i.d. random variables, with common law (conditional on $d_{(k+1)}$) the restriction $\tilde{\mu}$ of $\mu$ to the open ball $S_{\mathbf{x}, d_{(k+1)}}$, then it can be shown (see for example Lemma A.1 in Cérou and Guyader 2006) that

$$\mathcal{L}(\mathbf{X}_1^\star, \ldots, \mathbf{X}_k^\star | d_{(k+1)}) = \mathcal{L}(\tilde{\mathbf{X}}_1, \ldots, \tilde{\mathbf{X}}_k). \tag{5}$$

Next, given $d_{(k+1)}$, denote

$$p_k = \mathbb{P}\left(\|\tilde{\mathbf{X}} - \mathbf{x}\| < \frac{d_{(k+1)}}{2} \middle| \tilde{\mathbf{X}} \sim \tilde{\mu}\right) = \frac{\int_{S_{\mathbf{x}, d_{(k+1)}/2}} f(\mathbf{u}) \mathrm{d}\mathbf{u}}{\int_{S_{\mathbf{x}, d_{(k+1)}}} f(\mathbf{u}) \mathrm{d}\mathbf{u}},$$

where the denominator is strictly positive since $\mathbf{x}$ belongs to the support of $\mu$. Concerning $p_k$, recall that Lebesgue's differentiation Theorem ensures that for $\lambda$-almost all $\mathbf{x} \in \mathbb{R}^d$,

$$\frac{1}{\lambda(S_{x,\delta})} \int_{S_{x,\delta}} f(\mathbf{u}) \mathrm{d}\mathbf{u} \to f(\mathbf{x}),$$

when $\delta$ tends to 0 (see for example Theorem A.10 in Devroye et al. 1996). Notice that for $\mu$ almost every $\mathbf{x}$ in the support of $\mu$, we have $f(\mathbf{x}) > 0$. Consequently, since $\lambda(S_{\mathbf{x},h}) = V_d h^d$ with $V_d$ the volume of the unit ball of $\mathbb{R}^d$, we have that for $\mu$-almost every $\mathbf{x}$ in $\mathbb{R}^d$,

$$p(\delta) := \frac{\int_{S_{\mathbf{x},\delta/2}} f(\mathbf{u}) \mathrm{d}\mathbf{u}}{\int_{S_{\mathbf{x},\delta}} f(\mathbf{u}) \mathrm{d}\mathbf{u}} \to \frac{1}{2^d}, \tag{6}$$

when $h$ tends to 0. Hence, let us choose $\delta_0 > 0$ such that

$$\delta \in (0, \delta_0] \ \Rightarrow \left| p(\delta) - \frac{1}{2^d} \right| < \frac{1}{2^{d+2}}.$$

Then we may write

$$\mathbb{P}\left(M_k(x) < \frac{k}{2^{d+1}}\right) \leq \mathbb{P}\left(M_k(x) < \frac{k}{2^{d+1}}, d_{(k+1)} \leq \delta_0\right) + \mathbb{P}(d_{(k+1)} > \delta_0).$$

Denoting

$$q_0 = \mathbb{P}(\|\mathbf{X}-\mathbf{x}\| \leq \delta_0) = \int_{S_{\mathbf{x},\delta_0}} f(\mathbf{u})\mathrm{d}\mathbf{u},$$

we have $q_0 > 0$. Following the proof of Lemma 4 in Devroye (1982), denote $Z$ a binomial $(n, q_0)$ random variable. If $k/n \to 0$, then for $n$ large enough, Hoeffding's inequality yields

$$\mathbb{P}(d_{(k+1)} > \delta_0) \leq \mathbb{P}(Z < k+1) \leq \mathbb{P}\left(Z - nq_0 < -\frac{nq_0}{2}\right) \leq e^{-nq_0^2/2},$$

which is summable in $n$ for all $\delta_0 > 0$. Next, observe that

$$\begin{aligned}\mathbb{P}&\left(M_k(x) < \frac{k}{2^{d+1}}, d_{(k+1)} \leq \delta_0\right) \\ &= \int_0^{\delta_0} \mathbb{P}\left(M_k(x) < \frac{k}{2^{d+1}} \middle| d_{(k+1)} = \delta\right) \mathrm{dP}_{d_{(k+1)}}(\delta).\end{aligned}$$

Given $\delta$ and defining $B$ as the number of $\mathbf{X}_i$'s among the $k$ nearest neighbors of $\mathbf{x}$ which belong to $S_{\mathbf{x},\delta/2}$, then according to (5), the random variable $B$ has binomial distribution $\mathcal{B}(k, p(\delta))$ and (1) implies $M_k(\mathbf{x}) \geq B$, so that

$$\mathbb{P}\left(M_k(\mathbf{x}) < \frac{k}{2^{d+1}} \middle| d_{(k+1)} = \delta\right) \leq \mathbb{P}\left(B < \frac{k}{2^{d+1}} \middle| p(\delta)\right).$$

In this respect, Hoeffding's inequality and (6) lead to

$$\mathbb{P}\left(B < \frac{k}{2^{d+1}} \middle| p(\delta)\right) \leq \exp\left(-2\left(p(\delta) - \frac{1}{2^{d+1}}\right)^2 k\right) \leq \exp\left(-\frac{k}{2^{2d+3}}\right),$$

which is summable in $n$ provided that $k/\log n \to \infty$.

Let us turn now to second term of (4). This time, we write

$$\begin{aligned}\mathbb{P}&\left(|m_n(\mathbf{x}) - \tilde{m}_n(\mathbf{x})| > \varepsilon, M_k(\mathbf{x}) \geq \frac{k}{2^{d+1}}\right) \\ &= \mathbb{E}\left[\mathbf{1}_{\{M_k(\mathbf{x}) \geq \frac{k}{2^{d+1}}\}} \mathbb{P}\left(|m_n(\mathbf{x}) - \tilde{m}_n(\mathbf{x})| > \varepsilon | \mathbf{X}_1, \ldots, \mathbf{X}_n\right)\right] \\ &= \mathbb{E}\left[\mathbf{1}_{\{M_k(\mathbf{x}) \geq \frac{k}{2^{d+1}}\}} \mathbb{P}\left(\left|\sum_{i=1}^n W_i(Y_i - m(\mathbf{X}_i))\right| > \varepsilon \middle| \mathbf{X}_1, \ldots, \mathbf{X}_n\right)\right].\end{aligned}$$

Given $\mathbf{X}_1, \ldots, \mathbf{X}_n$, the random variables $Y_1 - m(\mathbf{X}_1), \ldots, Y_n - m(\mathbf{X}_n)$ are independent, centered, and bounded by $2L$. Moreover, the weights $W_1, \ldots, W_n$ are deterministic and, since $M_k(\mathbf{x}) \geq k/2^{d+1}$, bounded by $2^{d+1}/k$. Consequently, Lemma 6 in Devroye (1982) leads to

$$\mathbb{P}\left(|m_n(\mathbf{x}) - \tilde{m}_n(\mathbf{x})| > \varepsilon, M_k(\mathbf{x}) \geq \frac{k}{2^{d+1}}\right) \leq 2\exp\left(-\frac{k\varepsilon^2}{2^{d+3}(2L^2 + L\varepsilon)}\right),$$

which is summable in $n$ for all $\varepsilon > 0$, provided that $k/\log n \to \infty$.

The last term of (4) is easier. First we notice that, since $m$ is assumed continuous at point $\mathbf{x}$, there exists $\delta_1 = \delta_1(\varepsilon)$ such that

$$\|\mathbf{x}' - \mathbf{x}\| \leq \delta_1 \;\Rightarrow \left|m(\mathbf{x}') - m(\mathbf{x})\right| \leq \varepsilon.$$

The following inequalities are then straightforward

$$\begin{aligned}
&\mathbb{P}\left(|\tilde{m}_n(\mathbf{x}) - m(\mathbf{x})| > \varepsilon, M_k(x) \geq \frac{k}{2^{d+1}}\right) \\
&\quad = \mathbb{P}\left(\left|\sum_{i=1}^{n} W_i(m(\mathbf{X}_i) - m(\mathbf{x}))\right| > \varepsilon, M_k(x) \geq \frac{k}{2^{d+1}}\right) \\
&\quad \leq \mathbb{P}\left(\max_{1\leq i\leq k} \left|m(\mathbf{X}_{(i)}) - m(\mathbf{x})\right| > \varepsilon\right) \\
&\quad \leq \mathbb{P}\left(\left\|\mathbf{X}_{(k)} - \mathbf{x}\right\| > \delta_1\right),
\end{aligned}$$

and the same reasoning as before yields

$$\mathbb{P}\left(|\tilde{m}_n(\mathbf{x}) - m(\mathbf{x})| > \varepsilon, M_k(x) \geq \frac{k}{2^{d+1}}\right) \leq e^{-nq_1^2/2},$$

where

$$q_1 = q_1(\varepsilon) = \mathbb{P}(\|\mathbf{X} - \mathbf{x}\| \leq \delta_1) = \int_{S_{\mathbf{x},\delta_1}} f(\mathbf{u})\mathrm{d}\mathbf{u}.$$

Putting all things together, we have proved that for any $\varepsilon > 0$, if $k/n \to 0$, then for $n$ large enough we have

$$\begin{aligned}
&\mathbb{P}\left(|m_n(\mathbf{x}) - m(\mathbf{x})| > 2\varepsilon\right) \\
&\leq 2\exp\left(\frac{-k\varepsilon^2}{2^{d+3}(2L^2 + L\varepsilon)}\right) + \exp\left(\frac{-k}{2^{2d+3}}\right) + \exp\left(\frac{-nq_0^2}{2}\right) + \exp\left(\frac{-nq_1^2}{2}\right),
\end{aligned}$$

which is summable in $n$ for all $\varepsilon > 0$, provided that $k/\log n \to \infty$. Since this is true for $\mu$ almost every $\mathbf{x}$, the strong consistency is established.

### 4.2 Proof of Theorem 3

As previously, setting

$$\tilde{m}_n(\mathbf{x}) = \sum_{i=1}^{n} W_i\, m(\mathbf{X}_i),$$

the proof of Theorem 3 will rely on the variance/bias decomposition

$$\mathbb{E}\left[m_n(\mathbf{x}) - m(\mathbf{x})\right]^2 = \mathbb{E}\left[m_n(\mathbf{x}) - \tilde{m}_n(\mathbf{x})\right]^2 + \mathbb{E}\left[\tilde{m}_n(\mathbf{x}) - m(\mathbf{x})\right]^2. \tag{7}$$

The first term is easily bounded by noting that, for all $\mathbf{x}$ in $\mathbb{R}^d$,

$$\begin{aligned}
\mathbb{E}\left[m_n(\mathbf{x})-\tilde{m}_n(\mathbf{x})\right]^2 &= \mathbb{E}\left[\sum_{i=1}^n W_i\left(Y_i-m(\mathbf{X}_i)\right)\right]^2 \\
&= \mathbb{E}\left[\sum_{i=1}^n W_i^2\left(Y_i-m(\mathbf{X}_i)\right)^2\right] \\
&= \mathbb{E}\left[\sum_{i=1}^n W_i^2\mathbb{E}\left[\left(Y_i-m(\mathbf{X}_i)\right)^2\middle|\mathbf{X}_i\right]\right] \\
&= \mathbb{E}\left[\sum_{i=1}^n W_i^2\sigma^2\left(\mathbf{X}_i\right)\right] \\
&\leq \sigma^2\,\mathbb{E}\left[\sum_{i=1}^n W_i^2\right].
\end{aligned}$$

With the convention that 0/0=0, notice that by definition of the weights $W_i$,

$$\mathbb{E}\left[\sum_{i=1}^n W_i^2\right] = \mathbb{E}\left[\frac{1}{M_k(\mathbf{X})}\mathbf{1}_{M_k(\mathbf{X})\neq 0}\right].$$

As in the proof of Theorem 1, given $d_{(k+1)}$, denote

$$p_k = \mathbb{P}\left(\|\tilde{\mathbf{X}}-\mathbf{x}\| < \frac{d_{(k+1)}}{2}\middle|\,\tilde{\mathbf{X}}\sim\tilde{\mu}\right) = \frac{\mu\left(S_{\mathbf{x},d_{(k+1)}/2}\right)}{\mu\left(S_{\mathbf{x},d_{(k+1)}}\right)},$$

and define $B$ as the number of $\mathbf{X}_i$'s among the $k$ nearest neighbors of $\mathbf{x}$ which belong to $S_{\mathbf{x},d_{(k+1)}/2}$. Then, given $p_k$, the random variable $B$ has binomial distribution $\mathcal{B}(k,p_k)$, and (1) implies

$$M_k(\mathbf{x}) \geq B. \tag{8}$$

In particular,

$$\frac{1}{M_k(\mathbf{x})}\mathbf{1}_{M_k(\mathbf{X})\neq 0} \leq \frac{2}{1+M_k(\mathbf{x})} \leq \frac{2}{1+B},$$

so that

$$\mathbb{E}\left[\sum_{i=1}^n W_i^2\right] \leq \mathbb{E}\left[\mathbb{E}\left[\frac{2}{1+B}\middle|\,p_k\right]\right] = 2\mathbb{E}\left[\frac{1-(1-p_k)^k}{(k+1)p_k}\right].$$

Since $p_k \geq p$, we are led to

$$\mathbb{E}\left[m_n(\mathbf{x})-\tilde{m}_n(\mathbf{x})\right]^2 \leq \frac{2\sigma^2}{kp},$$

and integrating with respect to the distribution of $\mathbf{X}$ yields

$$\mathbb{E}\left[m_n(\mathbf{X})-\tilde{m}_n(\mathbf{X})\right]^2 \leq \frac{2\sigma^2}{kp}.$$

Concerning the bias term in (7), again fix $\mathbf{x}$ in $\mathbb{R}^d$, denote by $L_m$ an upper-bound of the continuous function $m$ on the compact $\mathcal{S}(\mu)$, and write

$$\mathbb{E}\left[\tilde{m}_n(\mathbf{x}) - m(\mathbf{x})\right]^2 \leq \mathbb{E}\left[(\tilde{m}_n(\mathbf{x}) - m(\mathbf{x}))^2 \mathbf{1}_{\{M_k(\mathbf{x})>0\}}\right] + L_m^2\, \mathbb{P}(M_k(\mathbf{x}) = 0).$$

The second term is bounded thanks to (8),

$$\mathbb{P}(M_k(\mathbf{x}) = 0 | p_k) \leq \mathbb{P}(B = 0 | p_k) = (1 - p_k)^k,$$

so that

$$\mathbb{P}(M_k(\mathbf{x}) = 0) \leq \mathbb{E}\left[(1 - p_k)^k\right],$$

and since $p_k \geq p$,

$$\mathbb{P}(M_k(\mathbf{x}) = 0) \leq \mathbb{E}\left[(1 - p_k)^k\right] \leq (1 - p)^k.$$

For the first term, with the convention $0/0 = 0$, one has

$$\begin{aligned}
&\mathbb{E}\left[(\tilde{m}_n(\mathbf{x}) - m(\mathbf{x}))^2 \mathbf{1}_{\{M_k(\mathbf{x})>0\}}\right] \\
&\quad = \mathbb{E}\left[\left(\frac{1}{M_k(\mathbf{x})} \sum_{i:\mathbf{X}_i \in \mathcal{M}_k(\mathbf{x})} (m(\mathbf{X}_i) - m(\mathbf{x}))\right)^2 \mathbf{1}_{\{M_k(\mathbf{x})>0\}}\right] \\
&\quad \leq C^2 \mathbb{E}\left[\left(\frac{1}{M_k(\mathbf{x})} \sum_{i:\mathbf{X}_i \in \mathcal{M}_k(\mathbf{x})} \|\mathbf{X}_i - \mathbf{x}\|\right)^2 \mathbf{1}_{\{M_k(\mathbf{x})>0\}}\right].
\end{aligned}$$

Next we apply Jensen's inequality to get

$$\mathbb{E}\left[(\tilde{m}_n(\mathbf{x}) - m(\mathbf{x}))^2 \mathbf{1}_{\{M_k(\mathbf{x})>0\}}\right] \leq C^2 \mathbb{E}\left[\frac{\mathbf{1}_{\{M_k(\mathbf{x})>0\}}}{M_k(\mathbf{x})} \sum_{i:\mathbf{X}_i \in \mathcal{M}_k(\mathbf{x})} \|\mathbf{X}_i - \mathbf{x}\|^2\right].$$

Since any mutual nearest neighbor of $\mathbf{x}$ belongs to its $k$ nearest neighbors, we deduce

$$\mathbb{E}\left[(\tilde{m}_n(\mathbf{x}) - m(\mathbf{x}))^2 \mathbf{1}_{\{M_k(\mathbf{x})>0\}}\right] \leq C^2 \mathbb{E}\left[\left\|\mathbf{X}_{(k,n)} - \mathbf{x}\right\|^2\right].$$

Therefore, by integrating with respect to the distribution of $\mathbf{X}$, we obtain the following upper-bound for the bias term

$$\mathbb{E}\left[\tilde{m}_n(\mathbf{X}) - m(\mathbf{X})\right]^2 \leq C^2 \mathbb{E}\left[\|\mathbf{X}_{(k,n)} - \mathbf{X}\|^2\right] + L_m^2 (1 - p)^k.$$

Next, let us denote

$$\rho = \inf\left\{r > 0 \ : \ \exists\, \mathbf{x}_0 \in \mathbb{R}^d \text{ such that } \mathcal{S}(\mu) \subset S_{\mathbf{x}_0, r}\right\},$$

and notice that $2\rho$ is an upper-bound of the diameter of $\mathcal{S}(\mu)$. Then we are in a position to apply Proposition 2.3 in Biau et al. (2010), that is for $d = 1$,

$$\mathbb{E}\left[\|\mathbf{X}_{(k,n)} - \mathbf{X}\|^2\right] \leq \frac{16\rho^2 k}{n},$$

and for $d \geq 3$,

$$\mathbb{E}\left[\|\mathbf{X}_{(k,n)} - \mathbf{X}\|^2\right] \leq \frac{8\rho^2 \lfloor n/k \rfloor^{-\frac{2}{d}}}{1-2/d}.$$

It turns out that, for $d = 2$, the bound given in Biau et al. (2010) is not optimal, since it leads to

$$\mathbb{E}\left[\|\mathbf{X}_{(k,n)} - \mathbf{X}\|^2\right] \leq \frac{8\rho^2 k}{n}\left(1 + \log\frac{n}{k}\right),$$

whereas Theorem 3.2 in Liitiäinen et al. (2010) allows to get rid of the logarithmic term. Namely, the application of their result in our context leads to

$$\mathbb{E}\left[\|\mathbf{X}_{(k,n)} - \mathbf{X}\|^2\right] \leq \frac{32\rho^2 k}{n}.$$

This terminates the proof of Theorem 3.

## Acknowledgments

Arnaud Guyader wish to thank Hervé Jégou to have made him aware of the mutual nearest neighbors rule, and Frédéric Cérou for helpful discussions on the second version of the article. The authors are greatly indebted to Luigi Ambrosio and Giovanni Alberti for fruitful discussions on doubling measure theory. They also thank three referees and the Action Editor for their thoughtful critical comments.

## References

L. Ambrosio and P. Tilli. *Topics on Analysis in Metric Spaces*. Oxford University Press, Oxford, 2004.

L. Ambrosio, M. Miranda, Jr., and D. Pallara. Special functions of bounded variation in doubling metric measure spaces. In *Calculus of Variations: Topics from the Mathematical Heritage of E. De Giorgi*, volume 14 of *Quad. Mat.*, pages 1–45. Dept. Math., Seconda Univ. Napoli, Caserta, 2004.

G. Biau and L. Devroye. On the layered nearest neighbour estimate, the bagged nearest neighbour estimate and the random forest method in regression and classification. *Journal of Multivariate Analysis*, 101(10):2499–2518, 2010.

G. Biau, F. Cérou, and A. Guyader. On the rate of convergence of the bagged nearest neighbor estimate. *Journal of Machine Learning Research (JMLR)*, 11:687–712, 2010.

G. Biau, L. Devroye, V. Dujmović, and A. Krzyżak. An affine invariant $k$-nearest neighbor regression estimate. *Journal of Multivariate Analysis*, 112:24–34, 2012.

F. Cérou and A. Guyader. Nearest neighbor classification in infinite dimension. *ESAIM: Probability and Statistics*, 10:340–355, 2006.

K. Chidananda Gowda and G. Krishna. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern Recognition*, 10(2):105–112, 1978.

K. Chidananda Gowda and G. Krishna. The condensed nearest neighbor rule using the concept of mutual nearest neighborhood. *IEEE Transactions on Information Theory*, 25(4), 1979.

L. Devroye. Necessary and sufficient conditions for the pointwise convergence of nearest neighbor regression function estimates. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 61(4):467–481, 1982.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, 1996.

E. Fix and J.L. Hodges. Discriminatory analysis, non-parametric discrimination: consistency properties. Technical report, USAF school of aviation and medicine, Randolph Field, 1951.

E. Fix and J.L. Hodges. Discriminatory analysis: Small sample performance. Technical report, USAF school of aviation and medicine, Randolph Field, 1952.

L. Györfi, M. Kohler, A. Krzyżak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer-Verlag, New York, 2002.

J. Heinonen. *Lectures on Analysis on Metric Spaces*. Universitext. Springer-Verlag, New York, 2001.

I.A. Ibragimov and R.Z. Khasminskii. Nonparametric regression estimation. *Doklady Akademii Nauk SSSR*, 252(4):780–784, 1980.

I.A. Ibragimov and R.Z. Khasminskii. *Statistical Estimation: Asymptotic Theory*. Springer-Verlag, New York, 1981.

I.A. Ibragimov and R.Z. Khasminskii. Bounds for the quality of nonparametric estimation of regression. *Akademiya Nauk SSSR. Teoriya Veroyatnosteĭ i ee Primeneniya*, 27(1):81–94, 1982.

H. Jégou, C. Schmid, H. Harzallah, and J. Verbeek. Accurate image search using the contextual dissimilarity measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1): 2–11, 2010.

S. Kpotufe. $k$-NN regression adapts to local intrinsic dimension. In *NIPS Proceedings*, pages 729–737, 2011.

E. Liitiäinen, F. Corona, and A. Lendasse. Residual variance estimation using a nearest neighbor statistic. *Journal of Multivariate Analysis*, 101(4):811–823, 2010.

H. Liu, S. Zhang, J. Zhao, X. Zhao, and Y. Mo. A new classification algorithm using mutual nearest neighbors. In *Conference on Grid and Cloud Computing*, pages 52–57, 2010.

D. Qin, S. Gammeter, L. Bossard, T. Quack, and L.J. Van Gool. Hello neighbor: accurate object retrieval with $k$-reciprocal nearest neighbors. In *Computer Vision and Pattern Recognition*, pages 777–784, 2011.

M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research (JMLR)*, 11:2487–2531, 2010.

C.J. Stone. Consistent nonparametric regression. *The Annals of Statistics*, 5(4):595–645, 1977.

C.J. Stone. Optimal rates of convergence for nonparametric estimators. *The Annals of Statistics*, 8 (6):1348–1360, 1980.

C.J. Stone. Optimal global rates of convergence for nonparametric regression. *The Annals of Statistics*, 10(4):1040–1053, 1982.

**Editor's note:**

The page numbering within volume 14 inadvertently skipped pages 2377-2414.

This page and its reverse intentionally serve as a notification of and placeholder for the skipped page numbers.

**Editor's note:**

The page numbering within volume 14 inadvertently skipped pages 2377-2414.

This page and its reverse intentionally serve as a notification of and placeholder for the skipped page numbers.

Journal of Machine Learning Research 14 (2013) 2415-2448 Submitted 1/12; Revised 1/13; Published 8/13

# Distance Preserving Embeddings for General $n$-Dimensional Manifolds

**Nakul Verma** NAVERMA@CS.UCSD.EDU
*Department of Computer Science and Engineering*
*University of California, San Diego*
*La Jolla, CA 92093-0404, USA*

**Editor:** Mikhail Belkin

## Abstract

Low dimensional embeddings of manifold data have gained popularity in the last decade. However, a systematic finite sample analysis of manifold embedding algorithms largely eludes researchers. Here we present two algorithms that embed a general $n$-dimensional manifold into $\mathbb{R}^d$ (where $d$ only depends on some key manifold properties such as its intrinsic dimension, volume and curvature) that *guarantee* to approximately preserve all interpoint geodesic distances.

**Keywords:** manifold learning, isometric embeddings, non-linear dimensionality reduction, Nash's embedding theorem

## 1. Introduction

Finding low dimensional representations of manifold data has gained popularity in the last decade. One typically assumes that points are sampled from an $n$-dimensional manifold residing in some high-dimensional ambient space $\mathbb{R}^D$ and analyzes to what extent their low dimensional embedding maintains some important manifold property, say, interpoint geodesic distances.

Despite an abundance of manifold embedding algorithms, only a few provide any kind of distance preserving guarantee. Isomap (Tenebaum et al., 2000), for instance, provides an asymptotic guarantee that as one increases the amount of data sampled from an underlying manifold, one can approximate the geodesic distances between the sample points well (Bernstein et al., 2000). Then, under a very restricted class of $n$-dimensional manifolds, one can show that the $n$-dimensional embedding returned by Isomap is approximately distance preserving on the input samples.

Unfortunately any kind of systematic finite sample analysis of manifold embedding algorithms—especially for general classes of manifolds—still largely eludes the manifold learning community. Part of the difficulty is due to the restriction of finding an embedding in exactly $n$ dimensions. It turns out that many simple manifolds (such as a closed loop, a cylinder, a section of a sphere) cannot be isometrically embedded in $\mathbb{R}^n$, where $n$ is the manifold's intrinsic dimension. If these manifolds reside in some high dimensional ambient space, we would at least like to embed them in a lower dimensional space (possibly slightly larger than $n$) while still preserving interpoint geodesic distances.

Here we are interested in investigating low-dimensional distance-preserving manifold embeddings more formally. Given a sample $X$ from an underlying $n$-dimensional manifold $M \subset \mathbb{R}^D$, and an embedding procedure $\mathcal{A} : M \to \mathbb{R}^d$ that (uses $X$ in training and) maps points from $M$ into some low dimensional space $\mathbb{R}^d$, we define the quality of the embedding $\mathcal{A}$ as $(1 \pm \varepsilon)$-isometric if for all

©2013 Nakul Verma.

$p, q \in M$, we have

$$(1-\varepsilon)D_G(p,q) \leq D_G(\mathcal{A}(p),\mathcal{A}(q)) \leq (1+\varepsilon)D_G(p,q),$$

where $D_G$ denotes the geodesic distance. We would like to know i) can one come up with an embedding algorithm $\mathcal{A}$ that achieves $(1\pm\varepsilon)$-isometry for *all* points in $M$? ii) how much can one reduce the target dimension $d$ and still have $(1\pm\varepsilon)$-isometry? and, iii) what kinds of restrictions (if any) does one need on $M$ and $X$?

Since $\mathcal{A}$ only gets to access a finite size sample $X$ from the underlying non-linear manifold $M$, it is essential to assume certain amount of curvature regularity on $M$. Niyogi et al. (2008) provide a nice characterization of manifold curvature via a notion of manifold condition number that will be useful throughout the text (details later).

Perhaps the first algorithmic result for embedding a general $n$-dimensional manifold is due to Baraniuk and Wakin (2009). They show that an orthogonal linear projection of a well-conditioned $n$-dimensional manifold $M \subset \mathbb{R}^D$ into a sufficiently high dimensional *random* subspace is enough to approximately preserve *all* pairwise geodesic distances. To get $(1\pm\varepsilon)$-isometry, they show that a target dimension $d$ of size about $O\left(\frac{n}{\varepsilon^2}\log\frac{VD}{\tau}\right)$ is sufficient, where $V$ is the $n$-dimensional volume of the manifold and $\tau$ is the manifold's curvature condition number. This result was sharpened by Clarkson (2008) and Verma (2011) by completely removing the dependence on ambient dimension $D$ and partially substituting the curvature-condition $\tau$ with more average-case manifold properties. In either case, the $1/\varepsilon^2$ dependence is troublesome: if we want an embedding with all distances within 99% of the original distances (i.e., $\varepsilon = 0.01$), the bounds require the dimension of the target space to be at least 10,000!

### 1.1 Our Contributions

In this work, we give two algorithms that achieve $(1\pm\varepsilon)$-isometry where the dimension of the target space is *independent* of the isometry constant $\varepsilon$. As one expects, this dependency shows up in the sampling density (i.e., the size of $X$) required to compute the embedding. The first algorithm we propose is simple and easy to implement but embeds the given $n$-dimensional manifold in $\tilde{O}(2^{cn})$ dimensions[1] (where $c$ is an absolute constant). The second algorithm, a variation on the first, focuses on minimizing the target dimension. It is computationally more involved and serves a more theoretical purpose: it shows that one can embed the manifold in just $\tilde{O}(n)$ dimensions.

We would like to highlight that both of our proposed algorithms work for a very general class of well-conditioned manifolds. There is no requirement that the underlying manifold is connected, or is globally isometric (or even globally diffeomorphic) to some subset of $\mathbb{R}^n$ as is frequently assumed by several manifold embedding algorithms. In addition, unlike spectrum-based embedding algorithms in the literature, our algorithms yield an explicit embedding that cleanly embeds out-of-sample data points, and provide isometry guarantees over the entire manifold (not just the input samples).

As we shall discuss in the next section, our algorithms are heavily inspired by Nash's embedding technique (Nash, 1954). It is worth noting that the techniques used in our proof are different from what Nash uses in his work; unlike traditional differential-geometric settings, we can only access the underlying manifold through a finite size sample. This makes it difficult to compute quantities (such as the curvature tensor and local functional form of the input manifold, etc.) that are important

1. $\tilde{O}(\cdot)$ notation suppresses the logarithmic dependence on quantities that depend on the intrinsic geometry of the underlying manifold, such as its volume and curvature-condition terms.

in Nash's approach for constructing an isometric embedding. Our work provides insight on how and under what conditions can one use just the samples to construct an approximate isometric embedding of the underlying manifold. In that sense, this work can be viewed as an algorithmic realization of Nash's Embedding Theorem.

## 2. Isometrically Embedding $n$-Dimensional Manifolds: Intuition

Given an underlying $n$-dimensional manifold $M \subset \mathbb{R}^D$, we shall use ideas from Nash's embedding (Nash, 1954) to develop our algorithms. To ease the burden of finding a $(1 \pm \varepsilon)$-isometric embedding directly, our proposed algorithm will be divided in two stages. The first stage will embed $M$ in a lower dimensional space without having to worry about preserving any distances. Since interpoint distances will potentially be distorted by the first stage, the second stage will focus on adjusting these distances by applying a series of corrections. The combined effect of both stages is a distance preserving embedding of $M$ in lower dimensions. We now describe the stages in detail.

### 2.1 Embedding Stage

We shall use the random projection result by Clarkson (2008) (with $\varepsilon$ set to a constant) to embed $M$ into $d = \tilde{O}(n)$ dimensions. This gives an easy one-to-one low-dimensional embedding that doesn't collapse interpoint distances. Note that a projection does contract interpoint distances; by appropriately scaling the random projection, we can make sure that the distances are contracted by at most a constant amount, with high probability.

### 2.2 Correction Stage

Since the random projection can contract different parts of the manifold by different amounts, we will apply several corrections—each corresponding to a different local region—to stretch-out and restore the local distances.

To understand a single correction better, we can consider its effect on a small section of the contracted manifold. Since manifolds are locally linear, the section effectively looks like a contracted $n$-dimensional affine space. Our correction map needs to restore distances over this $n$-flat.

For simplicity, let us temporarily assume $n = 1$ (this corresponds to a 1-dimensional manifold), and let $t \in [0,1]$ parameterize a unit-length segment of the contracted 1-flat. Suppose we want to stretch the segment by a factor of $L \geq 1$ to restore the contracted distances. How can we accomplish this?

Perhaps the simplest thing to do is apply a linear correction $\Psi : t \mapsto Lt$. While this mapping works well for individual local regions, it turns out that this mapping makes it difficult to control the interference between different corrections with overlapping localities.

We instead use extra coordinates and apply a *non-linear* map $\Psi : t \mapsto (t, \sin(Ct), \cos(Ct))$, where $C$ controls the stretch-size. Note that such a spiral map stretches the length of the tangent vectors by a factor of $\sqrt{1+C^2}$, since $\|\Psi'\| = \|d\Psi/dt\| = \|(1, C\cos(Ct), -C\sin(Ct))\| = \sqrt{1+C^2}$. Now since the geodesic distance between any two points $p$ and $q$ on a manifold is given by the expression $\int \|\gamma'(s)\| ds$, where $\gamma$ is a parameterization of the geodesic curve between points $p$ and $q$ (that is, length of a curve is infinitesimal sum of the length of tangent vectors along its path), $\Psi$ stretches the interpoint geodesic distances by a factor of $\sqrt{1+C^2}$ on the resultant surface as well. Thus, to stretch the distances by a factor of $L$, we can set $C := \sqrt{L^2 - 1}$.

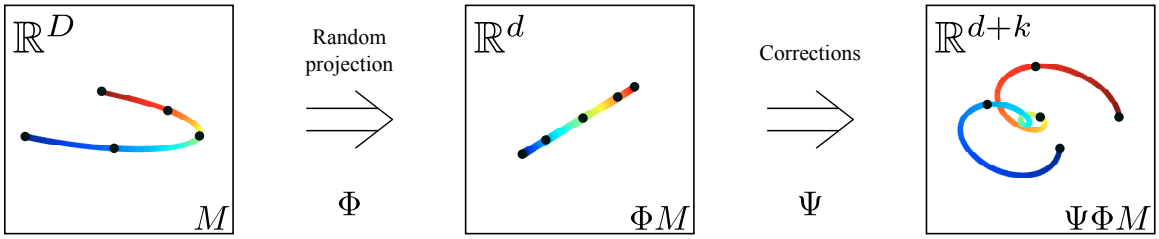



Figure 1: A simple example demonstrating our embedding technique on a 1-dimensional manifold. Left: The original 1-dimensional manifold in some high dimensional space. Middle: A low dimensional mapping of the original manifold via, say, a linear projection onto the vertical plane. Different parts of the manifold are contracted by different amounts—distances at the tail-ends are contracted more than the distances in the middle. Right: Final embedding after applying a series of spiraling corrections. Small size spirals are applied to regions with small distortion (middle), large spirals are applied to regions with large distortions (tail-ends). Resulting embedding is isometric (i.e., geodesic distance preserving) to the original manifold.

Now generalizing this to a local region for an arbitrary $n$-dimensional manifold, let $U := [u^1, \ldots, u^n]$ be a $d \times n$ matrix whose columns form an orthonormal basis for the (local) contracted $n$-flat in the embedded space $\mathbb{R}^d$ and let $\sigma^1, \ldots, \sigma^n$ be the corresponding shrinkages along the $n$ orthogonal directions. Then one can consider applying an $n$-dimensional analog of the spiral mapping: $\Psi : t \mapsto (t, \Psi_{\sin}(t), \Psi_{\cos}(t))$, where $t \in \mathbb{R}^d$

$$\Psi_{\sin}(t) := (\sin((Ct)_1), \ldots, \sin((Ct)_n)), \text{ and}$$
$$\Psi_{\cos}(t) := (\cos((Ct)_1), \ldots, \cos((Ct)_n)).$$

Here $C$ is an $n \times d$ "correction" matrix that encodes how much of the surface needs to stretch in the various orthogonal directions. It turns out that if one sets $C$ to be the matrix $SU^{\mathsf{T}}$, where $S$ is a diagonal matrix with entry $S_{ii} := \sqrt{(1/\sigma^i)^2 - 1}$ (recall that $\sigma^i$ was the shrinkage along direction $u^i$), then the correction $\Psi$ precisely restores the shrinkages along the $n$ orthonormal directions on the resultant surface (see Section 5.2.1 for a detailed derivation).

This takes care of the local regions individually. Now, globally, since different parts of the contracted manifold need to be stretched by different amounts, we localize the effect of the individual $\Psi$'s to a small enough neighborhood by applying a specific kind of kernel function known as the "bump" function in the analysis literature, given by (see also Figure 5 middle)

$$\lambda_x(t) := \mathbf{1}_{\{\|t-x\|<\rho\}} \cdot e^{-1/(1-(\|t-x\|/\rho)^2)}.$$

Applying different $\Psi$'s at different parts of the manifold has an aggregate effect of creating an approximate isometric embedding.

We now have a basic outline of our algorithm. Let $M$ be an $n$-dimensional manifold in $\mathbb{R}^D$. We first find a contraction of $M$ in $d = \tilde{O}(n)$ dimensions via a random projection. This embeds the manifold in low dimensions but distorts the interpoint geodesic distances. We estimate the distortion at different regions of the projected manifold by comparing a sample from $M$ (i.e., $X$)

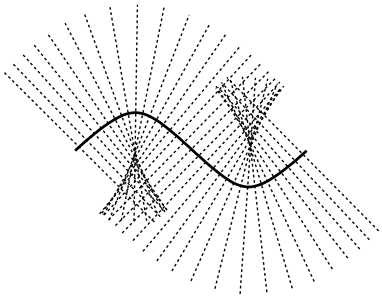


Figure 2: Tubular neighborhood of a manifold. Note that the normals (dotted lines) of a particular length incident at each point of the manifold (solid line) will intersect if the manifold is too curvy.

with its projection. We then perform a series of corrections—each applied locally—to adjust the lengths in the local neighborhoods. We will conclude that restoring the lengths in all neighborhoods yields a globally consistent approximately isometric embedding of $M$. See also Figure 1.

As briefly mentioned earlier, a key issue in preserving geodesic distances across points in different neighborhoods is reconciling the interference between different corrections with overlapping localities. Based on exactly *how* we apply these different local $\Psi$'s gives rise to our two algorithms. For the first algorithm, we shall allocate a fresh set of coordinates for each correction $\Psi$ so that the different corrections don't interfere with each other. Since a local region of an $n$-dimensional manifold can potentially have up to $O(2^{cn})$ overlapping regions, we shall require $O(2^{cn})$ additional coordinates to apply the corrections, making the final embedding dimension of $\tilde{O}(2^{cn})$ (where $c$ is an absolute constant). For the second algorithm, we will follow Nash's technique (Nash, 1954) more closely and apply $\Psi$ maps iteratively in the same embedding space without the use of extra coordinates. At each iteration we need to compute a pair of vectors *normal* to the embedded manifold. Since locally the manifold spreads across its tangent space, these normals indicate the locally empty regions in the embedded space. Applying the local $\Psi$ correction in the direction of these normals gives a way to mitigate the interference between different $\Psi$'s. Since we don't use extra coordinates, the final embedding dimension remains $\tilde{O}(n)$.

## 3. Preliminaries

Let $M$ be a smooth, $n$-dimensional compact Riemannian submanifold of $\mathbb{R}^D$. Note that we do not have any further topological restrictions on $M$; it may or may not have a boundary, or may or may not be orientable. We will frequently refer to such a manifold as an $n$-manifold.

Since we will be working with samples from $M$, we need to ensure certain amount of curvature regularity. Here we borrow the notation from Niyogi et al. (2008) about the condition number of $M$.

**Definition 1 (condition number (Niyogi et al., 2008))** *Let $M \subset \mathbb{R}^D$ be a compact Riemannian manifold. The condition number of $M$ is $\frac{1}{\tau}$, if $\tau$ is the largest number such that the normals of length $r < \tau$ at any two distinct points $p, q \in M$ don't intersect.*

The condition number is based on the notion of "reach" introduced by Federer (1959) and is closely related to the Second Fundamental Form of the manifold. Intuitively, it captures the *com-*

*plexity* of a manifold in terms of the manifold's curvature. If $M$ has condition number $1/\tau$, we can, for instance, bound the directional curvature at any $p \in M$ by $\tau$. Figure 2 depicts the normals of a manifold. Notice that long non-intersecting normals are possible only if the manifold is relatively flat. Hence, the condition number of $M$ gives us a handle on how curvy can $M$ be. As a quick example, let's calculate the condition number of an $n$-dimensional sphere of radius $r$ (embedded in $\mathbb{R}^D$). Note that in this case one can have non-intersecting normals of length less than $r$ (since otherwise they will start intersecting at the center of the sphere). Thus, the condition number of such a sphere is $1/r$. Henceforth we shall assume that $M$ is well-conditioned, that is, $M$ has condition number $1/\tau$. There are several useful properties of well-conditioned manifolds that would be helpful throughout the text; these are outlined in Appendix A.

Since we make minimal topological assumptions on $M$, even a well-conditioned $M$ can have computational degeneracies: $M$, for instance, can have an unbounded number of well-conditioned connected components, yielding unusually large cover sizes. Since we make use of a random projection for the Embedding Stage, it is essential to have good manifold covers. Thus in order to avoid degenerate cases, we shall assume covering regularity on $M$.

**Definition 2 (manifold regularity)** *Let $M \subset \mathbb{R}^D$ be an $n$-manifold with condition number $1/\tau$. We call $M$ as $C_M$-*regular*, if for any $r \leq \tau/2$, the $r$-*covering number *of $M$ is of size at most $(C_M/r)^n$, where $C_M$ is a universal constant dependent only on intrinsic properties of $M$ (such as its $n$-dimensional volume, etc.). That is, there exists a set $S \subset M$ of size at most $(C_M/r)^n$ such that for all $p \in M$, exists $x \in S$ such that $\|p-x\| \leq r$.*

We will use the notation $D_G(p,q)$ to indicate the geodesic distance between points $p$ and $q$ where the underlying manifold is understood from the context, and $\|p-q\|$ to indicate the Euclidean distance between points $p$ and $q$ where the ambient space is understood from the context.

To correctly estimate the distortion induced by the initial contraction mapping, our algorithm needs access to a high-resolution sample from our underlying manifold.

**Definition 3 (bounded manifold cover)** *Let $M \subset \mathbb{R}^D$ be a Riemannian $n$-manifold. We call $X \subset M$ an $\alpha$-bounded $(\rho,\delta)$-cover of $M$ if for all $p \in M$ and $\rho$-neighborhood $X_p := \{x \in X : \|x-p\| < \rho\}$ around $p$, we have*

- *there exist points $x_0,\ldots,x_n \in X_p$ such that $\left|\frac{x_i-x_0}{\|x_i-x_0\|} \cdot \frac{x_j-x_0}{\|x_j-x_0\|}\right| \leq 1/2n$, for $i \neq j$. (local spread criterion)*
- *$|X_p| \leq \alpha$. (local boundedness criterion)*
- *exists point $x \in X_p$ such that $\|x-p\| \leq \rho/2$. (covering criterion)*
- *for any $n+1$ points in $X_p$ satisfying the local spread criterion, let $\hat{T}_p$ denote the $n$-dimensional affine space passing through them (note that $\hat{T}_p$ does not necessarily pass through $p$). Then, for any unit vector $\hat{v}$ in $\hat{T}_p$, we have $\left|\hat{v} \cdot \frac{v}{\|v\|}\right| \geq 1-\delta$, where $v$ is the projection of $\hat{v}$ onto the tangent space of $M$ at $p$. (tangent space approximation criterion)*

The above is an intuitive notion of manifold sampling that can estimate the local tangent spaces. Curiously, we haven't found such "tangent-space approximating" notions of manifold sampling in the literature. We do note in passing that our sampling criterion is similar in spirit to the $(\varepsilon,\delta)$-sampling

(also known as "tight" $\varepsilon$-sampling) criterion popular in the Computational Geometry literature (see, e.g., Dey et al., 2002; Giesen and Wagner, 2003).

**Remark 4** *Given an n-manifold $M$ with condition number $1/\tau$, and some $0 < \delta \leq 1$. If $\rho \leq \tau\delta/16n$, then there exists a $2^{13n}$-bounded $(\rho,\delta)$-cover of $M$ (see Appendix B).*

We can now state our two algorithms.

## 4. The Algorithms

Inputs: We assume the following quantities are given.

(i) $n$ – the intrinsic dimension of $M$.

(ii) $1/\tau$ – the condition number of $M$.

(iii) $X$ – an $\alpha$-bounded $(\rho,\delta)$-cover of $M$.

(iv) $\rho$ – the $\rho$ parameter of the cover.

Notation: Let $\phi$ be a random orthogonal projection map that maps points from $\mathbb{R}^D$ into a random subspace of dimension $d$ ($n \leq d \leq D$). We will have $d$ to be about $\tilde{O}(n)$. Set $\Phi := (2/3)(\sqrt{D/d})\phi$ as a scaled version of $\phi$. Since $\Phi$ is linear, $\Phi$ can also be represented as a $d \times D$ matrix. In our discussion below we will use the function notation and the matrix notation interchangeably, that is, for any $p \in \mathbb{R}^D$, we will use the notation $\Phi(p)$ (applying function $\Phi$ to $p$) and the notation $\Phi p$ (matrix-vector multiplication) interchangeably.

For any $x \in X$, let $x_0, \ldots, x_n$ be $n+1$ points from the set $\{x' \in X : \|x - x'\| < \rho\}$ such that $\left| \frac{x_i - x_0}{\|x_i - x_0\|} \cdot \frac{x_j - x_0}{\|x_j - x_0\|} \right| \leq 1/2n$, for $i \neq j$ (cf. Definition 3). Let $F_x$ be the $D \times n$ matrix whose column vectors form some orthonormal basis of the $n$-dimensional subspace spanned by the vectors $\{x_i - x_0\}_{i \in [n]}$. Note that $F_x$ serves as a good approximation to the tangent spaces at different points in the neighborhood of $x \in M \subset \mathbb{R}^D$.

Estimating local contractions: We estimate the contraction caused by $\Phi$ at a small enough neighborhood of $M$ containing the point $x \in X$, by computing the "thin" Singular Value Decomposition (SVD) $U_x \Sigma_x V_x^{\mathsf{T}}$ of the $d \times n$ matrix $\Phi F_x$ and representing the singular values in the conventional descending order. That is, $\Phi F_x = U_x \Sigma_x V_x^{\mathsf{T}}$, and since $\Phi F_x$ is a tall matrix ($n \leq d$), we know that the bottom $d - n$ singular values are zero. Thus, we only consider the top $n$ (of $d$) left singular vectors in the SVD (so, $U_x$ is $d \times n$, $\Sigma_x$ is $n \times n$, and $V_x$ is $n \times n$) and $\sigma_x^1 \geq \sigma_x^2 \geq \ldots \geq \sigma_x^n$ where $\sigma_x^i$ is the $i^{\text{th}}$ largest singular value.

Observe that the singular values $\sigma_x^1, \ldots, \sigma_x^n$ are precisely the distortion amounts in the directions $u_x^1, \ldots, u_x^n$ at $\Phi(x) \in \mathbb{R}^d$ ($[u_x^1, \ldots, u_x^n] = U_x$) when we apply $\Phi$. To see this, consider the direction $w^i := F_x v_x^i$ in the column-span of $F_x$ ($[v_x^1, \ldots, v_x^n] = V_x$). Then $\Phi w^i = (\Phi F_x) v_x^i = \sigma_x^i u_x^i$, which can be interpreted as: $\Phi$ maps the vector $w^i$ in the column-space of $F_x$ (in $\mathbb{R}^D$) to the vector $u_x^i$ (in $\mathbb{R}^d$) with the scaling of $\sigma_x^i$.

Note that if $0 < \sigma_x^i \leq 1$ (for all $x \in X$ and $1 \leq i \leq n$), we can define an $n \times d$ correction matrix (corresponding to each $x \in X$) $C^x := S_x U_x^{\mathsf{T}}$, where $S_x$ is a diagonal matrix with $(S_x)_{ii} := \sqrt{(1/\sigma_x^i)^2 - 1}$. We can also write $S_x$ as $(\Sigma_x^{-2} - I)^{1/2}$. The correction matrix $C^x$ will have an effect of stretching the

direction $u_x^i$ by the amount $(S_x)_{ii}$ and killing any direction $v$ that is orthogonal to (the column-span of) $U_x$.

**Algorithm 1** Compute Corrections $C^x$'s

1: **for** $x \in X$ (in any order) **do**
2: Let $x_0, \ldots, x_n \in \{x' \in X : \|x' - x\| < \rho\}$ be such that $\left| \frac{x_i - x_0}{\|x_i - x_0\|} \cdot \frac{x_j - x_0}{\|x_j - x_0\|} \right| \leq 1/2n$ (for $i \neq j$).
3: Let $F_x$ be a $D \times n$ matrix whose columns form an orthonormal basis of the $n$-dimensional span of the vectors $\{x_i - x_0\}_{i \in [n]}$.
4: Let $U_x \Sigma_x V_x^\mathsf{T}$ be the "thin" SVD of $\Phi F_x$.
5: Set $C^x := (\Sigma_x^{-2} - I)^{1/2} U_x^\mathsf{T}$.
6: **end for**

**Algorithm 2** Embedding Technique I

**Preprocessing Stage:** Partition the given covering $X$ into disjoint subsets such that no subset contains points that are too close to each other. Let $x_1, \ldots, x_{|X|}$ be the points in $X$ in some arbitrary but fixed order. We can do the partition as follows:

1: Initialize $X^{(1)}, \ldots, X^{(K)}$ as empty sets.
2: **for** $x_i \in X$ (in any fixed order) **do**
3: Let $j$ be the smallest positive integer such that $x_i$ is not within distance $2\rho$ of any element in $X^{(j)}$. That is, the smallest $j$ such that for all $x \in X^{(j)}$, $\|x - x_i\| \geq 2\rho$.
4: $X^{(j)} \leftarrow X^{(j)} \cup \{x_i\}$.
5: **end for**

*The Embedding:* For *any* $p \in M \subset \mathbb{R}^D$, embed it in $\mathbb{R}^{d+2nK}$ as follows:

1: Let $t = \Phi(p)$.
2: Define $\Psi(t) := (t, \Psi_{1,\sin}(t), \Psi_{1,\cos}(t), \ldots, \Psi_{K,\sin}(t), \Psi_{K,\cos}(t))$ where

$$\Psi_{j,\sin}(t) := (\psi^1_{j,\sin}(t), \ldots, \psi^n_{j,\sin}(t)),$$
$$\Psi_{j,\cos}(t) := (\psi^1_{j,\cos}(t), \ldots, \psi^n_{j,\cos}(t)).$$

The individual terms are given by

$$\psi^i_{j,\sin}(t) := \sum_{x \in X^{(j)}} \left( \sqrt{\Lambda_{\Phi(x)}(t)}/\omega \right) \sin(\omega (C^x t)_i) \qquad i = 1, \ldots, n;$$
$$\psi^i_{j,\cos}(t) := \sum_{x \in X^{(j)}} \left( \sqrt{\Lambda_{\Phi(x)}(t)}/\omega \right) \cos(\omega (C^x t)_i) \qquad j = 1, \ldots, K$$

where $\Lambda_a(b) = \frac{\lambda_a(b)}{\sum_{q \in X} \lambda_{\Phi(q)}(b)}$.

3: **return** $\Psi(t)$ as the embedding of $p$ in $\mathbb{R}^{d+2nK}$.

A few remarks are in order.

**Remark 5** *The goal of the Preprocessing Stage is to identify samples from $X$ that can have overlapping ($\rho$-size) local neighborhoods. The partitioning procedure described above ensures that*

*corrections associated with nearby neighborhoods are applied in separate coordinates to minimize interference.*

**Remark 6** *If $\rho \leq \tau/8$, the number of subsets (i.e., K) produced by Embedding I is at most $\alpha 2^{cn}$ for an $\alpha$-bounded $(\rho, \delta)$ cover X of M (where $c \leq 4$). See Appendix C for details.*

**Remark 7** *The function $\Lambda$ acts as a (normalized) localizing kernel that helps in localizing the effects of the spiraling corrections (discussed in detail in Section 5.2).*

**Remark 8** *$\omega > 0$ is a free parameter that controls the interference due to overlapping local corrections.*

**Algorithm 3** Embedding Technique II

**The Embedding:** Let $x_1, \ldots, x_{|X|}$ be the points in $X$ in some arbitrary but fixed order. For *any* point $p \in M \subset \mathbb{R}^D$, we embed it in $\mathbb{R}^{2d+3}$ by:

1: Let $t = \Phi(p)$.
2: Define $\Psi_{0,n}(t) := (t, \underbrace{0, \ldots, 0}_{d+3})$. [Extension needed to efficiently find the normal vectors]
3: **for** $i = 1, \ldots, |X|$ **do**
4: Define $\Psi_{i,0} := \Psi_{i-1,n}$.
5: **for** $j = 1, \ldots, n$ **do**
6: Let $\eta_{i,j}(t)$ and $\nu_{i,j}(t)$ be two mutually orthogonal unit vectors normal to $\Psi_{i,j-1}(\Phi M)$ at $\Psi_{i,j-1}(t)$.
7: Define

$$\Psi_{i,j}(t) := \Psi_{i,j-1}(t) + \left( \frac{\sqrt{\Lambda_{\Phi(x_i)}(t)}}{\omega_{i,j}} \right) \left[ \eta_{i,j}(t) \sin(\omega_{i,j}(C^{x_i} t)_j) + \nu_{i,j}(t) \cos(\omega_{i,j}(C^{x_i} t)_j) \right],$$

where $\Lambda_a(b) = \frac{\lambda_a(b)}{\sum_{q \in X} \lambda_{\Phi(q)}(b)}$.
8: **end for**
9: **end for**
10: **return** $\Psi_{|X|,n}(t)$ as the embedding of $p$ into $\mathbb{R}^{2d+3}$.

**Remark 9** *The function $\Lambda$, and the free parameters $\omega_{i,j}$ (one for each $i, j$ iteration) have roles similar to those in Embedding I.*

**Remark 10** *The success of Embedding II depends upon finding a pair of normal unit vectors $\eta$ and $\nu$ in each iteration; we discuss how to approximate these in Appendix E.*

For appropriate choice of $d$, $\rho$, $\delta$ and $\omega$ (or $\omega_{i,j}$), we have the following.

### 4.1 Main Result

**Theorem 11** *Let $M \subset \mathbb{R}^D$ be a $C_M$-regular $n$-manifold with condition number $1/\tau$. Let the target dimension of the initial random projection mapping $d = \Omega(n\log(C_M/\tau))$ such that $d \leq D$. For any $0 < \varepsilon \leq 1$, let $\rho \leq (\tau d/D)(\varepsilon/350)^2$, $\delta \leq (d/D)(\varepsilon/250)^2$, and let $X \subset M$ be an $\alpha$-bounded $(\rho,\delta)$-cover of $M$. Now, given access to the sample $X$, let*

*i. $N_{\mathrm{I}} \subset \mathbb{R}^{d+2\alpha n 2^{cn}}$ be the embedding of $M$ returned by Algorithm* I *(where $c \leq 4$),*

*ii. $N_{\mathrm{II}} \subset \mathbb{R}^{2d+3}$ be the embedding of $M$ returned by Algorithm* II.

*Then, with probability at least $1 - 1/poly(n)$ over the choice of the initial random projection, for all $p,q \in M$ and their corresponding mappings $p_{\mathrm{I}}, q_{\mathrm{I}} \in N_{\mathrm{I}}$ and $p_{\mathrm{II}}, q_{\mathrm{II}} \in N_{\mathrm{II}}$, we have*

*i.* $(1-\varepsilon)D_G(p,q) \leq D_G(p_{\mathrm{I}}, q_{\mathrm{I}}) \leq (1+\varepsilon)D_G(p,q)$,

*ii.* $(1-\varepsilon)D_G(p,q) \leq D_G(p_{\mathrm{II}}, q_{\mathrm{II}}) \leq (1+\varepsilon)D_G(p,q)$.

## 5. Proof

Our goal is to show that the two proposed embeddings approximately preserve the lengths of all geodesic curves. Now, since the length of any given curve $\gamma : [a,b] \to M$ is given by $\int_a^b \|\gamma'(s)\| ds$, it is vital to study how our embeddings modify the length of the tangent vectors at any point $p \in M$.

In order to discuss tangent vectors, we need to introduce the notion of a tangent space $T_pM$ at a particular point $p \in M$. Consider any smooth curve $c : (-\varepsilon,\varepsilon) \to M$ such that $c(0) = p$, then we know that $c'(0)$ is the vector tangent to $c$ at $p$. The collection of all such vectors formed by all such curves is a well defined vector space (with origin at $p$), called the tangent space $T_pM$. In what follows, we will fix an arbitrary point $p \in M$ and a tangent vector $v \in T_pM$ and analyze how the various steps of the algorithm modify the length of $v$.

Let $\Phi$ be the initial (scaled) random projection map (from $\mathbb{R}^D$ to $\mathbb{R}^d$) that may contract distances on $M$ by various amounts, and let $\Psi$ be the subsequent correction map that attempts to restore these distances (as defined in Step 2 for Embedding I or as a sequence of maps in Step 7 for Embedding II). To get a firm footing for our analysis, we need to study how $\Phi$ and $\Psi$ modify the tangent vector $v$. It is well known from differential geometry that for any smooth map $F : M \to N$ that maps a manifold $M \subset \mathbb{R}^k$ to a manifold $N \subset \mathbb{R}^{k'}$, there exists a linear map $(DF)_p : T_pM \to T_{F(p)}N$, known as the derivative map or the pushforward (at $p$), that maps tangent vectors incident at $p$ in $M$ to tangent vectors incident at $F(p)$ in $N$. To see this, consider a vector $u$ tangent to $M$ at some point $p$. Then, there is some smooth curve $c : (-\varepsilon,\varepsilon) \to M$ such that $c(0) = p$ and $c'(0) = u$. By mapping the curve $c$ into $N$, that is, $F(c(t))$, we see that $F(c(t))$ includes the point $F(p)$ at $t = 0$. Now, by calculus, we know that the derivative at this point, $\left.\frac{dF(c(t))}{dt}\right|_{t=0}$ is the directional derivative $(\nabla F)_p(u)$, where $(\nabla F)_p$ is a $k' \times k$ matrix called the gradient (at $p$). The quantity $(\nabla F)_p$ is precisely the matrix representation of this linear "pushforward" map that sends tangent vectors of $M$ (at $p$) to the corresponding tangent vectors of $N$ (at $F(p)$). Figure 3 depicts how these quantities are affected by applying $F$. Also note that if $F$ is linear, then $DF = F$.

Observe that since pushforward maps are linear, without loss of generality we can assume that $v$ has unit length. Also, since for $n = 0$ there is nothing to prove, we shall assume that $n \geq 1$.

*A quick roadmap for the proof.* In the next three sections, we take a brief detour to study the effects of applying $\Phi$, applying $\Psi$ for Algorithm I, and applying $\Psi$ for Algorithm II separately. This

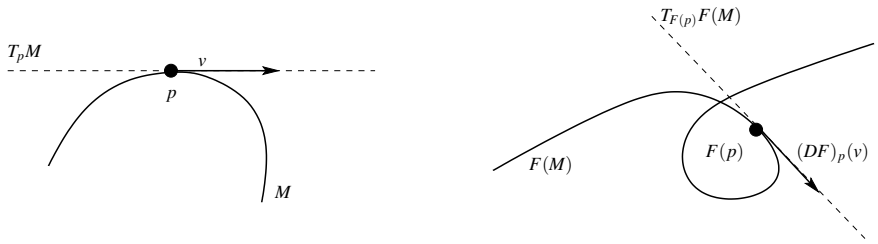



Figure 3: Effects of applying a smooth map $F$ on various quantities of interest. Left: A manifold $M$ containing point $p$. $v$ is a vector tangent to $M$ at $p$. Right: Mapping of $M$ under $F$. Point $p$ maps to $F(p)$, tangent vector $v$ maps to $(DF)_p(v)$.

will give us the necessary tools to analyze the combined effect of applying $\Psi \circ \Phi$ on $v$ (Section 5.4). We will conclude by relating tangent vectors to lengths of curves, showing approximate isometry (Section 5.5). Figure 4 provides a quick sketch of our two stage mapping with the quantities of interest. We defer the proofs of all the supporting lemmas to Appendix D.

### 5.1 Effects of Applying $\Phi$

It is well known as an application of Sard's theorem from differential topology (see, e.g., Milnor, 1972) that almost every smooth mapping of an $n$-dimensional manifold into $\mathbb{R}^{2n+1}$ is a differential structure preserving embedding of $M$. In particular, a projection onto a random subspace (of dimension $2n+1$) constitutes such an embedding with probability 1.

This translates to stating that a random projection into $\mathbb{R}^{2n+1}$ is enough to guarantee that $\Phi$ doesn't collapse the lengths of non-zero tangent vectors almost surely. However, due to computational issues, we additionally require that the lengths are bounded away from zero (that is, a statement of the form $\|(D\Phi)_p(v)\| \geq \Omega(1)\|v\|$ for all $v$ tangent to $M$ at all points $p$).

We can thus appeal to the random projections result by Clarkson (2008) (with the isometry parameter set to a constant, say $1/4$) to ensure this condition. In particular, the following holds.

**Lemma 12** *Let $M \subset \mathbb{R}^D$ be a $C_M$-regular $n$-manifold with condition number $1/\tau$. Let $R$ be a random projection matrix that maps points from $\mathbb{R}^D$ into a random subspace of dimension $d$ ($d \leq D$). Define $\Phi := (2/3)(\sqrt{D/d})R$ as a scaled projection mapping. If $d = \Omega(n\log(C_M/\tau))$, then with probability at least $1 - 1/poly(n)$ over the choice of the random projection matrix, we have*

*(a) For all $p \in M$ and all tangent vectors $v \in T_pM$, $(1/2)\|v\| \leq \|(D\Phi)_p(v)\| \leq (5/6)\|v\|$.*

*(b) For all $p, q \in M$, $(1/2)\|p-q\| \leq \|\Phi p - \Phi q\| \leq (5/6)\|p-q\|$.*

*(c) For all $x \in \mathbb{R}^D$, $\|\Phi x\| \leq (2/3)(\sqrt{D/d})\|x\|$.*

In what follows, we assume that $\Phi$ is such a scaled random projection map. Then, a bound on the length of tangent vectors also gives us a bound on the spectrum of $\Phi F_x$ (recall the definition of $F_x$ from Section 4).

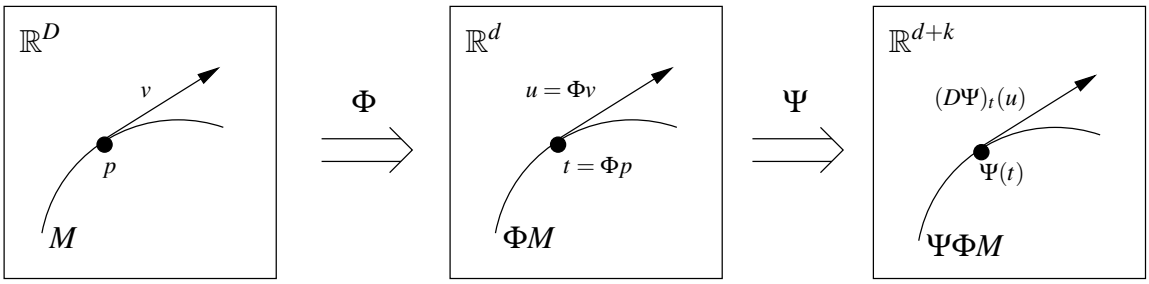



Figure 4: Two stage mapping of our embedding technique. Left: Underlying manifold $M \subset \mathbb{R}^D$ with the quantities of interest—a fixed point $p$ and a fixed unit-vector $v$ tangent to $M$ at $p$. Center: A (scaled) linear projection of $M$ into a random subspace of $d$ dimensions. The point $p$ maps to $\Phi p$ and the tangent vector $v$ maps to $u := (D\Phi)_p(v) = \Phi v$. The length of $v$ contracts to $\|u\|$. Right: Correction of $\Phi M$ via a non-linear mapping $\Psi$ into $\mathbb{R}^{d+k}$. We have $k = O(\alpha 2^{cn})$ for correction technique I, and $k = d+3$ for correction technique II (see also Section 4). Our goal is to show that $\Psi$ stretches length of contracted $v$ (i.e., $u$) back to approximately its original length.

**Corollary 13** *Let $\Phi$, $F_x$ and $n$ be as described above (recall that $x \in X$ that forms a bounded $(\rho, \delta)$-cover of $M$). Let $\sigma_x^i$ represent the $i^{th}$ largest singular value of the matrix $\Phi F_x$. Then, for $\delta \leq d/32D$, we have $1/4 \leq \sigma_x^n \leq \sigma_x^1 \leq 1$ (for all $x \in X$).*

We will be using these facts in our discussion below in Section 5.4.

### 5.2 Effects of Applying $\Psi$ (Algorithm I)

As discussed in Section 2, the goal of $\Psi$ is to restore the contraction induced by $\Phi$ on $M$. To understand the action of $\Psi$ on a tangent vector better, we will first consider a simple case of flat manifolds (Section 5.2.1), and then develop the general case (Section 5.2.2).

#### 5.2.1 Warm-up: Flat $M$

Let us first consider applying a simple one-dimensional spiral map $\bar{\Psi} : \mathbb{R} \to \mathbb{R}^3$ given by $t \mapsto (t, \sin(Ct), \cos(Ct))$, where $t \in I = (-\varepsilon, \varepsilon)$. Let $\bar{v}$ be a unit vector tangent to $I$ (at, say, 0). Then note that

$$(D\bar{\Psi})_{t=0}(\bar{v}) = \frac{d\bar{\Psi}}{dt}\Big|_{t=0} = (1, C\cos(Ct), -C\sin(Ct))\big|_{t=0}.$$

Thus, applying $\bar{\Psi}$ stretches the length of $\bar{v}$ from 1 to $\big\|(1, C\cos(Ct), -C\sin(Ct))|_{t=0}\big\| = \sqrt{1+C^2}$. Notice the advantage of applying the spiral map in computing the lengths: the sine and cosine terms combine together to yield a simple expression for the size of the stretch. In particular, if we want to stretch the length of $\bar{v}$ from 1 to, say, $L \geq 1$, then we simply need $C = \sqrt{L^2 - 1}$ (notice the similarity between this expression and our expression for the diagonal component $S_x$ of the correction matrix $C^x$ in Section 4).

We can generalize this to the case of $n$-dimensional flat manifold (a section of an $n$-flat) by considering a map similar to $\bar{\Psi}$. For concreteness, let $F$ be a $D \times n$ matrix whose column vectors form some orthonormal basis of the $n$-flat manifold (in the original space $\mathbb{R}^D$). Let $U\Sigma V^\mathsf{T}$ be the "thin" SVD of $\Phi F$. Then $FV$ forms an orthonormal basis of the $n$-flat manifold (in $\mathbb{R}^D$) that maps to an orthogonal basis $U\Sigma$ of the projected $n$-flat manifold (in $\mathbb{R}^d$) via the contraction mapping $\Phi$. Define the spiral map $\bar{\Psi} : \mathbb{R}^d \to \mathbb{R}^{d+2n}$ in this case as follows. $\bar{\Psi}(t) := (t, \bar{\Psi}_{\sin}(t), \bar{\Psi}_{\cos}(t))$, with $\bar{\Psi}_{\sin}(t) := (\bar{\psi}^1_{\sin}(t), \ldots, \bar{\psi}^n_{\sin}(t))$ and $\bar{\Psi}_{\cos}(t) := (\bar{\psi}^1_{\cos}(t), \ldots, \bar{\psi}^n_{\cos}(t))$. The individual terms are given as

$$\begin{array}{l} \bar{\psi}^i_{\sin}(t) := \sin((Ct)_i) \\ \bar{\psi}^i_{\cos}(t) := \cos((Ct)_i) \end{array} \quad i = 1, \ldots, n,$$

where $C$ is now an $n \times d$ correction matrix. It turns out that setting $C = (\Sigma^{-2} - I)^{1/2} U^\mathsf{T}$ precisely restores the contraction caused by $\Phi$ to the tangent vectors (notice the similarity between this expression with the correction matrix in the general case $C^x$ in Section 4 and our motivating intuition in Section 2). To see this, let $v$ be a vector tangent to the $n$-flat at some point $p$ (in $\mathbb{R}^D$). We will represent $v$ in the $FV$ basis (that is, $v = \sum_i \alpha_i (Fv^i)$ where $[Fv^1, \ldots, Fv^n] = FV$). Note that $\|\Phi v\|^2 = \|\sum_i \alpha_i \Phi F v^i\|^2 = \|\sum_i \alpha_i \sigma^i u^i\|^2 = \sum_i (\alpha_i \sigma^i)^2$ (where $\sigma^i$ are the individual singular values of $\Sigma$ and $u^i$ are the left singular vectors forming the columns of $U$). Now, let $w$ be the pushforward of $v$ (that is, $w = (D\Phi)_p(v) = \Phi v = \sum_i w_i e^i$, where $\{e^i\}_i$ forms the standard basis of $\mathbb{R}^d$). Now, since $D\bar{\Psi}$ is linear, we have $\|(D\bar{\Psi})_{\Phi(p)}(w)\|^2 = \|\sum_i w_i (D\bar{\Psi})_{\Phi(p)}(e^i)\|^2$, where $(D\bar{\Psi})_{\Phi(p)}(e^i) = \frac{d\bar{\Psi}}{dt^i}\big|_{t=\Phi(p)} = \left(\frac{dt}{dt^i}, \frac{d\bar{\Psi}_{\sin}(t)}{dt^i}, \frac{d\bar{\Psi}_{\cos}(t)}{dt^i}\right)\Big|_{t=\Phi(p)}$. The individual components are given by

$$\begin{array}{l} d\bar{\psi}^k_{\sin}(t)/dt^i = +\cos((Ct)_k) C_{k,i} \\ d\bar{\psi}^k_{\cos}(t)/dt^i = -\sin((Ct)_k) C_{k,i} \end{array} \quad k = 1, \ldots, n;\ i = 1, \ldots, d.$$

By algebra, we see that

$$\begin{aligned} \|(D(\bar{\Psi} \circ \Phi))_p(v)\|^2 &= \|(D\bar{\Psi})_{\Phi(p)}((D\Phi)_p(v))\|^2 = \|(D\bar{\Psi})_{\Phi(p)}(w)\|^2 \\ &= \sum_{i=1}^d w_i^2 \Big(\frac{dt}{dt^i}\Big)^2 + \sum_{i=1}^d \sum_{k=1}^n w_i^2 \Big(\frac{d\psi^k_{\sin}(t)}{dt^i}\Big)^2 + \sum_{i=1}^d \sum_{k=1}^n w_i^2 \Big(\frac{d\psi^k_{\cos}(t)}{dt^i}\Big)^2 \Big|_{t=\Phi(p)} \\ &= \sum_{k=1}^d w_k^2 + \sum_{k=1}^n \cos^2((C\Phi(p))_k)((C\Phi v)_k)^2 + \sum_{k=1}^n \sin^2((C\Phi(p))_k)((C\Phi v)_k)^2 \\ &= \sum_{k=1}^d w_k^2 + \sum_{k=1}^n ((C\Phi v)_k)^2 = \|\Phi v\|^2 + \|C\Phi v\|^2 = \|\Phi v\|^2 + (\Phi v)^\mathsf{T} C^\mathsf{T} C (\Phi v) \\ &= \|\Phi v\|^2 + (\sum_i \alpha_i \sigma^i u^i)^\mathsf{T} U (\Sigma^{-2} - I) U^\mathsf{T} (\sum_i \alpha_i \sigma^i u^i) \\ &= \|\Phi v\|^2 + [\alpha_1 \sigma^1, \ldots, \alpha_n \sigma^n] (\Sigma^{-2} - I) [\alpha_1 \sigma^1, \ldots, \alpha_n \sigma^n]^\mathsf{T} \\ &= \|\Phi v\|^2 + (\sum_i \alpha_i^2 - \sum_i (\alpha_i \sigma^i)^2) = \|\Phi v\|^2 + \|v\|^2 - \|\Phi v\|^2 = \|v\|^2. \end{aligned}$$

In other words, our non-linear correction map $\bar{\Psi}$ can *exactly* restore the contraction caused by $\Phi$ for *any* vector tangent to an $n$-flat manifold.

In the fully general case, the situation gets slightly more complicated since we need to apply different spiral maps, each corresponding to a different size correction at different locations on the

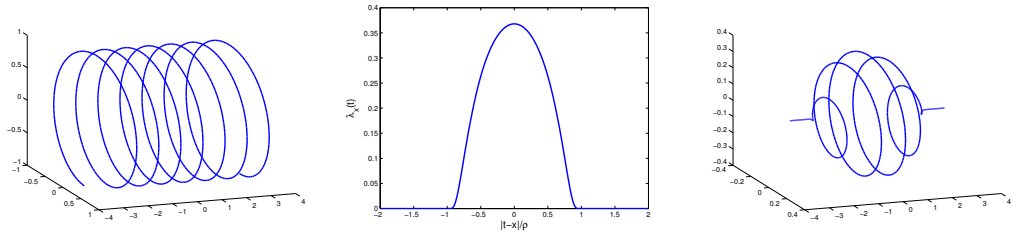



Figure 5: Effects of applying a bump function on a spiral mapping. Left: Spiral mapping $t \mapsto (t, \sin(t), \cos(t))$. Middle: Bump function $\lambda_x$: a smooth function with compact support. The parameter $x$ controls the location while $\rho$ controls the width. Right: The combined effect: $t \mapsto (t, \lambda_x(t)\sin(t), \lambda_x(t)\cos(t))$. Note that the effect of the spiral is localized while keeping the mapping smooth.

contracted manifold. Recall that we localize the effect of a correction by applying the so-called "bump" function (details below). These bump functions, although important for localization, have an undesirable effect on the stretched length of the tangent vector. Thus, to ameliorate their effect on the length of the resulting tangent vector, we control their contribution via a free parameter $\omega$.

### 5.2.2 The General Case

More specifically, Embedding Technique I restores the contraction induced by $\Phi$ by applying a non-linear map $\Psi(t) := (t, \Psi_{1,\sin}(t), \Psi_{1,\cos}(t), \ldots, \Psi_{K,\sin}(t), \Psi_{K,\cos}(t))$ (recall that $K$ is the number of subsets we decompose $X$ into—cf. description in Embedding I in Section 4), with $\Psi_{j,\sin}(t) := (\psi^1_{j,\sin}(t), \ldots, \psi^n_{j,\sin}(t))$ and $\Psi_{j,\cos}(t) := (\psi^1_{j,\cos}(t), \ldots, \psi^n_{j,\cos}(t))$. The individual terms are given as

$$\begin{aligned} \psi^i_{j,\sin}(t) &:= \textstyle\sum_{x \in X^{(j)}} \left(\sqrt{\Lambda_{\Phi(x)}(t)}/\omega\right) \sin(\omega (C^x t)_i) \\ \psi^i_{j,\cos}(t) &:= \textstyle\sum_{x \in X^{(j)}} \left(\sqrt{\Lambda_{\Phi(x)}(t)}/\omega\right) \cos(\omega (C^x t)_i) \end{aligned} \qquad i = 1, \ldots, n; j = 1, \ldots, K,$$

where $C^x$'s are the correction amounts for different locations $x$ on the manifold, $\omega > 0$ controls the frequency (cf. Section 4), and $\Lambda_{\Phi(x)}(t)$ is defined to be $\lambda_{\Phi(x)}(t) / \sum_{q \in X} \lambda_{\Phi(q)}(t)$, with

$$\lambda_{\Phi(x)}(t) := \begin{cases} \exp(-1/(1 - \|t - \Phi(x)\|^2/\rho^2)) & \text{if } \|t - \Phi(x)\| < \rho. \\ 0 & \text{otherwise.} \end{cases}$$

$\lambda$ is a classic example of a *bump function* (see Figure 5 middle). It is a smooth function with compact support. Its applicability arises from the fact that it can be made "to specifications". That is, it can be made to vanish outside any interval of our choice. Here we exploit this property to localize the effect of our corrections. The normalization of $\lambda$ (the function $\Lambda$) creates the so-called smooth partition of unity that helps to vary smoothly between the spirals applied at different regions of $M$.

Since any tangent vector in $\mathbb{R}^d$ can be expressed in terms of the basis vectors, it suffices to study how $D\Psi$ acts on the standard basis $\{e^i\}$. Note that

$$(D\Psi)_t(e^i) = \left(\frac{dt}{dt^i}, \frac{d\Psi_{1,\sin}(t)}{dt^i}, \frac{d\Psi_{1,\cos}(t)}{dt^i}, \ldots, \frac{d\Psi_{K,\sin}(t)}{dt^i}, \frac{d\Psi_{K,\cos}(t)}{dt^i}\right)\Big|_t,$$

where ($k \in [n]$, $i \in [d]$, $j \in [K]$)

$$\begin{array}{l} \frac{d\psi^k_{j,\sin}(t)}{dt^i} = \sum_{x\in X^{(j)}} \frac{1}{\omega}\Big(\sin(\omega(C^x t)_k)\frac{d\Lambda^{1/2}_{\Phi(x)}(t)}{dt^i}\Big) + \sqrt{\Lambda_{\Phi(x)}(t)}\cos(\omega(C^x t)_k)C^x_{k,i} \\ \frac{d\psi^k_{j,\cos}(t)}{dt^i} = \sum_{x\in X^{(j)}} \frac{1}{\omega}\Big(\cos(\omega(C^x t)_k)\frac{d\Lambda^{1/2}_{\Phi(x)}(t)}{dt^i}\Big) - \sqrt{\Lambda_{\Phi(x)}(t)}\sin(\omega(C^x t)_k)C^x_{k,i} \end{array}.$$

One can now observe the advantage of having the term $\omega$. By picking $\omega$ sufficiently large, we can make the first part of the expression sufficiently small. Now, for any tangent vector $u = \sum_i u_i e^i$ such that $\|u\| \leq 1$, we have (by algebra)

$$\begin{aligned} \big\|(D\Psi)_t(u)\big\|^2 &= \Big\|\sum_i u_i (D\Psi)_t(e^i)\Big\|^2 \\ &= \sum_{i=1}^d u_i^2\Big(\frac{dt}{dt^i}\Big)^2 + \sum_{i=1}^d\sum_{j=1}^K\sum_{k=1}^n u_i^2\Big(\frac{d\psi^k_{j,\sin}(t)}{dt^i}\Big)^2 + \sum_{i=1}^d\sum_{j=1}^K\sum_{k=1}^n u_i^2\Big(\frac{d\psi^k_{j,\cos}(t)}{dt^i}\Big)^2 \\ &= \sum_{k=1}^d u_k^2 + \sum_{k=1}^n\sum_{j=1}^K\Big[\sum_{x\in X^{(j)}}\Big(\frac{A^{k,x}_{\sin}(t)}{\omega}\Big) + \sqrt{\Lambda_{\Phi(x)}(t)}\cos(\omega(C^x t)_k)(C^x u)_k\Big]^2 \\ &\qquad + \Big[\sum_{x\in X^{(j)}}\Big(\frac{A^{k,x}_{\cos}(t)}{\omega}\Big) - \sqrt{\Lambda_{\Phi(x)}(t)}\sin(\omega(C^x t)_k)(C^x u)_k\Big]^2, \end{aligned} \tag{1}$$

where the individual terms $A^{k,x}_{\sin}(t) := \sum_i u_i \sin(\omega(C^x t)_k)(d\Lambda^{1/2}_{\Phi(x)}(t)/dt^i)$, and similarly $A^{k,x}_{\cos}(t) := \sum_i u_i \cos(\omega(C^x t)_k)(d\Lambda^{1/2}_{\Phi(x)}(t)/dt^i)$. We can further simplify Equation (1) and get

**Lemma 14** *Let $t$ be any point in $\Phi(M)$ and $u$ be any vector tangent to $\Phi(M)$ at $t$ such that $\|u\| \leq 1$. Let $d$, $\varepsilon$, $\rho$ and $\alpha$ be as per the statement of Theorem 11. Pick $\omega \geq \Omega(n\alpha^2 16^n\sqrt{d}/\rho\varepsilon)$, then*

$$\|(D\Psi)_t(u)\|^2 = \|u\|^2 + \sum_{x\in X}\Lambda_{\Phi(x)}(t)\sum_{k=1}^n (C^x u)_k^2 + \zeta, \tag{2}$$

*where $|\zeta| \leq \varepsilon/2$.*

We will use this derivation of $\|(D\Psi)_t(u)\|^2$ to study the combined effect of $\Psi \circ \Phi$ on $M$ in Section 5.4.

### 5.3 Effects of Applying $\Psi$ (Algorithm II)

The goal of the second algorithm is to apply the spiraling corrections while using the coordinates more economically. We achieve this goal by applying them sequentially in the same embedding space (rather than simultaneously by making use of extra $2nK$ coordinates as done in the first algorithm), see also Nash (1954). Since all the corrections will be sharing the same coordinate space, one needs to keep track of a pair of normal vectors in order to prevent interference among the different local corrections.

More specifically, $\Psi : \mathbb{R}^d \to \mathbb{R}^{2d+3}$ (in Algorithm II) is defined recursively as $\Psi := \Psi_{|X|,n}$ such that (see also Embedding II in Section 4)

$$\Psi_{i,j}(t) := \Psi_{i,j-1}(t) + \eta_{i,j}(t)\frac{\sqrt{\Lambda_{\Phi(x_i)}(t)}}{\omega_{i,j}}\sin(\omega_{i,j}(C^{x_i}t)_j) + \nu_{i,j}(t)\frac{\sqrt{\Lambda_{\Phi(x_i)}(t)}}{\omega_{i,j}}\cos(\omega_{i,j}(C^{x_i}t)_j),$$

where $\Psi_{i,0}(t) := \Psi_{i-1,n}(t)$, and the base function $\Psi_{0,n}(t)$ is given as $t \mapsto (t, \overbrace{0,\ldots,0}^{d+3})$. $\eta_{i,j}(t)$ and $\nu_{i,j}(t)$ are mutually orthogonal unit vectors that are approximately normal to $\Psi_{i,j-1}(\Phi M)$ at $\Psi_{i,j-1}(t)$. In this section we assume that the normals $\eta$ and $\nu$ have the following properties:

- $|\eta_{i,j}(t)\cdot v| \le \varepsilon_0$ and $|\nu_{i,j}(t)\cdot v| \le \varepsilon_0$ for all unit-length $v$ tangent to $\Psi_{i,j-1}(\Phi M)$ at $\Psi_{i,j-1}(t)$. (quality of normal approximation)
- For all $1 \le l \le d$, we have $\|d\eta_{i,j}(t)/dt^l\| \le K_{i,j}$ and $\|d\nu_{i,j}(t)/dt^l\| \le K_{i,j}$. (bounded directional derivatives)

We refer the reader to Appendix E for details on how to estimate such normals.

Now, as before, representing a tangent vector $u = \sum_l u_l e^l$ (such that $\|u\|^2 \le 1$) in terms of its basis vectors, it suffices to study how $D\Psi$ acts on basis vectors. Observe that $(D\Psi_{i,j})_t(e^l) = \left(\frac{d\Psi_{i,j}(t)}{dt^l}\right)_{k=1}^{2d+3}\Big|_t$, with the $k^{\text{th}}$ component given as

$$\begin{aligned}
&\left(\frac{d\Psi_{i,j-1}(t)}{dt^l}\right)_k + (\eta_{i,j}(t))_k\sqrt{\Lambda_{\Phi(x_i)}(t)}C^{x_i}_{j,l}B^{i,j}_{\cos}(t) - (\nu_{i,j}(t))_k\sqrt{\Lambda_{\Phi(x_i)}(t)}C^{x_i}_{j,l}B^{i,j}_{\sin}(t)\\
&\qquad + \frac{1}{\omega_{i,j}}\Big[\Big(\frac{d\eta_{i,j}(t)}{dt^l}\Big)_k\sqrt{\Lambda_{\Phi(x_i)}(t)}B^{i,j}_{\sin}(t) + \Big(\frac{d\nu_{i,j}(t)}{dt^l}\Big)_k\sqrt{\Lambda_{\Phi(x_i)}(t)}B^{i,j}_{\cos}(t)\\
&\qquad\qquad + (\eta_{i,j}(t))_k\frac{d\Lambda^{1/2}_{\Phi(x_i)}(t)}{dt^l}B^{i,j}_{\sin}(t) + (\nu_{i,j}(t))_k\frac{d\Lambda^{1/2}_{\Phi(x_i)}(t)}{dt^l}B^{i,j}_{\cos}(t)\Big],
\end{aligned}$$

where $B^{i,j}_{\cos}(t) := \cos(\omega_{i,j}(C^{x_i}t)_j)$ and $B^{i,j}_{\sin}(t) := \sin(\omega_{i,j}(C^{x_i}t)_j)$. For ease of notation, let $R^{k,l}_{i,j}$ be the terms in the bracket (being multiplied to $1/\omega_{i,j}$) in the above expression. Then, we have for any $i$, $j$

$$\begin{aligned}
\|(D\Psi_{i,j})_t(u)\|^2 &= \Big\|\sum_l u_l(D\Psi_{i,j})_t(e^l)\Big\|^2\\
&= \sum_{k=1}^{2d+3}\Big[\underbrace{\sum_l u_l\left(\frac{d\Psi_{i,j-1}(t)}{dt^l}\right)_k}_{\zeta^{k,1}_{i,j}} + \underbrace{(\eta_{i,j}(t))_k\sqrt{\Lambda_{\Phi(x_i)}(t)}\cos(\omega_{i,j}(C^{x_i}t)_j)\sum_l C^{x_i}_{j,l}u_l}_{\zeta^{k,2}_{i,j}}\\
&\qquad \underbrace{-(\nu_{i,j}(t))_k\sqrt{\Lambda_{\Phi(x_i)}(t)}\sin(\omega_{i,j}(C^{x_i}t)_j)\sum_l C^{x_i}_{j,l}u_l}_{\zeta^{k,3}_{i,j}} + (1/\omega_{i,j})\underbrace{\sum_l u_l R^{k,l}_{i,j}}_{\zeta^{k,4}_{i,j}}\Big]^2\\
&= \underbrace{\|(D\Psi_{i,j-1})_t(u)\|^2}_{=\sum_k\left(\zeta^{k,1}_{i,j}\right)^2} + \underbrace{\Lambda_{\Phi(x_i)}(t)(C^{x_i}u)^2_j}_{=\sum_k\left(\zeta^{k,2}_{i,j}\right)^2+\left(\zeta^{k,3}_{i,j}\right)^2}\\
&\qquad \underbrace{+\sum_k\Big[\left(\zeta^{k,4}_{i,j}/\omega_{i,j}\right)^2 + \left(2\zeta^{k,4}_{i,j}/\omega_{i,j}\right)\left(\zeta^{k,1}_{i,j}+\zeta^{k,2}_{i,j}+\zeta^{k,3}_{i,j}\right) + 2\left(\zeta^{k,1}_{i,j}\zeta^{k,2}_{i,j}+\zeta^{k,1}_{i,j}\zeta^{k,3}_{i,j}\right)\Big]}_{Z_{i,j}},
\end{aligned} \tag{3}$$

where the last equality is by expanding the square and by noting that $\sum_k \zeta^{k,2}_{i,j}\zeta^{k,3}_{i,j} = 0$ since $\eta$ and $\nu$ are orthogonal to each other. The base case $\|(D\Psi_{0,n})_t(u)\|^2$ equals $\|u\|^2$.

Again, by picking $\omega_{i,j}$ sufficiently large, and by noting that the cross terms $\sum_k(\zeta_{i,j}^{k,1}\zeta_{i,j}^{k,2})$ and $\sum_k(\zeta_{i,j}^{k,1}\zeta_{i,j}^{k,3})$ are very close to zero since $\eta$ and $\nu$ are approximately normal to the tangent vector, we have

**Lemma 15** *Let $t$ be any point in $\Phi(M)$ and $u$ be any vector tangent to $\Phi(M)$ at $t$ such that $\|u\| \leq 1$. Let $\varepsilon$ be the isometry parameter chosen in Theorem 11. Pick $\omega_{i,j} \geq \Omega\big((K_{i,j}+(\alpha 16^n/\rho))(nd|X|)^2/\varepsilon\big)$ (recall that $K_{i,j}$ is the bound on the directional derivative of $\eta$ and $\nu$). If $\varepsilon_0 \leq O\big(\varepsilon/\sqrt{d}(n|X|)^2\big)$ (recall that $\varepsilon_0$ is the quality of approximation of the normals $\eta$ and $\nu$), then we have*

$$\|(D\Psi)_t(u)\|^2 = \|(D\Psi_{|X|,n})_t(u)\|^2 = \|u\|^2 + \sum_{i=1}^{|X|}\Lambda_{\Phi(x_i)}(t)\sum_{j=1}^{n}(C^{x_i}u)_j^2 + \zeta, \tag{4}$$

*where $|\zeta| \leq \varepsilon/2$.*

### 5.4 Combined Effect of $\Psi(\Phi(M))$

We can now analyze the aggregate effect of both our embeddings on the length of an arbitrary unit vector $v$ tangent to $M$ at $p$. Let $u := (D\Phi)_p(v) = \Phi v$ be the pushforward of $v$. Then $\|u\| \leq 1$ (cf. Lemma 12). See also Figure 4.

Now, recalling that $D(\Psi\circ\Phi) = D\Psi\cdot D\Phi$, and noting that pushforward maps are linear, we have $\|(D(\Psi\circ\Phi))_p(v)\|^2 = \big\|(D\Psi)_{\Phi(p)}(u)\big\|^2$. Thus, representing $u$ as $\sum_i u_i e^i$ in ambient coordinates of $\mathbb{R}^d$, and using Equation (2) (for Algorithm I) or Equation (4) (for Algorithm II), we get

$$\big\|(D(\Psi\circ\Phi))_p(v)\big\|^2 \quad = \quad \big\|(D\Psi)_{\Phi(p)}(u)\big\|^2 = \|u\|^2 + \sum_{x\in X}\Lambda_{\Phi(x)}(\Phi(p))\|C^x u\|^2 + \zeta,$$

where $|\zeta| \leq \varepsilon/2$. We can give simple lower and upper bounds for the above expression by noting that $\Lambda_{\Phi(x)}$ is a localization function. Define $N_p := \{x \in X : \|\Phi(x)-\Phi(p)\| < \rho\}$ as the neighborhood around $p$ ($\rho$ as per the theorem statement). Then only the points in $N_p$ contribute to above equation, since $\Lambda_{\Phi(x)}(\Phi(p)) = d\Lambda_{\Phi(x)}(\Phi(p))/dt^i = 0$ for $\|\Phi(x)-\Phi(p)\| \geq \rho$. Also note that for all $x \in N_p$, $\|x-p\| < 2\rho$ (cf. Lemma 12).

Let $x_M := \arg\max_{x\in N_p}\|C^x u\|^2$ and $x_m := \arg\min_{x\in N_p}\|C^x u\|^2$ be the quantities that attain the maximum and the minimum respectively. Then:

$$\|u\|^2 + \|C^{x_m}u\|^2 - \varepsilon/2 \leq \|(D(\Psi\circ\Phi))_p(v)\|^2 \leq \|u\|^2 + \|C^{x_M}u\|^2 + \varepsilon/2. \tag{5}$$

Notice that ideally we would like to have the correction factor "$C^p u$" in Equation (5) since that would give the perfect stretch around the point $p$. But what about correction $C^x u$ for nearby $x$'s? The following lemma helps us continue in this situation.

**Lemma 16** *Let $p$, $v$, $u$ be as above. For any $x \in N_p \subset X$, let $C^x$ and $F_x$ also be as discussed above (recall that $\|p-x\| < 2\rho$, and $X \subset M$ forms a bounded $(\rho,\delta)$-cover of the fixed underlying manifold $M$ with condition number $1/\tau$). Define $\xi := (4\rho/\tau)+\delta+4\sqrt{\rho\delta/\tau}$. If $\rho \leq \tau/4$ and $\delta \leq d/32D$, then*

$$1 - \|u\|^2 - 40\cdot\max\Big\{\sqrt{\xi D/d}, \xi D/d\Big\} \leq \|C^x u\|^2 \leq 1 - \|u\|^2 + 51\cdot\max\Big\{\sqrt{\xi D/d}, \xi D/d\Big\}.$$

Note that we chose $\rho \leq (\tau d/D)(\varepsilon/350)^2$ and $\delta \leq (d/D)(\varepsilon/250)^2$ (cf. theorem statement). Thus, combining Equation (5) and Lemma 16, we get (recall $\|v\| = 1$)

$$(1-\varepsilon)\|v\|^2 \leq \|(D(\Psi \circ \Phi))_p(v)\|^2 \leq (1+\varepsilon)\|v\|^2.$$

So far we have shown that our embedding approximately preserves the length of a fixed tangent vector at a fixed point. Since the choice of the vector and the point was arbitrary, it follows that our embedding approximately preserves the tangent vector lengths throughout the embedded manifold uniformly. We will now show that preserving the tangent vector lengths implies preserving the geodesic curve lengths.

### 5.5 Preservation of the Geodesic Lengths

Pick any two (path-connected) points $p$ and $q$ in $M$, and let $\alpha$ be the geodesic[2] path between $p$ and $q$. Further let $\bar{p}$, $\bar{q}$ and $\bar{\alpha}$ be the images of $p$, $q$ and $\alpha$ under our embedding. Note that $\bar{\alpha}$ is not necessarily the geodesic path between $\bar{p}$ and $\bar{q}$, thus we need an extra piece of notation: let $\bar{\beta}$ be the geodesic path between $\bar{p}$ and $\bar{q}$ (under the embedded manifold) and $\beta$ be its inverse image in $M$. We need to show $(1-\varepsilon)L(\alpha) \leq L(\bar{\beta}) \leq (1+\varepsilon)L(\alpha)$, where $L(\cdot)$ denotes the length of the path $\cdot$ (end points are understood).

First recall that for any differentiable map $F$ and curve $\gamma$, $\bar{\gamma} = F(\gamma) \Rightarrow \bar{\gamma}' = (DF)(\gamma')$. By $(1\pm\varepsilon)$-isometry of tangent vectors, this immediately gives us $(1-\varepsilon)L(\gamma) \leq L(\bar{\gamma}) \leq (1+\varepsilon)L(\gamma)$ for any path $\gamma$ in $M$ and its image $\bar{\gamma}$ in embedding of $M$. So,

$$(1-\varepsilon)D_G(p,q) = (1-\varepsilon)L(\alpha) \leq (1-\varepsilon)L(\beta) \leq L(\bar{\beta}) = D_G(\bar{p},\bar{q}).$$

Similarly,

$$D_G(\bar{p},\bar{q}) = L(\bar{\beta}) \leq L(\bar{\alpha}) \leq (1+\varepsilon)L(\alpha) = (1+\varepsilon)D_G(p,q).$$

## 6. Conclusion

This work provides two algorithms for $(1\pm\varepsilon)$-isometric embedding of generic $n$-dimensional manifolds. Our algorithms are similar in spirit to Nash's construction (Nash, 1954), and manage to remove the dependence on the isometry constant $\varepsilon$ from the final embedding dimension. Note that this dependency does necessarily show up in the sampling density required to make the corrections.

The correction procedure discussed here can also be readily adapted to create isometric embeddings from any manifold embedding procedure (under some mild conditions). Take any off-the-shelf manifold embedding algorithm $\mathcal{A}$ (such as LLE, Laplacian Eigenmaps, etc.) that maps an $n$-dimensional manifold in, say, $d$ dimensions, but does not necessarily guarantee an approximate isometric embedding. Then as long as one can ensure that $\mathcal{A}$ is a one-to-one mapping that doesn't collapse interpoint distances, we can scale the output returned by $\mathcal{A}$ to create a contraction. The scaled version of $\mathcal{A}$ acts as the Embedding Stage of our algorithm. We can thus apply the Corrections Stage (either the one discussed in Algorithm I or Algorithm II) to produce an approximate isometric embedding of the given manifold in slightly higher dimensions. In this sense, the correction procedure presented here serves as a *universal procedure* for approximate isometric manifold embeddings.

2. Globally, geodesic paths between points are not necessarily unique; we are interested in a path that yields the shortest distance between the points.

## Acknowledgments

The author is grateful to Sanjoy Dasgupta for introducing the subject, and for his guidance throughout the project. The author would also like to thank Michael Wakin, Armin Eftekhari, and the anonymous reviewers for suggesting improvements to the manuscript.

## Appendix A. Properties of a Well-conditioned Manifold

Throughout this section we will assume that $M$ is a compact submanifold of $\mathbb{R}^D$ of dimension $n$, and condition number $1/\tau$. The following are some properties of such a manifold that would be useful throughout the text.

**Lemma 17 (relating nearby tangent vectors—implicit in the proof of Proposition 6.2 Niyogi et al., 2008)** *Pick any two (path-connected) points $p,q \in M$. Let $u \in T_pM$ be a unit length tangent vector and $v \in T_qM$ be its parallel transport along the (shortest) geodesic path to $q$. Then,*[3] *i) $u \cdot v \geq 1 - D_G(p,q)/\tau$, ii) $\|u-v\| \leq \sqrt{2D_G(p,q)/\tau}$.*

**Lemma 18 (relating geodesic distances to ambient distances—Proposition 6.3 of Niyogi et al., 2008)** *If $p,q \in M$ such that $\|p-q\| \leq \tau/2$, then $D_G(p,q) \leq \tau(1-\sqrt{1-2\|p-q\|/\tau}) \leq 2\|p-q\|$.*

**Lemma 19 (projection of a section of a manifold onto the tangent space)** *Pick any $p \in M$ and define $M_{p,r} := \{q \in M : \|q-p\| \leq r\}$. Let $f$ denote the orthogonal linear projection of $M_{p,r}$ onto the tangent space $T_pM$. Then, for any $r \leq \tau/4$*

*(i) the map $f : M_{p,r} \to T_pM$ is one-to-one. (follows from Lemma 5.4, 6.1-6.3 of Niyogi et al. (2008))*

*(ii) for any $x,y \in M_{p,r}$, $\|f(x)-f(y)\|^2 \geq \left(1-\left(\frac{r}{\tau}+\sqrt{\frac{2r}{\tau}}\right)^2\right)\cdot\|x-y\|^2$. (follows from Lemma 5.3, 6.2, 6.3 of Niyogi et al., 2008)*

**Lemma 20 (coverings of a section of a manifold)** *Pick any $p \in M$ and define $M_{p,r} := \{q \in M : \|q-p\| \leq r\}$. If $r \leq \tau/4$, then there exists $C \subset M_{p,r}$ of size at most $16^n$ with the property: for any $p' \in M_{p,r}$, exists $c \in C$ such that $\|p'-c\| \leq r/2$.*

**Proof** The proof closely follows the arguments presented in the proof of Theorem 22 of Dasgupta and Freund (2008).

For $r \leq \tau/4$, note that $M_{p,r} \subset \mathbb{R}^D$ is (path-)connected. Let $f$ denote the projection of $M_{p,r}$ onto $T_pM \cong \mathbb{R}^n$. Quickly note that $f$ is one-to-one (see Lemma 19(i)). Then, $f(M_{p,r}) \subset \mathbb{R}^n$ is contained in an $n$-dimensional ball of radius $r$. By standard volume arguments, $f(M_{p,r})$ can be covered by at most $16^n$ balls of radius $r/7$ (see, e.g., Lemma 5.2 of Vershynin, 2010)). WLOG we can assume that the centers of these covering balls are in $f(M_{p,r})$. Note that the inverse image of each of these

3. Technically, it is not possible to directly compare two vectors that reside in different tangent spaces. However, since we only deal with manifolds that are immersed in some ambient space, we can treat the tangent spaces as $n$-dimensional affine subspaces. We can thus parallel translate the vectors to the origin of the ambient space, and do the necessary comparison (such as take the dot product, etc.). We will make a similar abuse of notation for any calculation that uses vectors from different affine subspaces to mean to first translate the vectors and then perform the necessary calculation.

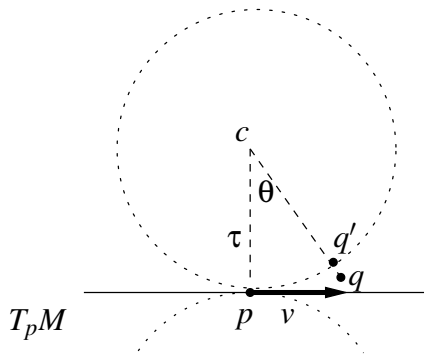



Figure 6: Plane spanned by vectors $q-p$ and $v \in T_pM$ (where $v$ is the projection of $q-p$ onto $T_pM$), with $\tau$-balls tangent to $p$. Note that $q'$ is the point on the ball such that $\angle pcq = \angle pcq' = \theta$.

covering balls (in $\mathbb{R}^n$) is contained in a $D$-dimensional ball of radius $r/2$ that is centered at some point in $M_{p,r}$ (by noting $r \leq \tau/4$ and using Lemma 19(ii)). Thus, the centers of these $D$-dimensional balls (containing the inverse images) forms the desired covering. ■

**Lemma 21 (relating nearby manifold points to tangent vectors)** *Pick any point $p \in M$ and let $q \in M$ (distinct from $p$) be such that $D_G(p,q) \leq \tau$. Let $v \in T_pM$ be the projection of the vector $q-p$ onto $T_pM$. Then, i) $\left| \frac{v}{\|v\|} \cdot \frac{q-p}{\|q-p\|} \right| \geq 1-(D_G(p,q)/2\tau)^2$, ii) $\left\| \frac{v}{\|v\|} - \frac{q-p}{\|q-p\|} \right\| \leq D_G(p,q)/\tau\sqrt{2}$.*

**Proof** If vectors $v$ and $q-p$ are in the same direction, we are done. Otherwise, consider the plane spanned by vectors $v$ and $q-p$. Then since $M$ has condition number $1/\tau$, we know that the point $q$ cannot lie within any $\tau$-ball tangent to $M$ at $p$ (see Figure 6). Consider such a $\tau$-ball (with center $c$) whose center is closest to $q$ and let $q'$ be the point on the surface of the ball which subtends the same angle ($\angle pcq'$) as the angle formed by $q$ ($\angle pcq$). Let this angle be called $\theta$. Then using cosine rule, we have $\cos\theta = 1 - \|q'-p\|^2/2\tau^2$.

Define $\alpha$ as the angle subtended by vectors $v$ and $q-p$, and $\alpha'$ the angle subtended by vectors $v$ and $q'-p$. WLOG we can assume that the angles $\alpha$ and $\alpha'$ are less than $\pi/2$. Then, $\cos\alpha \geq \cos\alpha' = \cos\theta/2$. Using the trigonometric identity $\cos\theta = 2\cos^2\left(\frac{\theta}{2}\right) - 1$, and noting $\|q-p\|^2 \geq \|q'-p\|^2$, we have

$$\left| \frac{v}{\|v\|} \cdot \frac{q-p}{\|q-p\|} \right| = \cos\alpha \geq \cos\frac{\theta}{2} \geq \sqrt{1 - \|q-p\|^2/4\tau^2} \geq 1-(D_G(p,q)/2\tau)^2.$$

Now, by applying the cosine rule, we have $\left\| \frac{v}{\|v\|} - \frac{q-p}{\|q-p\|} \right\|^2 = 2(1-\cos\alpha)$. The lemma follows. ■

**Lemma 22 (approximating tangent space by nearby samples)** *Let $0 < \delta \leq 1$. Pick any point $p_0 \in M$ and let $p_1, \ldots, p_n \in M$ be $n$ points distinct from $p_0$ such that (for all $1 \leq i \leq n$)*

*(i) $D_G(p_0, p_i) \leq \tau\delta/\sqrt{n}$,*

*(ii)* $\left|\frac{p_i-p_0}{\|p_i-p_0\|}\cdot\frac{p_j-p_0}{\|p_j-p_0\|}\right| \le 1/2n$ *(for* $i \neq j$*).*

*Let* $\hat{T}$ *be the* $n$ *dimensional subspace spanned by vectors* $\{p_i-p_0\}_{i\in[n]}$*. For any unit vector* $\hat{u}\in\hat{T}$*, let* $u$ *be the projection of* $\hat{u}$ *onto* $T_{p_0}M$*. Then,* $\left|\hat{u}\cdot\frac{u}{\|u\|}\right| \ge 1-\delta$*.*

**Proof** Define the vectors $\hat{v}_i := \frac{p_i-p_0}{\|p_i-p_0\|}$ (for $1\le i\le n$). Observe that $\{\hat{v}_i\}_{i\in[n]}$ forms a basis of $\hat{T}$. For $1\le i\le n$, define $v_i$ as the projection of vector $\hat{v}_i$ onto $T_{p_0}M$. Also note that by applying Lemma 21, we have that for all $1\le i\le n$, $\|\hat{v}_i - v_i\|^2 \le \delta^2/2n$.

Now consider any unit $\hat{u}\in\hat{T}$, and its projection $u$ in $T_{p_0}M$. Let $V=[\hat{v}_1,\ldots,\hat{v}_n]$ be the $D\times n$ matrix with columns $v_1,\ldots,v_n$. We represent the unit vector $\hat{u}$ as $V\alpha=\sum_i \alpha_i\hat{v}_i$. Also, since $u$ is the projection of $\hat{u}$, we have $u=\sum_i\alpha_i v_i$. Then, $\|\alpha\|^2\le 2$. To see this, we first identify $\hat{T}$ with $\mathbb{R}^n$ via an isometry $S$ (a linear map that preserves the lengths and angles of all vectors in $\hat{T}$). Note that $S$ can be represented as an $n\times D$ matrix, and since the columns of $V$ form a basis for $\hat{T}$, $SV$ is an $n\times n$ invertible matrix. Then, since $S\hat{u}=SV\alpha$, we have $\alpha=(SV)^{-1}S\hat{u}$. Thus, (recall $\|S\hat{u}\|=1$)

$$\begin{aligned}\|\alpha\|^2 &\le \max_{x\in S^{n-1}}\|(SV)^{-1}x\|^2 = \lambda_{\max}((SV)^{-\mathsf{T}}(SV)^{-1})\\ &= \lambda_{\max}((SV)^{-1}(SV)^{-\mathsf{T}}) = \lambda_{\max}((V^{\mathsf{T}}V)^{-1}) = 1/\lambda_{\min}(V^{\mathsf{T}}V)\\ &\le 1/\big(1-((n-1)/2n)\big)\le 2,\end{aligned}$$

where i) $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$ denote the largest and smallest eigenvalues of a square symmetric matrix $A$ respectively, and ii) the second inequality is by noting that $V^{\mathsf{T}}V$ is an $n\times n$ matrix with 1's on the diagonal and at most $1/2n$ on the off-diagonal elements, and applying the Gershgorin circle theorem.

Now we can bound the quantity of interest. Note that

$$\begin{aligned}\left|\hat{u}\cdot\frac{u}{\|u\|}\right| &\ge |\hat{u}^{\mathsf{T}}(\hat{u}-(\hat{u}-u))| \ge 1-\|\hat{u}-u\| = 1-\Big\|\sum_i\alpha_i(\hat{v}_i-v_i)\Big\|\\ &\ge 1-\sum_i|\alpha_i|\|\hat{v}_i-v_i\| \ge 1-(\delta/\sqrt{2n})\sum_i|\alpha_i| \ge 1-\delta,\end{aligned}$$

where the last inequality is by noting $\|\alpha\|_1\le\sqrt{2n}$. ■

## Appendix B. On Constructing a Bounded Manifold Cover

Given a compact $n$-manifold $M\subset\mathbb{R}^D$ with condition number $1/\tau$, and some $0<\delta\le 1$. We can construct an $\alpha$-bounded $(\rho,\delta)$ cover $X$ of $M$ (with $\alpha\le 2^{13n}$ and $\rho\le\tau\delta/16n$) as follows.

Set $\rho\le\tau\delta/16n$ and pick a $(\rho/2)$-net $C$ of $M$ (that is $C\subset M$ such that, i. for $c,c'\in C$ such that $c\neq c'$, $\|c-c'\|\ge\rho/2$, ii. for all $p\in M$, exists $c\in C$ such that $\|c-p\|<\rho/2$). WLOG we shall assume that all points of $C$ are in the interior of $M$. Then, for each $c\in C$, define $M_{c,\rho/2}:=\{p\in M : \|p-c\|\le\rho/2\}$, and the orthogonal projection map $f_c: M_{c,\rho/2}\to T_cM$ that projects $M_{c,\rho/2}$ onto $T_cM$ (note that, cf. Lemma 19(i), $f_c$ is one-to-one). Note that $T_cM$ can be identified with $\mathbb{R}^n$ with the $c$ as the origin. We will denote the origin as $x_0^{(c)}$, that is, $x_0^{(c)}=f_c(c)$.

Now, let $B_c$ be any $n$-dimensional closed ball centered at the origin $x_0^{(c)}\in T_cM$ of radius $r>0$ that is completely contained in $f_c(M_{c,\rho/2})$ (that is, $B_c\subset f_c(M_{c,\rho/2})$). Pick a set of $n$ points $x_1^{(c)},\ldots,x_n^{(c)}$ on the surface of the ball $B_c$ such that $(x_i^{(c)}-x_0^{(c)})\cdot(x_j^{(c)}-x_0^{(c)})=0$ for $i\neq j$.

Define the bounded manifold cover as

$$X := \bigcup_{c \in C, i=0,\ldots,n} f_c^{-1}(x_i^{(c)}). \tag{6}$$

**Lemma 23** *Let $0 < \delta \leq 1$ and $\rho \leq \tau\delta/16n$. Let $C$ be a $(\rho/2)$-net of $M$ as described above, and $X$ be as in Equation* (6)*. Then $X$ forms a $2^{13n}$-bounded $(\rho, \delta)$ cover of $M$.*

**Proof** Pick any point $p \in M$ and define $X_p := \{x \in X : \|x - p\| < \rho\}$. Let $c \in C$ be such that $\|p - c\| < \rho/2$. Then $X_p$ has the following properties.

*Local spread criterion:* For $0 \leq i \leq n$, since $\|f_c^{-1}(x_i^{(c)}) - c\| \leq \rho/2$ (by construction), we have $\|f_c^{-1}(x_i^{(c)}) - p\| < \rho$. Thus, $f_c^{-1}(x_i^{(c)}) \in X_p$ (for $0 \leq i \leq n$). Now, for $1 \leq i \leq n$, noting that $D_G(f_c^{-1}(x_i^{(c)}), f_c^{-1}(x_0^{(c)})) \leq 2\|f_c^{-1}(x_i^{(c)}) - f_c^{-1}(x_0^{(c)})\| \leq \rho$ (cf. Lemma 18), we have that for the vector $\hat{v}_i^{(c)} := \frac{f_c^{-1}(x_i^{(c)}) - f_c^{-1}(x_0^{(c)})}{\|f_c^{-1}(x_i^{(c)}) - f_c^{-1}(x_0^{(c)})\|}$ and its (normalized) projection $v_i^{(c)} := \frac{x_i^{(c)} - x_0^{(c)}}{\|x_i^{(c)} - x_0^{(c)}\|}$ onto $T_cM$, $\left\|\hat{v}_i^{(c)} - v_i^{(c)}\right\| \leq \rho/\sqrt{2}\tau$ (cf. Lemma 21). Thus, for $i \neq j$, we have (recall, by construction, we have $v_i^{(c)} \cdot v_j^{(c)} = 0$)

$$\begin{aligned}
|\hat{v}_i^{(c)} \cdot \hat{v}_j^{(c)}| &= |(\hat{v}_i^{(c)} - v_i^{(c)} + v_i^{(c)}) \cdot (\hat{v}_j^{(c)} - v_j^{(c)} + v_j^{(c)})| \\
&= |(\hat{v}_i^{(c)} - v_i^{(c)}) \cdot (\hat{v}_j^{(c)} - v_j^{(c)}) + v_i^{(c)} \cdot (\hat{v}_j^{(c)} - v_j^{(c)}) + (\hat{v}_i^{(c)} - v_i^{(c)}) \cdot v_j^{(c)}| \\
&\leq \|(\hat{v}_i^{(c)} - v_i^{(c)})\|\|(\hat{v}_j^{(c)} - v_j^{(c)})\| + \|\hat{v}_i^{(c)} - v_i^{(c)}\| + \|\hat{v}_j^{(c)} - v_j^{(c)}\| \\
&\leq 3\rho/\sqrt{2}\tau \leq 1/2n.
\end{aligned}$$

*Covering criterion:* There exists $x \in X_p$, namely $f_c^{-1}(x_0^{(c)})$ $(= c)$, such that $\|p - x\| \leq \rho/2$.

*Local boundedness criterion:* Define $M_{p,3\rho/2} := \{q \in M : \|q - p\| < 3\rho/2\}$. Note that $X_p \subset \{f_c^{-1}(x_i^{(c)}) : c \in C \cap M_{p,3\rho/2}, 0 \leq i \leq n\}$. Now, using Lemma 20 we have that there exists a cover $N \subset M_{p,3\rho/2}$ of size at most $16^{3n}$ such that for any point $q \in M_{p,3\rho/2}$, there exists $n' \in N$ such that $\|q - n'\| < \rho/4$. Note that, by construction of $C$, there cannot be an $n' \in N$ such that it is within distance $\rho/4$ of two (or more) distinct $c, c' \in C$ (since otherwise the distance $\|c - c'\|$ will be less than $\rho/2$, contradicting the packing of $C$). Thus, $|C \cap M_{p,3\rho/2}| \leq 16^{3n}$. It follows that $|X_p| \leq (n+1)16^{3n} \leq 2^{13n}$.

*Tangent space approximation criterion:* Pick any $n+1$ (distinct) points in $X_p$ (viz. $x_0, \ldots, x_n$) that satisfy the local spread criterion, that is, $\left|\frac{x_i - x_0}{\|x_i - x_0\|} \cdot \frac{x_j - x_0}{\|x_j - x_0\|}\right| \leq 1/2n$ $(i \neq j)$. Let $\hat{T}_p$ be the $n$-dimensional affine space passing through $x_0, \ldots, x_n$ (note that $\hat{T}_p$ does not necessarily pass through $p$). Then, for any unit vector $\hat{u} \in \hat{T}_p$, we need to show that its projection $u_p$ onto $T_pM$ has the property $|\hat{u} \cdot \frac{u_p}{\|u_p\|}| \geq 1 - \delta$. Let $\theta$ be the angle between vectors $\hat{u}$ and $u_p$. Let $u_{x_0}$ be the projection of $\hat{u}$ onto $T_{x_0}M$, and $\theta_1$ be the angle between vectors $\hat{u}$ and $u_{x_0}$, and let $\theta_2$ be the angle between vectors $u_{x_0}$ (at $x_0$) and its parallel transport along the geodesic path to $p$ (see Figure 7). WLOG we can assume that $\theta_1$ and $\theta_2$ are at most $\pi/2$. Then, $\theta \leq \theta_1 + \theta_2 \leq \pi$. We get the bound on the individual angles as follows. By applying Lemma 22, $\cos(\theta_1) \geq 1 - \delta/4$, and by applying Lemma 17, $\cos(\theta_2) \geq 1 - \delta/4$. Finally, by using Lemma 24, we have $\left|\hat{u} \cdot \frac{u_p}{\|u_p\|}\right| = \cos(\theta) \geq \cos(\theta_1 + \theta_2) \geq 1 - \delta$. ■

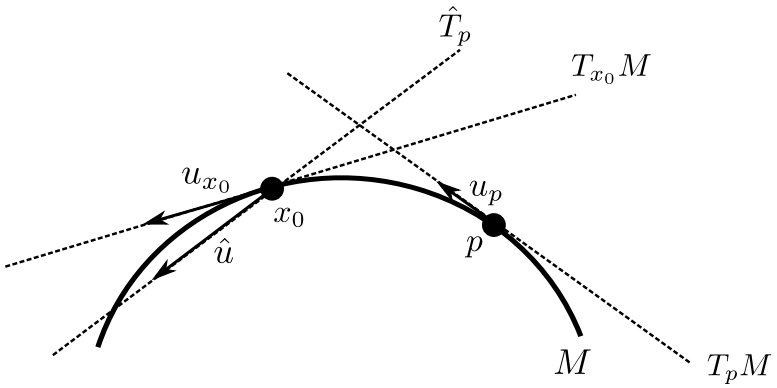



Figure 7: An example manifold $M$ with various quantities of interest. $\hat{T}_p$ is a sample-based approximation to $T_pM$ (using the nearby samples $x_0,\ldots,x_n$). The angle between $\hat{u}$ and its projection $u_p$ (into $T_pM$) is bounded by bounding $\hat{u}$ and its projection $u_{x_0}$ (into $T_{x_0}M$) and relating $u_{x_0}$ with its transport to $T_pM$.

**Lemma 24** *Let* $0 \leq \varepsilon_1, \varepsilon_2 \leq 1$. *If* $\cos\alpha \geq 1-\varepsilon_1$ *and* $\cos\beta \geq 1-\varepsilon_2$, *then* $\cos(\alpha+\beta) \geq 1-\varepsilon_1-\varepsilon_2-2\sqrt{\varepsilon_1\varepsilon_2}$.

**Proof** Applying the identity $\sin\theta = \sqrt{1-\cos^2\theta}$ immediately yields $\sin\alpha \leq \sqrt{2\varepsilon_1}$ and $\sin\beta \leq \sqrt{2\varepsilon_2}$. Now, $\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta \geq (1-\varepsilon_1)(1-\varepsilon_2) - 2\sqrt{\varepsilon_1\varepsilon_2} \geq 1-\varepsilon_1-\varepsilon_2-2\sqrt{\varepsilon_1\varepsilon_2}$. ■

**Remark 25** *A dense enough sample from M constitutes as a tangent space approximating cover. One can selectively prune the dense sampling to control the total number of points in each neighborhood, while still maintaining the cover properties, forming a bounded cover as per Definition 3.*

## Appendix C. Bounding the Number of Subsets $K$ in Embedding I

By construction (see the preprocessing stage of Embedding I), $K = \max_{x\in X} |X \cap B(x,2\rho)|$ (where $B(x,r)$ denotes a Euclidean ball centered at $x$ of radius $r$). That is, $K$ is the largest number of $x$'s ($\in X$) that are within a $2\rho$ ball of some $x \in X$.

Now, pick any $x \in X$ and consider the set $M_x := M \cap B(x,2\rho)$. Then, if $\rho \leq \tau/8$, $M_x$ can be covered by $2^{cn}$ balls of radius $\rho$ (see Lemma 20). By recalling that $X$ forms an $\alpha$-bounded $(\rho,\delta)$-cover, we have $|X \cap B(x,2\rho)| = |X \cap M_x| \leq \alpha 2^{cn}$ (where $c \leq 4$).

## Appendix D. Various Proofs

Here we provide proofs for the lemmas used throughout the text.

### D.1 Proof of Lemma 12

Since $R$ is a random orthoprojector from $\mathbb{R}^D$ to $\mathbb{R}^d$, it follows that

**Lemma 26 (random projection of $n$-manifolds—adapted from Theorem 1.5 of Clarkson (2008))** *Let $M$ be a $C_M$-regular $n$-manifold with condition number $1/\tau$. Let $\bar{R} := \sqrt{D/d}R$ be a scaling of $R$. Pick any $0 < \varepsilon \leq 1$ and $0 < \delta \leq 1$. If $d = \Omega\big(\varepsilon^{-2} n \log(C_M/\tau) + \varepsilon^{-2} n \log(1/\varepsilon) + \log(1/\delta)\big)$, then with probability at least $1-\delta$, for all $p, q \in M$*

$$(1-\varepsilon)\|p-q\| \leq \|\bar{R}p - \bar{R}q\| \leq (1+\varepsilon)\|p-q\|.$$

We apply this result with $\varepsilon = 1/4$. Then, for $d = \Omega(n\log(C_M/\tau))$, with probability at least $1 - 1/\text{poly}(n)$, $(3/4)\|p-q\| \leq \|\bar{R}p - \bar{R}q\| \leq (5/4)\|p-q\|$. Now let $\Phi : \mathbb{R}^D \to \mathbb{R}^d$ be defined as $\Phi x := (2/3)\bar{R}x = (2/3)(\sqrt{D/d})x$ (as per the lemma statement). Then we immediately get $(1/2)\|p-q\| \leq \|\Phi p - \Phi q\| \leq (5/6)\|p-q\|$.

Also note that for any $x \in \mathbb{R}^D$, we have $\|\Phi x\| = (2/3)(\sqrt{D/d})\|Rx\| \leq (2/3)(\sqrt{D/d})\|x\|$ (since $R$ is an orthoprojector).

Finally, for any point $p \in M$, a unit vector $u$ tangent to $M$ at $p$ can be approximated arbitrarily well by considering a sequence $\{p_i\}_i$ of points (in $M$) converging to $p$ (in $M$) such that $(p_i - p)/\|p_i - p\|$ converges to $u$. Since for all points $p_i$, $(1/2) \leq \|\Phi p_i - \Phi p\|/\|p_i - p\| \leq (5/6)$ (with high probability), it follows that $(1/2) \leq \|(D\Phi)_p(u)\| \leq (5/6)$.

### D.2 Proof of Corollary 13

Let $v_x^1$ and $v_x^n$ ($\in \mathbb{R}^n$) be the right singular vectors corresponding to singular values $\sigma_x^1$ and $\sigma_x^n$ respectively of the matrix $\Phi F_x$. Then, quickly note that $\sigma_x^1 = \|\Phi F_x v^1\|$, and $\sigma_x^n = \|\Phi F_x v^n\|$. Note that since $F_x$ is orthonormal, we have that $\|F_x v^1\| = \|F_x v^n\| = 1$. Now, since $F_x v^n$ is in the span of column vectors of $F_x$, by the sampling condition (cf. Definition 3), there exists a unit length vector $\vec{v}_x^n$ tangent to $M$ (at $x$) such that $|F_x v_x^n \cdot \vec{v}_x^n| \geq 1-\delta$. Thus, decomposing $F_x v_x^n$ into two vectors $a_x^n$ and $b_x^n$ such that $a_x^n \perp b_x^n$ and $a_x^n := (F_x v_x^n \cdot \vec{v}_x^n)\vec{v}_x^n$, we have (by Lemma 12)

$$\begin{aligned}
\sigma_x^n &= \|\Phi(F_x v^n)\| = \|\Phi((F_x v_x^n \cdot \vec{v}_x^n)\vec{v}_x^n) + \Phi b_x^n\| \\
&\geq (1-\delta)\,\|\Phi \vec{v}_x^n\| - \|\Phi b_x^n\| \\
&\geq (1-\delta)(1/2) - (2/3)\sqrt{2\delta D/d},
\end{aligned}$$

since $\|b_x^n\|^2 = \|F_x v_x^n\|^2 - \|a_x^n\|^2 \leq 1 - (1-\delta)^2 \leq 2\delta$ and $\|\Phi b_x^n\| \leq (2/3)(\sqrt{D/d})\|b_x^n\| \leq (2/3)\sqrt{2\delta D/d}$. Similarly decomposing $F_x v_x^1$ into two vectors $a_x^1$ and $b_x^1$ such that $a_x^1 \perp b_x^1$ and $a_x^1 := (F_x v_x^1 \cdot \vec{v}_x^1)\vec{v}_x^1$ (where $\vec{v}_x^1$ is a unit vector tangent to $M$ at $x$ such that $|F_x v_x^1 \cdot \vec{v}_x^1| \leq 1-\delta$), we have (by Lemma 12)

$$\begin{aligned}
\sigma_x^1 &= \|\Phi(F_x v_x^1)\| = \|\Phi((F_x v_x^1 \cdot \vec{v}_x^1)\vec{v}_x^1) + \Phi b_x^1\| \\
&\leq \big\|\Phi \vec{v}_x^1\big\| + \|\Phi b_x^1\| \\
&\leq (5/6) + (2/3)\sqrt{2\delta D/d},
\end{aligned}$$

where the last inequality is by noting $\|\Phi b_x^1\| \leq (2/3)\sqrt{2\delta D/d}$. Now, by our choice of $\delta$ ($\leq d/32D$), and by noting that $d \leq D$, the corollary follows.

### D.3 Proof of Lemma 14

We can simplify Equation (1) by recalling how the subsets $X^{(j)}$ were constructed (see preprocessing stage of Embedding I). Note that for any fixed $t$, at most one term in the set $\{\Lambda_{\Phi(x)}(t)\}_{x \in X^{(j)}}$ is non-zero. Thus,

$$\begin{aligned}\|(D\Psi)_t(u)\|^2 &= \sum_{k=1}^{d} u_k^2 + \sum_{k=1}^{n}\sum_{x\in X}\Lambda_{\Phi(x)}(t)\big(\cos^2(\omega(C^x t)_k)(C^x u)_k^2 + \sin^2(\omega(C^x t)_k)(C^x u)_k^2\big) \\ &\quad + \frac{1}{\omega}\Big[\underbrace{\big(\big(A^{k,x}_{\sin}(t)\big)^2 + \big(A^{k,x}_{\cos}(t)\big)^2\big)/\omega}_{\zeta_1} + \underbrace{2A^{k,x}_{\sin}(t)\sqrt{\Lambda_{\Phi(x)}(t)}\cos(\omega(C^x t)_k)(C^x u)_k}_{\zeta_2} \\ &\quad \underbrace{-2A^{k,x}_{\cos}(t)\sqrt{\Lambda_{\Phi(x)}(t)}\sin(\omega(C^x t)_k)(C^x u)_k}_{\zeta_3}\Big] \\ &= \|u\|^2 + \sum_{x\in X}\Lambda_{\Phi(x)}(t)\sum_{k=1}^{n}(C^x u)_k^2 + \zeta,\end{aligned}$$

where $\zeta := (\zeta_1+\zeta_2+\zeta_3)/\omega$. Noting that i) the terms $|A^{k,x}_{\sin}(t)|$ and $|A^{k,x}_{\cos}(t)|$ are at most $O(\alpha 16^n\sqrt{d}/\rho)$ (see Lemma 27), ii) $|(C^x u)_k| \le 4$, and iii) $\sqrt{\Lambda_{\Phi(x)}(t)} \le 1$, we can pick $\omega$ sufficiently large (say, $\omega \ge \Omega(n\alpha^2 16^n\sqrt{d}/\rho\varepsilon)$ such that $|\zeta| \le \varepsilon/2$ (where $\varepsilon$ is the isometry constant from our main theorem).

**Lemma 27** *For all k, x and t, the terms $|A^{k,x}_{\sin}(t)|$ and $|A^{k,x}_{\cos}(t)|$ are at most $O(\alpha 16^n\sqrt{d}/\rho)$.*

**Proof** We shall focus on bounding $|A^{k,x}_{\sin}(t)|$ (the steps for bounding $|A^{k,x}_{\cos}(t)|$ are similar). Note that

$$|A^{k,x}_{\sin}(t)| = \Big|\sum_{i=1}^{d} u_i \sin(\omega(C^x t)_k)\frac{d\Lambda^{1/2}_{\Phi(x)}(t)}{dt^i}\Big| \le \sum_{i=1}^{d}|u_i|\cdot\Big|\frac{d\Lambda^{1/2}_{\Phi(x)}(t)}{dt^i}\Big| \le \sqrt{\sum_{i=1}^{d}\Big|\frac{d\Lambda^{1/2}_{\Phi(x)}(t)}{dt^i}\Big|^2},$$

since $\|u\| \le 1$. Thus, we can bound $|A^{k,x}_{\sin}(t)|$ by $O(\alpha 16^n\sqrt{d}/\rho)$ by noting the following lemma. ■

**Lemma 28** *For all i, x and t, $|d\Lambda^{1/2}_{\Phi(x)}(t)/dt^i| \le O(\alpha 16^n/\rho)$.*

**Proof** Pick any $t \in \Phi(M)$, and let $p_0 \in M$ be (the unique element) such that $\Phi(p_0) = t$. Define $N_{p_0} := \{x \in X : \|\Phi(x) - \Phi(p_0)\| < \rho\}$ as the neighborhood around $p_0$. Fix an arbitrary $x_0 \in N_{p_0} \subset X$ (since if $x_0 \notin N_{p_0}$ then $d\Lambda^{1/2}_{\Phi(x_0)}(t)/dt^i = 0$), and consider the function

$$\Lambda^{1/2}_{\Phi(x_0)}(t) = \left(\frac{\lambda_{\Phi(x_0)}(t)}{\sum_{x\in N_{p_0}}\lambda_{\Phi(x)}(t)}\right)^{1/2} = \left(\frac{e^{-1/(1-(\|t-\Phi(x_0)\|^2/\rho^2))}}{\sum_{x\in N_{p_0}} e^{-1/(1-(\|t-\Phi(x)\|^2/\rho^2))}}\right)^{1/2}.$$

Define $A_t(x) := 1/(1-(\|t-\Phi(x)\|^2/\rho^2))$. Now, pick an arbitrary coordinate $i_0 \in \{1,\ldots,d\}$ and consider the (directional) derivative of this function

$$\frac{d\Lambda^{1/2}_{\Phi(x_0)}(t)}{dt^{i_0}} = \frac{1}{2}\big(\Lambda^{-1/2}_{\Phi(x_0)}(t)\big)\Big(\frac{d\Lambda_{\Phi(x_0)}(t)}{dt^{i_0}}\Big)$$

$$= \frac{\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^{1/2}}{2\Big(e^{-A_t(x_0)}\Big)^{1/2}} \Bigg[ \frac{\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)\Big(\frac{-2(t_{i_0}-\Phi(x_0)_{i_0})}{\rho^2}(A_t(x_0))^2\Big)\Big(e^{-A_t(x_0)}\Big)}{\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^2} - \frac{\Big(e^{-A_t(x_0)}\Big)\Big(\sum_{x\in N_{p_0}} \frac{-2(t_{i_0}-\Phi(x)_{i_0})}{\rho^2}(A_t(x))^2 e^{-A_t(x)}\Big)}{\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^2} \Bigg]$$

$$= \frac{\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)\Big(\frac{-2(t_{i_0}-\Phi(x_0)_{i_0})}{\rho^2}(A_t(x_0))^2\Big)\Big(e^{-A_t(x_0)}\Big)^{1/2}}{2\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^{1.5}} - \frac{\Big(e^{-A_t(x_0)}\Big)^{1/2}\Big(\sum_{x\in N_{p_0}} \frac{-2(t_{i_0}-\Phi(x)_{i_0})}{\rho^2}(A_t(x))^2 e^{-A_t(x)}\Big)}{2\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^{1.5}}.$$

Observe that the domain of the function $A_t$ is $\{x \in X : \|t-\Phi(x)\| < \rho\}$ and the range is $[1,\infty)$. Recalling that for any $\beta \geq 1$, $|\beta^2 e^{-\beta}| \leq 1$ and $|\beta^2 e^{-\beta/2}| \leq 3$, we have that $|A_t(\cdot)^2 e^{-A_t(\cdot)}| \leq 1$ and $|A_t(\cdot)^2 e^{-A_t(\cdot)/2}| \leq 3$. Thus,

$$\begin{aligned}
\Big|\frac{d\Lambda^{1/2}_{\Phi(x_0)}(t)}{dt^{i_0}}\Big| &\leq \frac{3\cdot\Big|\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big|\cdot\Big|\frac{2(t_{i_0}-\Phi(x_0)_{i_0})}{\rho^2}\Big| + \Big|e^{-A_t(x_0)/2}\Big|\cdot\Big|\sum_{x\in N_{p_0}} \frac{2(t_{i_0}-\Phi(x)_{i_0})}{\rho^2}\Big|}{2\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^{1.5}} \\
&\leq \frac{(3)(2/\rho)\Big|\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big| + \Big|e^{-A_t(x_0)/2}\Big|\sum_{x\in N_{p_0}} (2/\rho)}{2\Big(\sum_{x\in N_{p_0}} e^{-A_t(x)}\Big)^{1.5}} \\
&\leq O(\alpha 16^n/\rho),
\end{aligned}$$

where the last inequality is by noting: i) $|N_{p_0}| \leq \alpha 16^n$ (since for all $x \in N_{p_0}$, $\|x-p_0\| \leq 2\rho$—cf. Lemma 12, $X$ is an $\alpha$-bounded cover, and by noting that for $\rho \leq \tau/8$, a ball of radius $2\rho$ can be covered by $16^n$ balls of radius $\rho$ on the given $n$-manifold—cf. Lemma 20), ii) $|e^{-A_t(x)}| \leq |e^{-A_t(x)/2}| \leq 1$ (for all $x$), and iii) $\sum_{x\in N_{p_0}} e^{-A_t(x)} \geq \Omega(1)$ (since our cover $X$ ensures that for any $p_0$, there exists $x \in N_{p_0} \subset X$ such that $\|p_0 - x\| \leq \rho/2$—see also Remark 4, and hence $e^{-A_t(x)}$ is non-negligible for some $x \in N_{p_0}$). ■

### D.4 Proof of Lemma 15

Note that by definition, $\|(D\Psi)_t(u)\|^2 = \|(D\Psi_{|X|,n})_t(u)\|^2$. Thus, using Equation (3) and expanding the recursion, we have

$$\begin{aligned}\|(D\Psi)_t(u)\|^2 &= \|(D\Psi_{|X|,n})_t(u)\|^2 \\ &= \|(D\Psi_{|X|,n-1})_t(u)\|^2 + \Lambda_{\Phi(x_{|X|})}(t)(C^{x_{|X|}}u)_n^2 + Z_{|X|,n} \\ &\vdots \\ &= \|(D\Psi_{0,n})_t(u)\|^2 + \Big[\sum_{i=1}^{|X|}\Lambda_{\Phi(x_i)}(t)\sum_{j=1}^{n}(C^{x_i}u)_j^2\Big] + \sum_{i,j}Z_{i,j}.\end{aligned}$$

Note that $(D\Psi_{i,0})_t(u) := (D\Psi_{i-1,n})_t(u)$. Now recalling that $\|(D\Psi_{0,n})_t(u)\|^2 = \|u\|^2$ (the base case of the recursion), all we need to show is that $|\sum_{i,j}Z_{i,j}| \le \varepsilon/2$. This follows directly from the lemma below.

**Lemma 29** *For any i, j, let* $\omega_{i,j} \ge \Omega\big((K_{i,j}+(\alpha 16^n/\rho))(nd|X|)^2/\varepsilon\big)$ *(as per the statement of Lemma 15), and let* $\varepsilon_0 \le O\big(\varepsilon/\sqrt{d}(n|X|)^2\big)$*. Then, for any i, j,* $|Z_{i,j}| \le \varepsilon/2n|X|$.

**Proof** Recall that (cf. Equation (3))

$$Z_{i,j} = \underbrace{\frac{1}{\omega_{i,j}^2}\sum_k\big(\zeta_{i,j}^{k,4}\big)^2}_{(a)} + \underbrace{2\sum_k\frac{\zeta_{i,j}^{k,4}}{\omega_{i,j}}\big(\zeta_{i,j}^{k,1}+\zeta_{i,j}^{k,2}+\zeta_{i,j}^{k,3}\big)}_{(b)} + \underbrace{2\sum_k\zeta_{i,j}^{k,1}\zeta_{i,j}^{k,2}}_{(c)} + \underbrace{2\sum_k\zeta_{i,j}^{k,1}\zeta_{i,j}^{k,3}}_{(d)}.$$

*Term (a):* Note that $|\sum_k(\zeta_{i,j}^{k,4})^2| \le O\big(d^3(K_{i,j}+(\alpha 16^n/\rho))^2\big)$ (cf. Lemma 30 (iv)). By our choice of $\omega_{i,j}$, we have term (a) at most $O(\varepsilon/n|X|)$.

*Term (b):* Note that $\big|\zeta_{i,j}^{k,1}+\zeta_{i,j}^{k,2}+\zeta_{i,j}^{k,3}\big| \le O(n|X|+(\varepsilon/dn|X|))$ (by noting Lemma 30 (i)-(iii), recalling the choice of $\omega_{i,j}$, and summing over all $i'$, $j'$). Thus, $\big|\sum_k\zeta_{i,j}^{k,4}(\zeta_{i,j}^{k,1}+\zeta_{i,j}^{k,2}+\zeta_{i,j}^{k,3})\big| \le O\big(\big(d^2(K_{i,j}+(\alpha 16^n/\rho))\big)\big(n|X|+(\varepsilon/dn|X|)\big)\big)$. Again, by our choice of $\omega_{i,j}$, term (b) is at most $O(\varepsilon/n|X|)$.

*Terms (c) and (d):* We focus on bounding term (c) (the steps for bounding term (d) are same). Note that $|\sum_k\zeta_{i,j}^{k,1}\zeta_{i,j}^{k,2}| \le 4|\sum_k\zeta_{i,j}^{k,1}(\eta_{i,j}(t))_k|$ (by combining the definition of $\zeta_{i,j}^1$ and $\zeta_{i,j}^2$ from Equation (3) with Lemma 32(b) and Corollary 13). Now, observe that $\big(\zeta_{i,j}^{k,1}\big)_{k=1,\ldots,2d+3}$ is a tangent vector with length at most $O(\sqrt{dn}|X|)$ (cf. Lemma 30 (i)). Thus, by noting that $\eta_{i,j}$ is almost normal (with quality of approximation $\varepsilon_0$), we have term (c) at most $O(\varepsilon/n|X|)$.

By choosing the constants in the order terms appropriately, we can get the lemma. ■

**Lemma 30** *Let* $\zeta_{i,j}^{k,1}$, $\zeta_{i,j}^{k,2}$, $\zeta_{i,j}^{k,3}$, *and* $\zeta_{i,j}^{k,4}$ *be as defined in Equation* (3)*. Then for all* $1 \le i \le |X|$ *and* $1 \le j \le n$*, we have*

*(i)* $|\zeta_{i,j}^{k,1}| \le 1 + 8n|X| + \sum_{i'=1}^{i}\sum_{j'=1}^{j-1}O(d(K_{i',j'}+(\alpha 16^n/\rho))/\omega_{i',j'})$,

*(ii)* $|\zeta_{i,j}^{k,2}| \le 4$,

*(iii)* $|\zeta_{i,j}^{k,3}| \leq 4$,

*(iv)* $|\zeta_{i,j}^{k,4}| \leq O(d(K_{i,j} + (\alpha 16^n/\rho)))$.

**Proof** First note for any $\|u\| \leq 1$ and for any $x_i \in X$, $1 \leq j \leq n$ and $1 \leq l \leq d$, we have $|\sum_l C_{j,l}^{x_i} u_l| = |(C^{x_i}u)_j| \leq 4$ (cf. Lemma 32 (b) and Corollary 13).

Noting that for all $i$ and $j$, $\|\eta_{i,j}\| = \|\nu_{i,j}\| = 1$, we have $|\zeta_{i,j}^{k,2}| \leq 4$ and $|\zeta_{i,j}^{k,3}| \leq 4$.

Observe that $\zeta_{i,j}^{k,4} = \sum_l u_l R_{i,j}^{k,l}$. For all $i$, $j$, $k$ and $l$, note that i) $\|d\eta_{i,j}(t)/dt^l\| \leq K_{i,j}$ and $\|d\nu_{i,j}(t)/dt^l\| \leq K_{i,j}$ and ii) $|d\lambda_{\Phi(x_i)}^{1/2}(t)/dt^l| \leq O(\alpha 16^n/\rho)$ (cf. Lemma 28). Thus we have $|\zeta_{i,j}^{k,4}| \leq O(d(K_{i,j} + (\alpha 16^n/\rho)))$.

Now for any $i$, $j$, note that $\zeta_{i,j}^{k,1} = \sum_l u_l d\Psi_{i,j-1}(t)/dt^l$. Thus by recursively expanding, $|\zeta_{i,j}^{k,1}| \leq 1 + 8n|X| + \sum_{i'=1}^{i}\sum_{j'=1}^{j-1} O(d(K_{i',j'} + (\alpha 16^n/\rho))/\omega_{i',j'})$. ■

### D.5 Proof of Lemma 16

We start by stating the following useful observations:

**Lemma 31** *Let $A$ be a linear operator such that $\max_{\|x\|=1}\|Ax\| \leq \delta_{\max}$. Let $u$ be a unit-length vector. If $\|Au\| \geq \delta_{\min} > 0$, then for any unit-length vector $v$ such that $|u \cdot v| \geq 1-\varepsilon$, we have*

$$1 - \frac{\delta_{\max}\sqrt{2\varepsilon}}{\delta_{\min}} \leq \frac{\|Av\|}{\|Au\|} \leq 1 + \frac{\delta_{\max}\sqrt{2\varepsilon}}{\delta_{\min}}.$$

**Proof** Let $v' = v$ if $u \cdot v > 0$, otherwise let $v' = -v$. Quickly note that $\|u - v'\|^2 = \|u\|^2 + \|v'\|^2 - 2u \cdot v' = 2(1 - u \cdot v') \leq 2\varepsilon$. Thus, we have,

i. $\|Av\| = \|Av'\| \leq \|Au\| + \|A(u - v')\| \leq \|Au\| + \delta_{\max}\sqrt{2\varepsilon}$,

ii. $\|Av\| = \|Av'\| \geq \|Au\| - \|A(u - v')\| \geq \|Au\| - \delta_{\max}\sqrt{2\varepsilon}$.

Noting that $\|Au\| \geq \delta_{\min}$ yields the result. ■

**Lemma 32** *Let $x_1, \ldots, x_n \in \mathbb{R}^D$ be a set of orthonormal vectors, $F := [x_1, \ldots, x_n]$ be a $D \times n$ matrix and let $\Phi$ be a linear map from $\mathbb{R}^D$ to $\mathbb{R}^d$ ($n \leq d \leq D$) such that for all non-zero $a \in span(F)$ we have $0 < \|\Phi a\| \leq \|a\|$. Let $U\Sigma V^\mathsf{T}$ be the thin SVD of $\Phi F$. Define $C = (\Sigma^{-2} - I)^{1/2}U^\mathsf{T}$. Then,*

*(a)* $\|C(\Phi a)\|^2 = \|a\|^2 - \|\Phi a\|^2$, *for any $a \in span(F)$,*

*(b)* $\|C\|^2 \leq (1/\sigma^n)^2$, *where $\|\cdot\|$ denotes the spectral norm of a matrix and $\sigma^n$ is the $n^{th}$ largest singular value of $\Phi F$.*

**Proof** Note that the columns of $FV$ form an orthonormal basis for the subspace spanned by columns of $F$, such that $\Phi(FV) = U\Sigma$. Thus, since $a \in \text{span}(F)$, let $y$ be such that $a = FVy$. Note that i) $\|a\|^2 = \|y\|^2$, ii) $\|\Phi a\|^2 = \|U\Sigma y\|^2 = y^\mathsf{T}\Sigma^2 y$. Now,

$$\|C\Phi a\|^2 \quad = \quad \|((\Sigma^{-2} - I)^{1/2}U^\mathsf{T})\Phi FVy\|^2$$

$$\begin{aligned} &= \|(\Sigma^{-2}-I)^{1/2}U^{\mathsf{T}}U\Sigma V^{\mathsf{T}}Vy\|^2 \\ &= \|(\Sigma^{-2}-I)^{1/2}\Sigma y\|^2 \\ &= y^{\mathsf{T}}y - y^{\mathsf{T}}\Sigma^2 y \\ &= \|a\|^2 - \|\Phi a\|^2. \end{aligned}$$

Now, consider $\|C\|^2$.

$$\begin{aligned} \|C\|^2 &\leq \|(\Sigma^{-2}-I)^{1/2}\|^2\|U^{\mathsf{T}}\|^2 \\ &\leq \max_{\|x\|=1} \|(\Sigma^{-2}-I)^{1/2}x\|^2 \\ &\leq \max_{\|x\|=1} x^{\mathsf{T}}\Sigma^{-2}x \\ &= \max_{\|x\|=1} \sum_i x_i^2/(\sigma^i)^2 \\ &\leq (1/\sigma^n)^2, \end{aligned}$$

where $\sigma^i$ are the (top $n$) singular values forming the diagonal matrix $\Sigma$. ■

**Lemma 33** *Let $M \subset \mathbb{R}^D$ be a compact Riemannian n-manifold with condition number $1/\tau$. Pick any $x \in M$ and let $F_x$ be any n-dimensional affine space with the property: for any unit vector $v_x$ tangent to M at x, and its projection $v_{xF}$ onto $F_x$, $\left|v_x \cdot \frac{v_{xF}}{\|v_{xF}\|}\right| \geq 1-\delta$. Then for any $p \in M$ such that $\|x-p\| \leq \rho \leq \tau/2$, and any unit vector v tangent to M at p, ($\xi := (2\rho/\tau)+\delta+2\sqrt{2\rho\delta/\tau}$)*

*i.* $\left|v \cdot \frac{v_F}{\|v_F\|}\right| \geq 1-\xi$,

*ii.* $\|v_F\|^2 \geq 1-2\xi$,

*iii.* $\|v_r\|^2 \leq 2\xi$,

*where $v_F$ is the projection of v onto $F_x$ and $v_r$ is the residual (i.e., $v = v_F + v_r$ and $v_F \perp v_r$).*

**Proof** Let $\gamma$ be the angle between $v_F$ and $v$. We will bound this angle.

Let $v_x$ (at $x$) be the parallel transport of $v$ (at $p$) via the (shortest) geodesic path via the manifold connection. Let the angle between vectors $v$ and $v_x$ be $\alpha$. Let $v_{xF}$ be the projection of $v_x$ onto the subspace $F_x$, and let the angle between $v_x$ and $v_{xF}$ be $\beta$. WLOG, we can assume that the angles $\alpha$ and $\beta$ are acute. Then, since $\gamma \leq \alpha+\beta \leq \pi$, we have that $\left|v \cdot \frac{v_F}{\|v_F\|}\right| = \cos\gamma \geq \cos(\alpha+\beta)$. We bound the individual terms $\cos\alpha$ and $\cos\beta$ as follows.

Now, since $\|p-x\| \leq \rho$, using Lemmas 17 and 18, we have $\cos(\alpha) = |v \cdot v_x| \geq 1-2\rho/\tau$. We also have $\cos(\beta) = \left|v_x \cdot \frac{v_{xF}}{\|v_{xF}\|}\right| \geq 1-\delta$. Then, using Lemma 24, we finally get $\left|v \cdot \frac{v_F}{\|v_F\|}\right| = |\cos(\gamma)| \geq 1-2\rho/\tau-\delta-2\sqrt{2\rho\delta/\tau} = 1-\xi$.

Also note since $1 = \|v\|^2 = (v \cdot \frac{v_F}{\|v_F\|})^2 \left\|\frac{v_F}{\|v_F\|}\right\|^2 + \|v_r\|^2$, we have $\|v_r\|^2 = 1 - \left(v \cdot \frac{v_F}{\|v_F\|}\right)^2 \leq 2\xi$, and $\|v_F\|^2 = 1-\|v_r\|^2 \geq 1-2\xi$. ■

Now we are in a position to prove Lemma 16. Let $v_F$ be the projection of the unit vector $v$ (at $p$) onto the subspace spanned by (the columns of) $F_x$ and $v_r$ be the residual (i.e., $v = v_F + v_r$ and $v_F \perp v_r$). Then, noting that $p$, $x$, $v$ and $F_x$ satisfy the conditions of Lemma 33 (with $\rho$ in the Lemma 33 replaced with $2\rho$ from the statement of Lemma 16), we have ($\xi := (4\rho/\tau) + \delta + 4\sqrt{\rho\delta/\tau}$)

a) $\left| v \cdot \frac{v_F}{\|v_F\|} \right| \geq 1 - \xi$,

b) $\|v_F\|^2 \geq 1 - 2\xi$,

c) $\|v_r\|^2 \leq 2\xi$.

We can now bound the required quantity $\|C^x u\|^2$. Note that

$$\begin{aligned} \|C^x u\|^2 &= \|C^x \Phi v\|^2 = \|C^x \Phi (v_F + v_r)\|^2 \\ &= \|C^x \Phi v_F\|^2 + \|C^x \Phi v_r\|^2 + 2C^x \Phi v_F \cdot C^x \Phi v_r \\ &= \underbrace{\|v_F\|^2 - \|\Phi v_F\|^2}_{(a)} + \underbrace{\|C^x \Phi v_r\|^2}_{(b)} + \underbrace{2C^x \Phi v_F \cdot C^x \Phi v_r}_{(c)} \end{aligned}$$

where the last equality is by observing $v_F$ is in the span of $F_x$ and applying Lemma 32 (a). We now bound the terms $(a)$,$(b)$, and $(c)$ individually.

*Term $(a)$:* Note that $1 - 2\xi \leq \|v_F\|^2 \leq 1$ and observing that $\Phi$ satisfies the conditions of Lemma 31 with $\delta_{\max} = (2/3)\sqrt{D/d}$, $\delta_{\min} = (1/2) \leq \|\Phi v\|$ (cf. Lemma 12) and $\left| v \cdot \frac{v_F}{\|v_F\|} \right| \geq 1 - \xi$, we have (recall $\|\Phi v\| = \|u\| \leq 1$)

$$\begin{aligned} \|v_F\|^2 - \|\Phi v_F\|^2 &\leq 1 - \|v_F\|^2 \left\| \Phi \frac{v_F}{\|v_F\|} \right\|^2 \\ &\leq 1 - (1 - 2\xi) \left\| \Phi \frac{v_F}{\|v_F\|} \right\|^2 \\ &\leq 1 + 2\xi - \left\| \Phi \frac{v_F}{\|v_F\|} \right\|^2 \\ &\leq 1 + 2\xi - \left(1 - (4/3)\sqrt{2\xi D/d}\right)^2 \|\Phi v\|^2 \\ &\leq 1 - \|u\|^2 + \left(2\xi + (8/3)\sqrt{2\xi D/d}\right), \end{aligned} \tag{7}$$

where the fourth inequality is by using Lemma 31. Similarly, in the other direction

$$\begin{aligned} \|v_F\|^2 - \|\Phi v_F\|^2 &\geq 1 - 2\xi - \|v_F\|^2 \left\| \Phi \frac{v_F}{\|v_F\|} \right\|^2 \\ &\geq 1 - 2\xi - \left\| \Phi \frac{v_F}{\|v_F\|} \right\|^2 \\ &\geq 1 - 2\xi - \left(1 + (4/3)\sqrt{2\xi D/d}\right)^2 \|\Phi v\|^2 \\ &\geq 1 - \|u\|^2 - \left(2\xi + (32/9)\xi(D/d) + (8/3)\sqrt{2\xi D/d}\right). \end{aligned} \tag{8}$$

*Term* $(b)$*:* Note that for any $x$, $\|\Phi x\| \leq (2/3)(\sqrt{D/d})\|x\|$. We can apply Lemma 32 (b) with $\sigma_x^n \geq 1/4$ (cf. Corollary 13) and noting that $\|v_r\|^2 \leq 2\xi$, we immediately get

$$0 \leq \|C^x \Phi v_r\|^2 \leq 4^2 \cdot (4/9)(D/d)\|v_r\|^2 \leq (128/9)(D/d)\xi. \tag{9}$$

*Term* $(c)$*:* Recall that for any $x$, $\|\Phi x\| \leq (2/3)(\sqrt{D/d})\|x\|$, and using Lemma 32 (b) we have that $\|C^x\|^2 \leq 16$ (since $\sigma_x^n \geq 1/4$—cf. Corollary 13).

Now let $a := C^x \Phi v_F$ and $b := C^x \Phi v_r$. Then $\|a\| = \|C^x \Phi v_F\| \leq \|C^x\| \|\Phi v_F\| \leq 4$ (since $\|C^x\| \leq 4$, and noting that $v_F$ is vector in the column-span of $F_x$ such that $\|v_F\| \leq 1$ and the largest singular value of $\Phi F_x$ is at most 1 by Corollary 13), and $\|b\| = \|C^x \Phi v_r\| \leq (8/3)\sqrt{2\xi D/d}$ (see Equation (9)).

Thus, $|2a \cdot b| \leq 2\|a\|\|b\| \leq 2 \cdot 4 \cdot (8/3)\sqrt{2\xi D/d} = (64/3)\sqrt{2\xi D/d}$. Equivalently,

$$-(64/3)\sqrt{2\xi D/d} \leq 2C^x \Phi v_F \cdot C^x \Phi v_r \leq (64/3)\sqrt{2\xi D/d}. \tag{10}$$

Combining (7)-(10), and noting $d \leq D$, yields the lemma.

## Appendix E. Computing the Normal Vectors

The success of the second embedding technique crucially depends upon finding (at each iteration step) a pair of mutually orthogonal unit vectors that are normal to the embedding of manifold $M$ (from the previous iteration step) at a given point $p$. At a first glance finding such normal vectors seems infeasible since we only have access to a finite size sample $X$ from $M$. The saving grace comes from noting that the corrections are applied to the $n$-dimensional manifold $\Phi(M)$ that is actually a *submanifold* of $d$-dimensional space $\mathbb{R}^d$. Let us denote this space $\mathbb{R}^d$ as a flat $d$-manifold $N$ (containing our manifold of interest $\Phi(M)$). Note that even though we only have partial information about $\Phi(M)$ (since we only have samples from it), we have full information about $N$ (since it is the entire space $\mathbb{R}^d$). What it means is that given some point of interest $\Phi p \in \Phi(M) \subset N$, finding a vector normal to $N$ (at $\Phi p$) automatically is a vector normal to $\Phi(M)$ (at $\Phi p$). Of course, to find two mutually orthogonal normals to a $d$-manifold $N$, $N$ itself needs to be embedded in a larger dimensional Euclidean space (although embedding into $d+2$ should suffice, for computational reasons we will embed $N$ into Euclidean space of dimension $2d+3$). This is precisely the first thing we do before applying any corrections (cf. Step 2 of Embedding II in Section 4). See Figure 8 for an illustration of the setup before finding any normals.

Now for every iteration of the algorithm, note that we have complete knowledge of $N$ and exactly what function (namely $\Psi_{i,j}$ for iteration $i$, $j$) is being applied to $N$. Thus with additional computation effort, one can compute the necessary normal vectors.

More specifically, We can estimate a pair of mutually orthogonal unit vectors that are normal to $\Psi_{i,j}(N)$ at $\Phi p$ (for any step $i$, $j$) as follows.

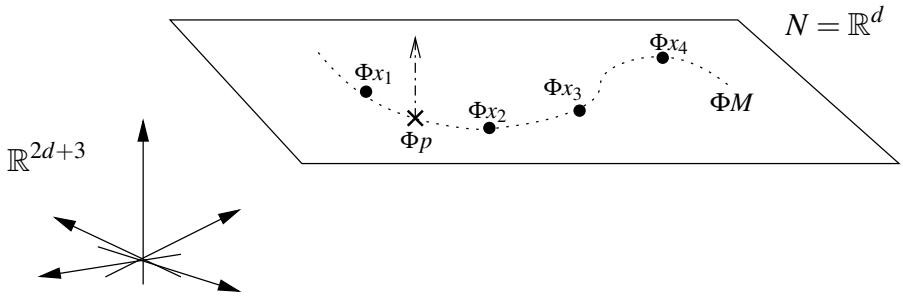



Figure 8: Basic setup for computing the normals to the underlying $n$-manifold $\Phi M$ at the point of interest $\Phi p$. Observe that even though it is difficult to find vectors normal to $\Phi M$ at $\Phi p$ within the containing space $\mathbb{R}^d$ (because we only have a finite-size sample from $\Phi M$, viz. $\Phi x_1$, $\Phi x_2$, etc.), we can treat the point $\Phi p$ as part of the bigger ambient manifold $N$ ($= \mathbb{R}^d$, that contains $\Phi M$) and compute the desired normals in a space that contains $N$ itself. Now, for each $i$, $j$ iteration of Algorithm II, $\Psi_{i,j}$ acts on the entire $N$, and since we have complete knowledge about $N$, we can compute the desired normals.

**Algorithm 4** Compute Normal Vectors

**Preprocessing Stage:**

1: Let $\eta_{i,j}^{\text{rand}}$ and $\nu_{i,j}^{\text{rand}}$ be vectors in $\mathbb{R}^{2d+3}$ drawn independently at random from the surface of the unit-sphere (for $1 \leq i \leq |X|$, $1 \leq j \leq n$).

**Compute Normals:** For any point of interest $p \in M$, let $t := \Phi p$ denote its projection into $\mathbb{R}^d$. Now, for any iteration $i, j$ (where $1 \leq i \leq |X|$, and $1 \leq j \leq n$), we shall assume that $\Psi_{i,j-1}$ (cf. Step 3) from the previous iteration $i, j-1$ is already given. Then we can compute the (approximated) normals $\eta_{i,j}(t)$ and $\nu_{i,j}(t)$ for the iteration $i$, $j$ as follows.

1: Let $\Delta > 0$ be the quality of approximation.
2: **for** $k = 1, \ldots, d$ **do**
3: Approximate the $k^{\text{th}}$ tangent vector as

$$T^k := \frac{\Psi_{i,j-1}(t + \Delta e^k) - \Psi_{i,j-1}(t)}{\Delta},$$

where $\Psi_{i,j-1}$ is as defined in Section 5.3, and $e^k$ is the $k^{\text{th}}$ standard vector.
4: **end for**
5: Let $\eta = \eta_{i,j}^{\text{rand}}$, and $\nu = \nu_{i,j}^{\text{rand}}$.
6: Use Gram-Schmidt orthogonalization process to extract $\hat{\eta}$ (from $\eta$) that is orthogonal to vectors $\{T^1, \ldots, T^d\}$.
7: Use Gram-Schmidt orthogonalization process to extract $\hat{\nu}$ (from $\nu$) that is orthogonal to vectors $\{T^1, \ldots, T^d, \hat{\eta}\}$.
8: **return** $\hat{\eta}/\|\hat{\eta}\|$ and $\hat{\nu}/\|\hat{\nu}\|$ as mutually orthogonal unit vectors that are approximately normal to $\Psi_{i,j-1}(\Phi M)$ at $\Psi_{i,j-1}(t)$.

A few remarks are in order.

**Remark 34** *The choice of target dimension of size* $2d+3$ *(instead of* $d+2$*) ensures that a pair of random unit-vectors* $\eta$ *and* $\nu$ *are not parallel to any vector in the tangent bundle of* $\Psi_{i,j-1}(N)$ *with probability* 1*. This follows from Sard's theorem (see, e.g., Milnor, 1972), and is the key observation in reducing the embedding size in Whitney's embedding (Whitney, 1936). This also ensures that our orthogonalization process (Steps 6 and 7) will not result in a null vector.*

**Remark 35** *By picking* $\Delta$ *sufficiently small, we can approximate the normals* $\eta$ *and* $\nu$ *arbitrarily well by approximating the tangents* $T^1,\ldots,T^d$ *well.*

**Remark 36** *For each iteration* $i,j$*, the vectors* $\hat{\eta}/\|\hat{\eta}\|$ *and* $\hat{\nu}/\|\hat{\nu}\|$ *that are returned (in Step 8) are a smooth modification to the starting vectors* $\eta_{i,j}^{\text{rand}}$ *and* $\nu_{i,j}^{\text{rand}}$ *respectively. Now, since we use the same starting vectors* $\eta_{i,j}^{\text{rand}}$ *and* $\nu_{i,j}^{\text{rand}}$ *regardless of the point of application (*$t=\Phi p$*), it follows that the respective directional derivatives of the returned vectors are bounded as well.*

By noting Remarks 35 and 36, the approximate normals we return satisfy the conditions needed for Embedding II (see our discussion in Section 5.3).

## References


R. Baraniuk and M. Wakin. Random projections of smooth manifolds. *Foundations of Computational Mathematics (FoCM)*, 9(1):51–77, 2009.

M. Bernstein, V. de Silva, J. Langford, and J. Tenebaum. Graph approximations to geodesics on embedded manifolds. *Techical Report*, 2000.

K. Clarkson. Tighter bounds for random projections of manifolds. *Symposium on Computational Geometry (SoCG)*, 24:39–49, 2008.

S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. *ACM Symposium on Theory of Computing (STOC)*, pages 537–546, 2008.

T. Dey, J. Giesen, S. Goswami, and W. Zhao. Shape dimension and approximation from samples. *Symposium on Discrete Algorithms (SODA)*, pages 772–780, 2002.

H. Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93(3):418–491, 1959.

J. Giesen and U. Wagner. Shape dimension and intrinsic metric from samples of manifolds with high co-dimension. *Symposium on Computational Geometry (SoCG)*, pages 329–337, 2003.

J. Milnor. *Topology from the Differential Viewpoint*. University of Virginia Press, 1972.

J. Nash. $C^1$ isometric imbeddings. *Annals of Mathematics*, 60(3):383–396, 1954.

P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete Computational Geometry*, 39(1):419–441, 2008.

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

N. Verma. A note on random projections for preserving paths on a manifold. *UC San Diego, Tech. Report CS2011-0971*, 2011.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing, Theory and Applications*. 2010.

H. Whitney. Differentiable manifolds. *Annals of Mathematics*, 37:645–680, 1936.

Journal of Machine Learning Research 14 (2013) 2449-2485 Submitted 4/12; Revised 3/13; Published 8/13

# Supervised Feature Selection in Graphs with Path Coding Penalties and Network Flows

**Julien Mairal**[*] JULIEN.MAIRAL@INRIA.FR
**Bin Yu**[†] BINYU@STAT.BERKELEY.EDU
*Department of Statistics*
*University of California*
*Berkeley, CA 94720-1776, USA.*

**Editor:** Ben Taskar

## Abstract

We consider supervised learning problems where the features are embedded in a graph, such as gene expressions in a gene network. In this context, it is of much interest to automatically select a subgraph with few connected components; by exploiting prior knowledge, one can indeed improve the prediction performance or obtain results that are easier to interpret. Regularization or penalty functions for selecting features in graphs have recently been proposed, but they raise new algorithmic challenges. For example, they typically require solving a combinatorially hard selection problem among all connected subgraphs. In this paper, we propose computationally feasible strategies to select a sparse and well-connected subset of features sitting on a directed acyclic graph (DAG). We introduce structured sparsity penalties over paths on a DAG called "path coding" penalties. Unlike existing regularization functions that model long-range interactions between features in a graph, path coding penalties are tractable. The penalties and their proximal operators involve path selection problems, which we efficiently solve by leveraging network flow optimization. We experimentally show on synthetic, image, and genomic data that our approach is scalable and leads to more connected subgraphs than other regularization functions for graphs.

**Keywords:** convex and non-convex optimization, network flow optimization, graph sparsity

## 1. Introduction

Supervised sparse estimation problems have been the topic of much research in statistical machine learning and signal processing. In high dimensional settings, restoring a signal or learning a model is often difficult without a priori knowledge. When the solution is known beforehand to be sparse—that is, has only a few non-zero coefficients, regularizing with sparsity-inducing penalties has been shown to provide better prediction and solutions that are easier to interpret. For that purpose, non-convex penalties and greedy algorithms have been proposed (Akaike, 1973; Schwarz, 1978; Rissanen, 1978; Mallat and Zhang, 1993; Fan and Li, 2001). More recently, convex relaxations such as the $\ell_1$-norm (Tibshirani, 1996; Chen et al., 1999) and efficient algorithms have been developed (Osborne et al., 2000; Nesterov, 2007; Beck and Teboulle, 2009; Wright et al., 2009).

In this paper, we consider supervised learning problems where more information is available than just sparsity of the solution. More precisely, we assume that the features (or predictors) can be

∗. Present address: LEAR Project-Team, INRIA Grenoble Rhône-Alpes, France.
†. Also in the department of Electrical Engineering & Computer Science.

©2013 Julien Mairal and Bin Yu.

identified to the vertices of a graph, such as gene expressions in a gene network. In this context, it can be desirable to take into account the graph structure in the regularization (Rapaport et al., 2007). In particular, we are interested in automatically identifying a subgraph with few connected components (Jacob et al., 2009; Huang et al., 2011), groups of genes involved in a disease for example. There are two equally important reasons for promoting the connectivity of the problem solution: either connectivity is a prior information, which might improve the prediction performance, or connected components may be easier to interpret than isolated variables.

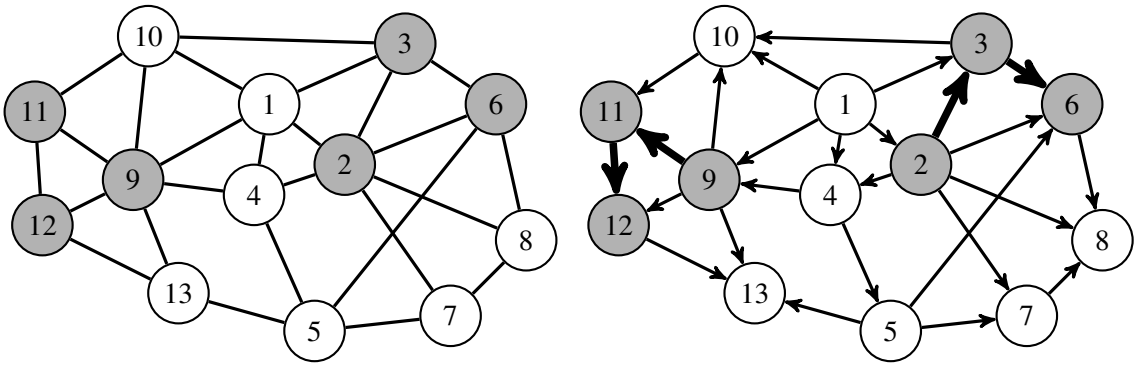
Formally, let us consider a supervised sparse estimation problem involving $p$ features, and let us assume that we are given an undirected or directed graph $G = (V,E)$, where $V$ is a vertex set identified to $\{1,\dots,p\}$, and $E \subseteq V \times V$ is an arc (edge) set. Classical empirical risk minimization problems can be formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^p} [L(\mathbf{w}) + \lambda\Omega(\mathbf{w})], \tag{1}$$

where $\mathbf{w}$ is a weight vector in $\mathbb{R}^p$, which we wish to estimate; $L : \mathbb{R}^p \to \mathbb{R}$ is a convex loss function, and $\Omega : \mathbb{R}^p \to \mathbb{R}$ is a regularization function. In order to obtain a sparse solution, $\Omega$ is often chosen to be the $\ell_0$- (cardinality of the support) or $\ell_1$-penalty. In this paper, we are also interested in encouraging the sparsity pattern of $\mathbf{w}$ (the set of non-zero coefficients) to form a subgraph of $G$ with few connected components.

To the best of our knowledge, penalties promoting the connectivity of sparsity patterns in a graph can be classified into two categories. The ones of the first category involve pairwise interactions terms between vertices linked by an arc (Cehver et al., 2008; Jacob et al., 2009; Chen et al., 2011); each term encourages two neighbors in the graph to be simultaneously selected. Such regularization functions usually lead to tractable optimization problems, but they do not model long-range interactions between variables in the graph, and they do not promote large connected components. Penalties from the second category are more complex, and directly involve hard combinatorial problems (Huang et al., 2011). As such, they cannot be used without approximations. The problem of finding tractable penalties that model long-range interactions is therefore acute. The main contribution of our paper is a solution to this problem when the graph is *directed and acyclic*.

Of much interest to us are the non-convex penalty of Huang et al. (2011) and the convex penalty of Jacob et al. (2009). Given a pre-defined set of possibly overlapping groups of variables $\mathcal{G}$, these two structured sparsity-inducing regularization functions encourage a sparsity pattern to be *in the union of a small number of groups from* $\mathcal{G}$. Both penalties induce a similar regularization effect and are strongly related to each other. In fact, we show in Section 3 that the penalty of Jacob et al. (2009) can be interpreted as a convex relaxation of the non-convex penalty of Huang et al. (2011). These two penalties go beyond classical unstructured sparsity, but they are also complex and raise new challenging combinatorial problems. For example, Huang et al. (2011) define $\mathcal{G}$ as the set of all connected subgraphs of $G$, which leads to well-connected solutions but also leads to intractable optimization problems; the latter are approximately addressed by Huang et al. (2011) with greedy algorithms. Jacob et al. (2009) choose a different strategy and define $\mathcal{G}$ as the pairs of vertices linked by an arc, which, as a result, encourages neighbors in the graph to be simultaneously selected. This last formulation is computationally tractable, but does not model long-range interactions between features. Another suggestion from Jacob et al. (2009) and Huang et al. (2011) consists of defining $\mathcal{G}$ as the set of connected subgraphs up to a size $k$. The number of such subgraphs is however exponential in $k$, making this approach difficult to use even for small subgraph sizes ($k = 3,4$) as




(a) Sparsity pattern in an undirected graph.

(b) Selected paths in a DAG.

Figure 1: Left (a): an undirected graph. A sparsity pattern forming a subgraph with two connected components is represented by gray nodes. Right (b): when the graph is a DAG, the sparsity pattern is covered by two paths $(2,3,6)$ and $(9,11,12)$ represented by bold arrows.

soon as the graph is large ($p \approx 10000$) and connected enough.[1] These observations naturally raise the question: *can we replace connected subgraphs by another structure that is rich enough to model long-range interactions in the graph and leads to computationally feasible penalties?*

When the graph $G$ is directed and acyclic, we propose a solution built upon two ideas. First, we use in the penalty framework of Jacob et al. (2009) and Huang et al. (2011) a novel group structure $\mathcal{G}_p$ that contains *all the paths* in $G$; a path is defined as a sequence of vertices $(v_1,\ldots,v_k)$ such that for all $1 \leq i < k$, we have $(v_i, v_{i+1}) \in E$. The second idea is to use appropriate costs for each path (the "price" one has to pay to select a path), which, as we show in the sequel, allows us to leverage network flow optimization. We call the resulting regularization functions "path coding" penalties. They go beyond pairwise interactions between vertices and model long-range interactions between the variables in the graph. They encourage sparsity patterns forming subgraphs that can be covered by a small number of paths, therefore promoting connectivity of the solution. We illustrate the "path coding" concept for DAGs in Figure 1. Even though the number of paths in a DAG is exponential in the graph size, we map the *path selection* problems our penalties involve to network flow formulations (see Ahuja et al., 1993; Bertsekas, 1998), which can be solved in polynomial time. As shown in Section 3, we build minimum cost flow formulations such that sending a positive amount of flow along a path for minimizing a cost is equivalent to selecting the path. This allows us to efficiently compute the penalties and their proximal operators, a key tool to address regularized problems (see Bach et al., 2012, for a review).

Therefore, we make in this paper a new link between structured graph penalties in DAGs and network flow optimization. The development of network flow optimization techniques has been very active from the 60's to the 90's (see Ford and Fulkerson, 1956; Goldberg and Tarjan, 1986; Ahuja et al., 1993; Goldberg, 1997; Bertsekas, 1998). They have attracted a lot of attention during the last decade in the computer vision community for their ability to solve large-scale combinatorial problems typically arising in image segmentation tasks (Boykov et al., 2001). Concretely, by

1. This issue was confirmed to us in a private communication with Laurent Jacob, and this was one of our main motivation for developing new algorithmic tools overcoming this problem.

mapping a problem at hand to a network flow formulation, one can possibly obtain fast algorithms to solve the original problem. Of course, such a mapping does not always exist or can be difficult to find. This is made possible in the context of path coding penalties thanks to decomposability properties of the path costs, which we make explicit in Section 3.

We remark that different network flow formulations have also been used recently for sparse estimation (Cehver et al., 2008; Chambolle and Darbon, 2009; Hoefling, 2010; Mairal et al., 2011). Cehver et al. (2008) combine for example sparsity and Markov random fields for signal reconstruction tasks. They introduce a non-convex penalty consisting of pairwise interaction terms between vertices of a graph, and their approach requires iteratively solving maximum flow problems. It has also been shown by Chambolle and Darbon (2009) and Hoefling (2010) that for the anisotropic total-variation penalty, called "fused lasso" in statistics, the solution to problem (1) can be obtained by solving a sequence of parametric maximum flow problems. The total-variation penalty can be useful to obtain piecewise constant solutions on a graph (see Chen et al., 2011). Finally, Mairal et al. (2011) have shown that the structured sparsity-inducing regularization function of Jenatton et al. (2011) is related to network flows in a similar way as the total variation penalty. Note that both Jacob et al. (2009) and Jenatton et al. (2011) use the same terminology of "group Lasso with overlapping groups", leading to some confusion in the literature. Yet, their works are significantly different and are in fact complementary: given a group structure $\mathcal{G}$, the penalty of Jacob et al. (2009) encourages solutions whose sparsity pattern is a *union* of a few groups, whereas the penalty of Jenatton et al. (2011) promotes an *intersection* of groups. It is natural to use the framework of Jacob et al. (2009) to encourage connectivity of a problem solution in a graph, e.g., by choosing $\mathcal{G}$ as the pairs of vertices linked by arc. It is however not obvious how to obtain this effect with the penalty of Jenatton et al. (2011). We discuss this question in more details in Appendix A.

To summarize, we have designed non-convex and convex penalty functions to do feature selection in directed acyclic graphs. Because our penalties involve an exponential number of variables, one for every path in the graph, existing optimization techniques cannot be used. To deal with this issue, we introduce network flow optimization tools that implicitly handle the exponential number of paths, allowing the penalties and their proximal operators to be computed in polynomial time. As a result, our penalties can model long-range interactions in the graph and are tractable.

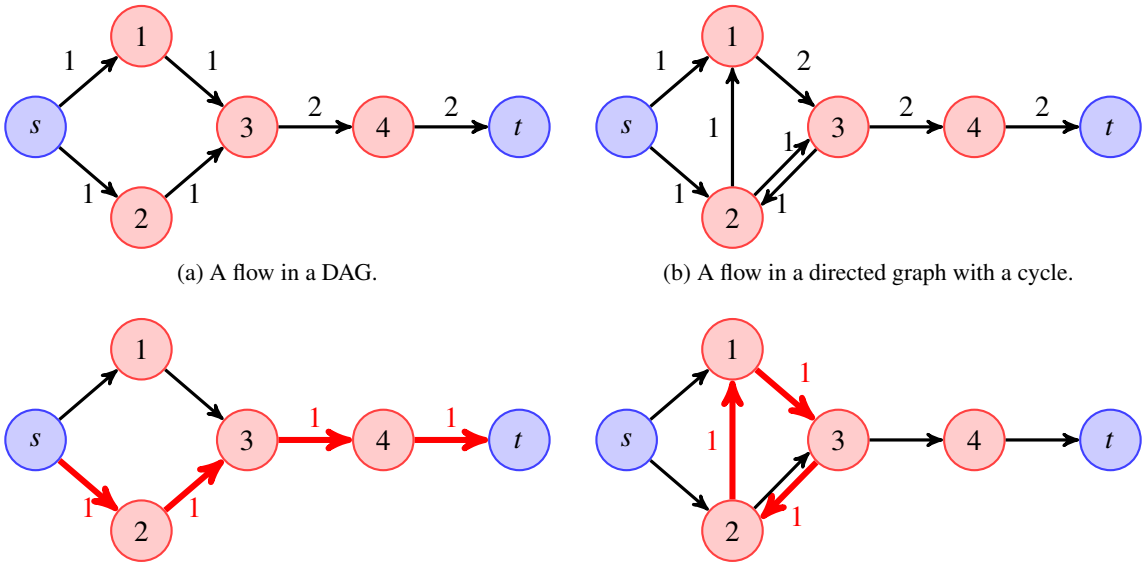
The paper is organized as follows: In Section 2, we present preliminary tools, notably a brief introduction to network flows. In Section 3, we propose the path coding penalties and optimization techniques for solving the corresponding sparse estimation problems. Section 4 is devoted to experiments on synthetic, genomic, and image data to demonstrate the benefits of path coding penalties over existing ones and the scalability of our approach. Section 5 concludes the paper.

## 2. Preliminaries

As we show later, our path coding penalties are related to the concept of flow in a graph. Since this concept is not widely used in the machine learning literature, we provide a brief overview of this topic in Section 2.1. In Section 2.2, we also present proximal gradient methods, which have become very popular for solving sparse regularized problems (see Bach et al., 2012).

### 2.1 Network Flow Optimization

Network flows have been well studied in the computer science community, and have led to efficient dedicated algorithms for solving particular linear programs (see Ahuja et al., 1993; Bertsekas, 1998).




(a) A flow in a DAG.

(b) A flow in a directed graph with a cycle.

(c) $(s,t)$-path flow in a DAG.

(d) A cycle flow in a directed graph.

Figure 2: Examples of flows in a graph. (a) The flow on the DAG can be interpreted as two units of flow sent from $s$ to $t$ along the paths $(s,1,3,4,t)$ and $(s,2,3,4,t)$. (b) The flow can be interpreted as two units of flow sent from $s$ to $t$ on the same paths as in (a) plus a unit of flow circulating along the cycle $(1,3,2,1)$. (c) $(s,t)$-path flow along the path $(s,2,3,4,t)$. (d) Cycle flow along $(1,3,2,1)$.

Let us consider a directed graph $G=(V,E)$ with two special nodes $s$ and $t$, respectively dubbed *source* and *sink*. A *flow* $f$ on the graph $G$ is defined as a non-negative function on arcs $[f_{uv}]_{(u,v)\in E}$ that satisfies two sets of linear constraints:

- **capacity constraints**: the value of the flow $f_{uv}$ on an arc $(u,v)$ in $E$ should satisfy the constraint $l_{uv} \leq f_{uv} \leq \delta_{uv}$, where $l_{uv}$ and $\delta_{uv}$ are respectively called lower and upper capacities;
- **conservation constraints**: the sum of incoming flow at a vertex is equal to the sum of outgoing flow except for the source $s$ and the sink $t$.

We present examples of flows in Figures 2a and 2b, and denote by $\mathcal{F}$ the set of flows on a graph $G$. We remark that with appropriate graph transformations, the flow definition we have given admits several variants. It is indeed possible to consider several source and sink nodes, capacity constraints on the amount of flow going through vertices, or several arcs with different capacities between two vertices (see more details in Ahuja et al., 1993).

Some network flow problems have attracted a lot of attention because of their wide range of applications, for example in engineering, physics, transportation, or telecommunications (see Ahuja et al., 1993). In particular, the *maximum flow problem* consists of computing how much flow can be sent from the source to the sink through the network (Ford and Fulkerson, 1956). In other words, it consists of finding a flow $f$ in $\mathcal{F}$ maximizing $\sum_{u\in V:(s,u)\in E} f_{su}$. Another more general problem, which is of interest to us, is the *minimum cost flow* problem. It consists of finding a flow $f$ in $\mathcal{F}$ minimizing

a linear cost $\sum_{(u,v)\in E} c_{uv} f_{uv}$, where every arc $(u,v)$ in $E$ has a cost $c_{uv}$ in $\mathbb{R}$. Both the maximum flow and minimum cost flow problems are linear programs, and can therefore be solved using generic linear programming tools, e.g., interior points methods (see Boyd and Vandenberghe, 2004; Nocedal and Wright, 2006). Dedicated algorithms exploiting the network structure of flows have however proven to be much more efficient. It has indeed been shown that minimum cost flow problems can be solved in strongly polynomial time—that is, an exact solution can be obtained in a finite number of steps that is polynomial in $|V|$ and $|E|$ (see Ahuja et al., 1993). More important, these dedicated algorithms are empirically efficient and can often handle large-scale problems (Goldberg and Tarjan, 1986; Goldberg, 1997; Boykov et al., 2001).

Among linear programs, network flow problems have a few distinctive features. The most striking one is the "physical" interpretation of a flow as a sum of quantities circulating in the network. The *flow decomposition theorem* (see Ahuja et al., 1993, Theorem 3.5) makes this interpretation more precise by saying that every flow vector can always be decomposed into a sum of $(s,t)$-path flows (units of flow sent from $s$ to $t$ along a path) and cycle flows (units of flow circulating along a cycle in the graph). We give examples of $(s,t)$-path and cycle flow in Figures 2c and 2d, and present examples of flows in Figures 2a and 2b along with their decompositions. Built upon the interpretation of flows as quantities circulating in the network, efficient algorithms have been developed, e.g., the classical *augmenting path* algorithm of Ford and Fulkerson (1956) for solving maximum flow problems. Another feature of flow problems is the locality of the constraints; each one only involves neighbors of a vertex in the graph. This locality is also exploited to design algorithms (Goldberg and Tarjan, 1986; Goldberg, 1997). Finally, minimum cost flow problems have a remarkable *integrality property*: a minimum cost flow problem where all capacity constraints are integers can be shown to have an integral solution (see Ahuja et al., 1993).

Later in our paper, we will map path selection problems to network flows by exploiting the flow decomposition theorem. In a nutshell, this apparently simple theorem has an interesting consequence: minimum cost flow problems can be seen from two equivalent viewpoints. Either one is looking for the value $f_{uv}$ of a flow on every arc $(u,v)$ of a graph minimizing the cost $\sum_{(u,v)\in E} c_{uv} f_{uv}$, or one is looking for the quantity of flow that should circulate on every $(s,t)$-path and cycle flow for minimizing the same cost. Of course, when the graph $G$ is a DAG, cycle flows do not exist. We will define flow problems such that selecting a path in the context of our path coding penalties is equivalent to sending some flow along a corresponding $(s,t)$-path. We will also exploit the *integrality property* to develop tools both adapted to non-convex penalties and convex ones, respectively involving discrete and continuous optimization problems. With these tools in hand, we will be able to deal efficiently with a simple class of optimization problems involving our path coding penalties. To deal with the more complex problem (1), we will need additional tools, which we now present.

### 2.2 Proximal Gradient Methods

Proximal gradient methods are iterative schemes for minimizing objective functions of the same form as (1), when the function $L$ is convex and differentiable with a Lipschitz continuous gradient. The simplest proximal gradient method consists of linearizing at each iteration the function $L$ around a current estimate $\tilde{\mathbf{w}}$, and this estimate is updated as the (unique by strong convexity) solution to

$$\min_{\mathbf{w}\in\mathbb{R}^p} \Big[ \underbrace{L(\tilde{\mathbf{w}}) + \nabla L(\tilde{\mathbf{w}})^\top (\mathbf{w}-\tilde{\mathbf{w}})}_{\text{linear approximation of } L} + \underbrace{\frac{\rho}{2}\|\mathbf{w}-\tilde{\mathbf{w}}\|_2^2}_{\text{quadratic term}} + \underbrace{\lambda\Omega(\mathbf{w})}_{\text{non-smooth part}} \Big], \tag{2}$$

which is assumed to be easier to solve than the original problem (1). The quadratic term keeps the update in a neighborhood where $L$ is close to its linear approximation, and the parameter $\rho$ is an upper bound on the Lipschitz constant of $\nabla L$. When $\Omega$ is convex, this scheme is known to converge to the solution to problem (1) and admits variants with optimal convergence rates among first-order methods (Nesterov, 2007; Beck and Teboulle, 2009). When $\Omega$ is non-convex, the guarantees are weak (finding the global optimum is out of reach), but it is easy to show that these updates can be seen as a majorization-minimization algorithm (Hunter and Lange, 2004) iteratively decreasing the value of the objective function (Wright et al., 2009; Mairal, 2013). When $\Omega$ is the $\ell_1$- or $\ell_0$-penalty, the optimization schemes (2) are respectively known as iterative soft- and hard-thresholding algorithms (Daubechies et al., 2004; Blumensath and Davies, 2009). Note that when $L$ is not differentiable, similar schemes exist, known as mirror-descent (Nemirovsky and Yudin, 1983).

Another insight about these methods can be obtained by rewriting sub-problem (2) as

$$\min_{\mathbf{w}\in\mathbb{R}^p} \left[ \frac{1}{2} \left\| \tilde{\mathbf{w}} - \frac{1}{\rho}\nabla L(\tilde{\mathbf{w}}) - \mathbf{w} \right\|_2^2 + \frac{\lambda}{\rho}\Omega(\mathbf{w}) \right].$$

When $\lambda = 0$, the solution is obtained by a classical gradient step $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} - (1/\rho)\nabla L(\tilde{\mathbf{w}})$. Thus, proximal gradient methods can be interpreted as a generalization of gradient descent algorithms when dealing with a nonsmooth term. They are, however, only interesting when problem (2) can be efficiently solved. Formally, we wish to be able to compute the *proximal operator* defined as:

**Definition 1 (Proximal Operator.)**
*The proximal operator associated with a regularization term $\lambda\Omega$, which we denote by $Prox_{\lambda\Omega}$, is the function that maps a vector $\mathbf{u} \in \mathbb{R}^p$ to the unique (by strong convexity) solution to*

$$\min_{\mathbf{w}\in\mathbb{R}^p} \left[ \frac{1}{2}\|\mathbf{u} - \mathbf{w}\|_2^2 + \lambda\Omega(\mathbf{w}) \right]. \tag{3}$$

Computing efficiently this operator has been shown to be possible for many penalties $\Omega$ (see Bach et al., 2012). We will show in the sequel that it is also possible for our path coding penalties.

## 3. Sparse Estimation in Graphs with Path Coding Penalties

We now present our path coding penalties, which exploit the structured sparsity frameworks of Jacob et al. (2009) and Huang et al. (2011). Because we choose a group structure $\mathcal{G}_p$ with an exponential number of groups, one for every path in the graph, the optimization techniques presented by Jacob et al. (2009) or Huang et al. (2011) cannot be used anymore. We will deal with this issue by introducing flow definitions of the path coding penalties.

### 3.1 Path Coding Penalties

The so-called "block coding" penalty of Huang et al. (2011) can be written for a vector $\mathbf{w}$ in $\mathbb{R}^p$ and any set $\mathcal{G}$ of groups of variables as

$$\varphi_{\mathcal{G}}(\mathbf{w}) \triangleq \min_{\mathcal{J}\subseteq\mathcal{G}} \Big\{ \sum_{g\in\mathcal{J}} \eta_g \ \text{ s.t. } \ \text{Supp}(\mathbf{w}) \subseteq \bigcup_{g\in\mathcal{J}} g \Big\}, \tag{4}$$

where the $\eta_g$'s are non-negative weights, and $\mathcal{J}$ is a subset of groups in $\mathcal{G}$ whose union covers the support of $\mathbf{w}$ formally defined as $\text{Supp}(\mathbf{w}) \triangleq \{j \in \{1,\dots,p\} : \mathbf{w}_j \neq 0\}$. When the weights $\eta_g$ are well

chosen, the non-convex penalty $\varphi_{\mathcal{G}}$ encourages solutions $\mathbf{w}$ whose support is in the union of a small number of groups; in other words, the cardinality of $\mathcal{J}$ should be small. We remark that Huang et al. (2011) originally introduce this regularization function under a more general information-theoretic point of view where $\varphi_{\mathcal{G}}$ is a code length (see Barron et al., 1998; Cover and Thomas, 2006), and the weights $\eta_g$ represent the number of bits encoding the fact that a group $g$ is selected.[2] One motivation for using $\varphi_{\mathcal{G}}$ is that the selection of a few groups might be easier to interpret than the selection of isolated variables. This formulation extends non-convex group sparsity regularization by allowing any group structure $\mathcal{G}$ to be considered. Nevertheless, a major drawback is that computing this non-convex penalty $\varphi_{\mathcal{G}}(\mathbf{w})$ for a general group structure $\mathcal{G}$ is difficult. Equation (4) is indeed an instance of a set cover problem, which is NP-hard (see Cormen et al., 2001), and appropriate approximations, e.g., greedy algorithms, have to be used in practice.

As often when dealing with non-convex penalties, one can either try to solve directly the corresponding non-convex problems or look for a convex relaxation. As we empirically show in Section 4, having both non-convex and convex variants of a penalty can be a significant asset. One variant can indeed outperform the other one in some situations, while being the other way around in some other cases. It is therefore interesting to look for a convex relaxation of $\varphi_{\mathcal{G}}$. We denote by $\eta$ the vector $[\eta_g]_{g\in\mathcal{G}}$ in $\mathbb{R}_+^{|\mathcal{G}|}$, and by $\mathbf{N}$ the binary matrix in $\{0,1\}^{p\times|\mathcal{G}|}$ whose columns are indexed by the groups $g$ in $\mathcal{G}$, such that the entry $\mathbf{N}_{jg}$ is equal to one when the index $j$ is in the group $g$, and zero otherwise. Equation (4) can be rewritten as a Boolean linear program, a form which will be more convenient in the rest of the paper:

$$\varphi_{\mathcal{G}}(\mathbf{w}) = \min_{\mathbf{x}\in\{0,1\}^{|\mathcal{G}|}} \left\{ \eta^\top \mathbf{x} \ \text{ s.t. } \ \mathbf{N}\mathbf{x} \geq \text{Supp}(\mathbf{w}) \right\}, \tag{5}$$

where, with an abuse of notation, $\text{Supp}(\mathbf{w})$ is here a vector in $\{0,1\}^p$ such that its $j$-th entry is 1 if $j$ is in the support of $\mathbf{w}$ and 0 otherwise. Let us also denote by $|\mathbf{w}|$ the vector in $\mathbb{R}_+^p$ obtained by replacing the entries of $\mathbf{w}$ by their absolute value. We can now consider a convex relaxation of $\varphi_{\mathcal{G}}$:

$$\psi_{\mathcal{G}}(\mathbf{w}) \triangleq \min_{\mathbf{x}\in\mathbb{R}_+^{|\mathcal{G}|}} \left\{ \eta^\top \mathbf{x} \ \text{ s.t. } \ \mathbf{N}\mathbf{x} \geq |\mathbf{w}| \right\}, \tag{6}$$

where not only the optimization problem above is a linear program, but in addition $\psi_{\mathcal{G}}$ is a convex function—in fact it can be shown to be a norm. Such a relaxation is classical and corresponds to the same mechanism relating the $\ell_0$- to the $\ell_1$-penalty, replacing $\text{Supp}(\mathbf{w})$ by $|\mathbf{w}|$. The next lemma tells us that we have in fact obtained a variant of the penalty introduced by Jacob et al. (2009).

**Lemma 2 (Relation Between $\psi_{\mathcal{G}}$ and the Penalty of Jacob et al. (2009).)**
*Suppose that any pattern in $\{0,1\}^p$ can be represented by a union of groups in $\mathcal{G}$. Then, the function $\psi_{\mathcal{G}}$ defined in (6) is equal to the penalty of Jacob et al. (2009) with $\ell_\infty$-norms.*

Note that Jacob et al. (2009) have introduced their penalty from a different perspective, and the link between (6) and their work is not obvious at first sight. In addition, their penalty involves a sum of $\ell_2$-norms, which needs to be replaced by $\ell_\infty$-norms for the lemma to hold. Hence, $\psi_{\mathcal{G}}$ is a

2. Note that Huang et al. (2011) do not directly use the function $\varphi_{\mathcal{G}}$ as a regularization function. The "coding complexity" they introduce for a vector $\mathbf{w}$ counts the number of bits to code the support of $\mathbf{w}$, which is achieved by $\varphi_{\mathcal{G}}$, but also use an $\ell_0$-penalty to count the number of bits encoding the values of the non-zero coefficients in $\mathbf{w}$.

"variant" of the penalty of Jacob et al. (2009). We give more details and the proof of this lemma in Appendix B.[3]

Now that $\varphi_{\mathcal{G}}$ and $\psi_{\mathcal{G}}$ have been introduced, we are interested in automatically selecting a small number of connected subgraphs from a directed acyclic graph $G = (V,E)$. In Section 1, we already discussed group structures $\mathcal{G}$ and introduced $\mathcal{G}_p$ the *set of paths in G*. As a result, the path coding penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ encourage solutions that are sparse while forming a subgraph that can be covered by a small number of paths. As we show in this section, this choice leads to tractable formulations when the weights $\eta_g$ for every path $g$ in $\mathcal{G}_p$ are appropriately chosen.

We will show in the sequel that a natural choice is to define for all $g$ in $\mathcal{G}_p$

$$\eta_g \triangleq \gamma + |g|, \tag{7}$$

where $\gamma$ is a new parameter encouraging the connectivity of the solution whereas $|g|$ encourages sparsity. It is indeed possible to show that when $\gamma = 0$, the functions $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ respectively become the $\ell_0$- and the $\ell_1$-penalties, therefore encouraging sparsity but not connectivity. On the other hand, when $\gamma$ is large and the term $|g|$ is negligible, $\varphi_{\mathcal{G}_p}(\mathbf{w})$ simply "counts" how many paths are required to cover the support of $\mathbf{w}$, thereby encouraging connectivity regardless of the sparsity of $\mathbf{w}$.

In fact, the choice (7) is a particular case of a more general class of weights $\eta_g$, which our algorithmic framework can handle. Let us enrich the original directed acyclic graph $G$ by introducing a source node $s$ and a sink node $t$. Formally, we define a new graph $G' = (V',E')$ with

$$\begin{aligned} V' &\triangleq V \cup \{s,t\}, \\ E' &\triangleq E \cup \{(s,v) : v \in V\} \cup \{(u,t) : u \in V\}. \end{aligned}$$

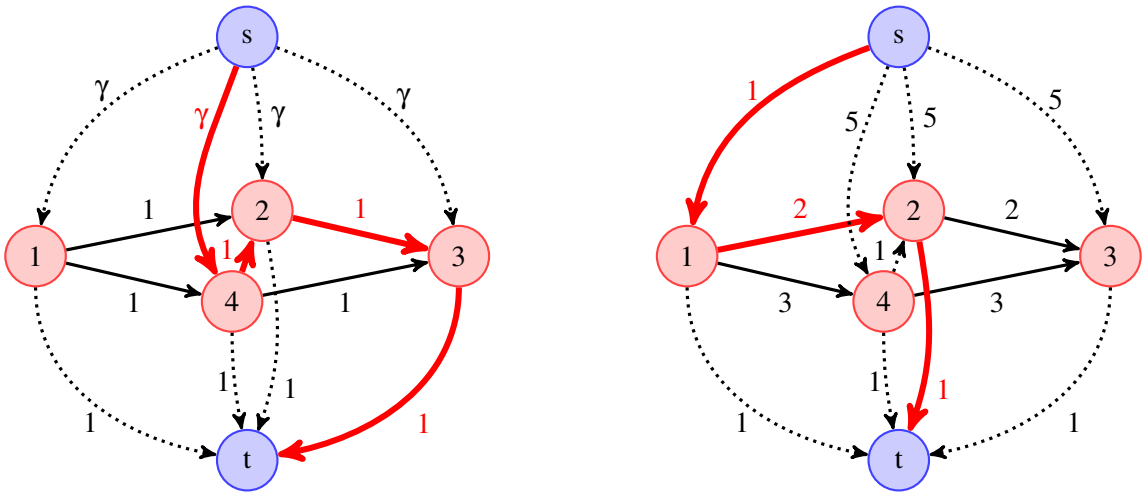
In plain words, the graph $G'$, which is a DAG, contains the graph $G$ and two nodes $s,t$ that are linked to every vertices of $G$. Let us also assume that some costs $c_{uv}$ in $\mathbb{R}$ are defined for all arcs $(u,v)$ in $E'$. Then, for a path $g = (u_1,u_2,\ldots,u_k)$ in $\mathcal{G}_p$, we define the weight $\eta_g$ as

$$\eta_g \triangleq c_{su_1} + \Big(\sum_{i=1}^{k-1} c_{u_i u_{i+1}}\Big) + c_{u_k t} = \sum_{(u,v)\in(s,g,t)} c_{uv}, \tag{8}$$

where the notation $(s,g,t)$ stands for the path $(s,u_1,u_2,\ldots,u_k,t)$ in $G'$. The decomposition of the weights $\eta_g$ as a sum of costs on $(s,t)$-paths of $G'$ (the paths $(s,g,t)$ with $g$ in $\mathcal{G}_p$) is a key component of the algorithmic framework we present next. The construction of the graph $G'$ is illustrated in Figures 3a and 3b for two cost configurations. We remark that the simple choice of weights (7) corresponds to the choice (8) with the costs $c_{su} = \gamma$ for all $u$ in $V$ and $c_{uv} = 1$ otherwise (see Figure 3a). Designing costs $c_{uv}$ that go beyond the simple choice (7) can be useful whenever one has additional knowledge about the graph structure. For example, we experimentally exploit this property in Section 4.2 to privilege or penalize paths $g$ in $\mathcal{G}_p$ starting from a particular vertex. This is illustrated in Figure 3b where the cost on the arc $(s,1)$ is much smaller than on the arcs $(s,2)$, $(s,3)$, $(s,4)$, therefore encouraging paths starting from vertex 1.

3. At the same time as us, Obozinski and Bach (2012) have studied a larger class of non-convex combinatorial penalties and their corresponding convex relaxations, obtaining in particular a more general result than Lemma 2, showing that $\psi_{\mathcal{G}}$ is the tightest convex relaxation of $\varphi_{\mathcal{G}}$.

Another interpretation connecting the path-coding penalties with coding lengths and random walks can be drawn using information theoretic arguments derived from Huang et al. (2011). We find these connections interesting, but for simplicity only present them in Appendix C. In the next sections, we address the following issues: (i) how to compute the penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ given a vector $\mathbf{w}$ in $\mathbb{R}^p$? (ii) how to optimize the objective function (1)? (iii) in the convex case (when $\Omega = \psi_{\mathcal{G}_p}$), can we obtain practical optimality guarantees via a duality gap? All of these questions will be answered using network flow and convex optimization, or algorithmic tools on graphs.




(a) Graph $G'$ with arcs costs and a path $g$ in bold red.

(b) Graph $G'$ with different arcs costs and a path $g$.

Figure 3: (a) $G'$ is obtained by adding a source $s$ and sink $t$ to a DAG with four nodes. The cost configuration is such that the weights $\eta_g$ satisfy $\eta_g = \gamma + |g|$. For example, for $g = (4,2,3)$, the sum of costs along $(s,g,t)$ is $\eta_g = \gamma + 3$. (b) Same graph $G'$ as (a) but with different costs. The weight $\eta_g$ associated to the path $g = (1,2)$ is the sum of costs along $(s,1,2,t)$—that is, $\eta_g = 4$.

### 3.2 Flow Definitions of the Path Coding Penalties

Before precisely stating the flow definitions of $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$, let us sketch the main ideas. The first key component is to transform the optimization problems (4) and (6) over the paths in $G$ into optimization problems over $(s,t)$-*path flows* in $G'$. We recall that $(s,t)$-path flows are defined as flow vectors carrying the same positive value on every arc of a path between $s$ and $t$. It intuitively corresponds to sending from $s$ to $t$ a positive amount of flow along a path, an interpretation we have presented in Figure 2 from Section 2.1. Then, we use the *flow decomposition theorem* (see Section 2.1), which provides two equivalent viewpoints for solving a minimum cost flow problem on a DAG. One should be either looking for the value $f_{uv}$ of a flow on every arc $(u,v)$ of the graph, or one should decide how much flow should be sent on every $(s,t)$-path.

We assume that a cost configuration $[c_{uv}]_{(u,v)\in E'}$ is available and that the weights $\eta_g$ are defined according to Equation (8). We denote by $\mathcal{F}$ the set of flows on $G'$. The second key component of our approach is the fact that the cost of a flow $f$ in $\mathcal{F}$ sending one unit from $s$ to $t$ along a path $g$ in $G$,

defined as $\sum_{(u,v)\in E'} f_{uv}c_{uv} = \sum_{(u,v)\in(s,g,t)} c_{uv}$ is exactly $\eta_g$, according to Equation (8). This enables us to reformulate our optimization problems (4) and (6) on paths in $G$ as optimization problems on $(s,t)$-path flows in $G'$, which in turn are equivalent to minimum cost flow problems and can be solved in polynomial time. Note that this equivalence does not hold when we have cycle flows (see Figure 2d), and this is the reason why we have assumed $G$ to be acyclic.

We can now formally state the mappings between the penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ on one hand, and network flows on the other hand. An important quantity in the upcoming propositions is the amount of flow going through a vertex $j$ in $V = \{1,\dots,p\}$, which we denote by

$$s_j(f) \triangleq \sum_{u\in V' : (u,j)\in E'} f_{uj}.$$

Our formulations involve capacity constraints and costs for $s_j(f)$, which can be handled by network flow solvers; in fact, a vertex can always be equivalently replaced in the network by two vertices, linked by an arc that carries the flow quantity $s_j(f)$ (Ahuja et al., 1993). The main propositions are presented below, and the proofs are given in Appendix D.

**Proposition 3 (Computing $\varphi_{\mathcal{G}_p}$.)**
*Let* $\mathbf{w}$ *be in* $\mathbb{R}^p$*. Consider the network* $G'$ *defined in Section 3.1 with costs* $[c_{uv}]_{(u,v)\in E'}$*, and define* $\eta_g$ *as in (8). Then,*

$$\varphi_{\mathcal{G}_p}(\mathbf{w}) = \min_{f\in\mathcal{F}} \left\{ \sum_{(u,v)\in E'} f_{uv}c_{uv} \;\; s.t. \;\; s_j(f) \geq 1,\ \forall j \in Supp(\mathbf{w}) \right\}, \tag{9}$$

*where* $\mathcal{F}$ *is the set of flows on* $G'$*. This is a minimum cost flow problem with some lower-capacity constraints, which can be computed in strongly polynomial time.*[4]

Given the definition of the penalty $\varphi_{\mathcal{G}}$ in Equation (5), computing $\varphi_{\mathcal{G}_p}$ seems challenging for two reasons: (i) Equation (5) is for a general group structure $\mathcal{G}$ a NP-hard Boolean linear program with $|\mathcal{G}|$ variables; (ii) the size of $\mathcal{G}_p$ is exponential in the graph size. Interestingly, Proposition 3 tells us that these two difficulties can be overcome when $\mathcal{G} = \mathcal{G}_p$ and that the non-convex penalty $\varphi_{\mathcal{G}_p}$ can be computed in polynomial time by solving the convex optimization problem defined in Equation (9). The key component to obtain the flow definition of $\varphi_{\mathcal{G}_p}$ is the decomposability property of the weights $\eta_g$ defined in (8). This allows us to identify the cost of sending one unit of flow in $G'$ from $s$ to $t$ along a path $g$ to the cost of selecting the path $g$ in the context of the path coding penalty $\varphi_{\mathcal{G}_p}$. We now show that the same methodology applies to the convex penalty $\psi_{\mathcal{G}_p}$.

**Proposition 4 (Computing $\psi_{\mathcal{G}_p}$.)**
*Let* $\mathbf{w}$ *be in* $\mathbb{R}^p$*. Consider the network* $G'$ *defined in Section 3.1 with costs* $[c_{uv}]_{(u,v)\in E'}$*, and define* $\eta_g$ *as in (8). Then,*

$$\psi_{\mathcal{G}_p}(\mathbf{w}) = \min_{f\in\mathcal{F}} \left\{ \sum_{(u,v)\in E'} f_{uv}c_{uv} \;\; s.t. \;\; s_j(f) \geq |\mathbf{w}_j|,\ \forall j \in \{1,\dots,p\} \right\}, \tag{10}$$

*where* $\mathcal{F}$ *is the set of flows on* $G'$*. This is a minimum cost flow problem with some lower-capacity constraints, which can be computed in strongly polynomial time.*

4. See the definition of "strongly polynomial time" in Section 2.1.

From the similarity between Equations (9) and (10), it is easy to see that $\psi_{\mathcal{G}_p}$ and $\varphi_{\mathcal{G}_p}$ are closely related, one being a convex relaxation of the other as explained in Section 3.1. We have shown here that $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ can be computed in polynomial time and will discuss in Section 3.4 practical algorithms to do it in practice. Before that, we address the problem of optimizing (1).

### 3.3 Using Proximal Gradient Methods with the Path Coding Penalties

To address the regularized problem (1), we use proximal gradient methods, which we have presented in Section 2.2. We need for that to compute the proximal operators of $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ from Definition 1. We show that this operator can be efficiently computed by using network flow optimization.

**Proposition 5 (Computing the Proximal Operator of $\varphi_{\mathcal{G}_p}$.)**
*Let $\mathbf{u}$ be in $\mathbb{R}^p$. Consider the network $G'$ defined in Section 3.1 with costs $[c_{uv}]_{(u,v)\in E'}$, and define $\eta_g$ as in (8). Let us define*

$$f^\star \in \arg\min_{f\in\mathcal{F}} \left\{ \sum_{(u,v)\in E'} f_{uv} c_{uv} + \sum_{j=1}^{p} \frac{1}{2} \max\left(\mathbf{u}_j^2 (1 - s_j(f)), 0\right) \right\}, \tag{11}$$

*where $\mathcal{F}$ is the set of flows on $G'$. This is a minimum cost flow problem, with piecewise linear costs, which can be computed in strongly polynomial time. Denoting by $\mathbf{w}^\star \triangleq Prox_{\varphi_{\mathcal{G}p}}[\mathbf{u}]$, we have for all $j$ in $V = \{1,\dots,p\}$ that $\mathbf{w}_j^\star = \mathbf{u}_j$ if $s_f(f^\star) > 0$ and $0$ otherwise.*

Note that even though the formulation (3) is non-convex when $\Omega$ is the function $\varphi_{\mathcal{G}_p}$, its global optimum can be found by solving the convex problem described in Equation (11). As before, the key component to establish the mapping to a network flow problem is the decomposability property of the weights $\eta_g$. More details are provided in the proofs of Appendix D. Note also that any minimum cost flow problem with convex piecewise linear costs can be equivalently recast as a classical minimum cost flow problem with linear costs (see Ahuja et al., 1993), and therefore the above problem can be solved in strongly polynomial time. We now present similar results for $\psi_{\mathcal{G}_p}$.

**Proposition 6 (Computing the Proximal Operator of $\psi_{\mathcal{G}_p}$.)**
*Let $\mathbf{u}$ be in $\mathbb{R}^p$. Consider the network $G'$ defined in Section 3.1 with costs $[c_{uv}]_{(u,v)\in E'}$, and define $\eta_g$ as in (8). Let us define*

$$f^\star \in \arg\min_{f\in\mathcal{F}} \left\{ \sum_{(u,v)\in E'} f_{uv} c_{uv} + \sum_{j=1}^{p} \frac{1}{2} \max\left(|\mathbf{u}_j| - s_j(f), 0\right)^2 \right\}, \tag{12}$$

*where $\mathcal{F}$ is the set of flows on $G'$. This is a minimum cost flow problem, with piecewise quadratic costs, which can be computed in polynomial time. Denoting by $\mathbf{w}^\star \triangleq Prox_{\psi_{\mathcal{G}p}}[\mathbf{u}]$, we have for all $j$ in $V = \{1,\dots,p\}$, $\mathbf{w}_j^\star = \operatorname{sign}(\mathbf{u}_j)\min(|\mathbf{u}_j|, s_j(f^\star))$.*

The proof of this proposition is presented in Appendix D. We remark that we are dealing in Proposition 6 with a minimum cost flow problem with quadratic costs, which is more difficult to solve than when the costs are linear. Such problems with quadratic costs can be solved in weakly (instead of strongly) polynomial time (see Hochbaum, 2007)—that is, a time polynomial in $|V|$, $|E|$ and $\log(\|\mathbf{u}\|_\infty/\varepsilon)$ to obtain an $\varepsilon$-accurate solution to problem (12), where $\varepsilon$ can possibly be set to the machine precision. We have therefore shown that the computations of $\varphi_{\mathcal{G}_p}$, $\psi_{\mathcal{G}_p}$, $\text{Prox}_{\varphi_{\mathcal{G}_p}}$ and $\text{Prox}_{\psi_{\mathcal{G}_p}}$ can be done in polynomial time. We now discuss practical algorithms, which have empirically shown to be efficient and scalable (Goldberg, 1997; Bertsekas, 1998).

### 3.4 Practical Algorithms for Solving the Network Flow Problems

The minimum cost flow problems involved in the computations of $\varphi_{\mathcal{G}_p}$, $\psi_{\mathcal{G}_p}$ and $\mathrm{Prox}_{\varphi_{\mathcal{G}_p}}$ can be solved in the worst-case with $O\big((|V|\log|V|)(|E|+|V|\log|V|)\big)$ operations (see Ahuja et al., 1993). However, this analysis corresponds to the worst-case possible and the empirical complexity of network flow solvers is often much better (Boykov et al., 2001). Instead of a strongly polynomial algorithm, we have chosen to implement the scaling push-relabel algorithm Goldberg (1997), also known as an $\varepsilon$-relaxation method (Bertsekas, 1998). This algorithm is indeed empirically efficient despite its weakly polynomial worst-case complexity. It requires transforming the capacities and costs of the minimum cost flow problems into integers with an appropriate scaling and rounding procedure, and denoting by $C$ the (integer) value of the maximum cost in the network its worst-case complexity is $O\big(|V|^2|E|\log(C|V|)\big)$. This algorithm is appealing because of its empirical efficiency when the right heuristics are used Goldberg (1997). We choose $C$ to be as large as possible (using 64 bits integers) not to lose numerical precision, even though choosing $C$ according to the desired statistical precision and the robustness of the proximal gradient algorithms would be more appropriate. It has indeed been shown recently by Schmidt et al. (2011) that proximal gradient methods for convex optimization are robust to inexact computations of the proximal operator, as long as the precision of these computations iteratively increases with an appropriate rate.

Computing the proximal operator $\mathrm{Prox}_{\psi_{\mathcal{G}_p}}[\mathbf{u}]$ requires dealing with piecewise quadratic costs, which are more complicated to deal with than linear costs. Fortunately, cost scaling or $\varepsilon$-relaxation techniques can be modified to handle any convex costs, while keeping a polynomial complexity (Bertsekas, 1998). Concisely describing $\varepsilon$-relaxation algorithms is difficult because their convergence properties do not come from classical convex optimization theory. We present here an interpretation of these methods, but we refer the reader to Chapter 9 of Bertsekas (1998) for more details and implementation issues. In a nutshell, consider a primal convex cost flow problem $\min_{f\in\mathcal{F}}\sum_{(u,v)\in E}C_{uv}(f_{uv})$, where the functions $C_{uv}$ are convex, and without capacity constraints. Using classical Lagrangian duality, it is possible to obtain the following dual formulation

$$\max_{\pi\in\mathbb{R}^p}\sum_{(u,v)\in E}q_{uv}(\pi_u-\pi_v),\quad\text{where}\quad q_{uv}(\pi_u-\pi_v)\triangleq\min_{f_{uv}\geq 0}\left[C_{uv}(f_{uv})-(\pi_u-\pi_v)f_{uv}\right].$$

This formulation is unconstrained and involves for each node $u$ in $V$ a dual variable $\pi_u$ in $\mathbb{R}$, which is called the *price* of node $u$. $\varepsilon$-relaxation techniques rely on this dual formulation, and can be interpreted as approximate dual coordinate ascent algorithms. They exploit the network structure to perform computationally cheap updates of the dual and primal variables, and can deal with the fact that the functions $q_{uv}$ are concave but not differentiable in general. Presenting how this is achieved exactly would be too long for this paper; we instead refer the reader to Chapter 9 of Bertsekas (1998). In the next section, we introduce algorithms to compute the dual norm of $\psi_{\mathcal{G}_p}$, which is an important quantity to obtain optimality guarantees for (1) with $\Omega=\psi_{\mathcal{G}_p}$, or for implementing active set methods that are adapted to very large-scale very sparse problems (see Bach et al., 2012).

### 3.5 Computing the Dual-Norm of $\psi_{\mathcal{G}_p}$

The dual norm $\psi^*_{\mathcal{G}_p}$ of the norm $\psi_{\mathcal{G}_p}$ is defined for any vector $\kappa$ in $\mathbb{R}^p$ as $\psi^*_{\mathcal{G}_p}(\kappa)\triangleq\max_{\psi_{\mathcal{G}_p}(\mathbf{w})\leq 1}\mathbf{w}^\top\kappa$ (see Boyd and Vandenberghe, 2004). We show in this section that $\psi^*_{\mathcal{G}_p}$ can be computed efficiently.

**Proposition 7 (Computing the Dual Norm $\psi^*_{\mathcal{G}_p}$.)**
*Let $\kappa$ be in $\mathbb{R}^p$. Consider the network $G'$ defined in Section 3.1 with costs $[c_{uv}]_{(u,v)\in E'}$, and define $\eta_g$*

*as in (8). For $\tau \geq 0$, and all $j$ in $\{1,\ldots,p\}$, we define an additional cost for the vertex $j$ to be $-|\kappa_j|/\tau$. We then define for every path $g$ in $\mathcal{G}_p$, the length $l_\tau(g)$ to be the sum of the costs along the corresponding $(s,t)$-path from $G'$. Then,*

$$\psi^*_{\mathcal{G}_p}(\kappa) = \min_{\tau \in \mathbb{R}_+} \left\{ \tau \;\; s.t. \;\; \min_{g \in \mathcal{G}_p} l_\tau(g) \geq 0 \right\},$$

*and $\psi^*_{\mathcal{G}_p}(\kappa)$ is the smallest factor $\tau$ such that the shortest $(s,t)$-path on $G'$ has nonnegative length.*

The proof is given in Appendix D. We note that the above quantity $l_\tau(g)$ satisfies $l_\tau(g) = \eta_g - \|\kappa_g\|_1/\tau$, for every $\tau > 0$ and $\kappa$ in $\mathbb{R}^p$. We present a simple way for computing $\psi^*_{\mathcal{G}_p}$ in Algorithm 1, which is proven in Proposition 8 to be correct and to converge in polynomial time.

**Algorithm 1** Computation of the Dual Norm $\psi^*_{\mathcal{G}_p}$.

**input** $\kappa \in \mathbb{R}^p$ such that $\kappa \neq 0$.

Choose any path $g \in \mathcal{G}_p$ such that $\kappa_g \neq 0$;
$\delta \leftarrow -\infty$;
**while** $\delta < 0$ **do**
  $\tau \leftarrow \frac{\|\kappa_g\|_1}{\eta_g}$;
  $g \leftarrow \arg\min_{h \in \mathcal{G}_p} l_\tau(h)$; (shortest path problem in a directed acyclic graph);
  $\delta \leftarrow l_\tau(g)$;
**end while**
**Return:** $\tau = \psi^*_{\mathcal{G}_p}(\kappa)$ (value of the dual norm).

**Proposition 8 (Correctness and Complexity of Algorithm 1.)**
*For $\kappa$ in $\mathbb{R}^p$, Algorithm 1 computes $\psi^*_{\mathcal{G}_p}(\kappa)$ in at most $O(p|E'|)$ operations.*

The proof is also presented in Appendix D. We note that this worst-case complexity bound might be loose. We have indeed observed in our experiments that the empirical complexity is close to be linear in $|E'|$. To concretely illustrate why computing the dual norm can be useful, we now give optimality conditions for problem (1) involving $\psi^*_{\mathcal{G}_p}$. The following lemma can immediately be derived from Bach et al. (2012, Proposition 1.2).

**Lemma 9 (Optimality Conditions for Problem (1) with $\Omega = \psi_{\mathcal{G}}$.)**
*A vector $\mathbf{w}$ be in $\mathbb{R}^p$ is optimal for problem (1) with $\Omega = \psi_{\mathcal{G}}$ if and only if*

$$\psi^*_{\mathcal{G}}(\nabla L(\mathbf{w})) \leq \lambda \quad and \quad -\nabla L(\mathbf{w})^\top \mathbf{w} = \lambda \psi_{\mathcal{G}}(\mathbf{w}).$$

The next section presents an active-set type of algorithm (see Bach et al., 2012, Chapter 6) building upon these optimality conditions and adapted to our penalty $\psi_{\mathcal{G}_p}$.

### 3.6 Active Set Methods for Solving Problem (1) when $\Omega = \psi_{\mathcal{G}_p}$

As experimentally shown later in Section 4, proximal gradient methods allows us to efficiently solve medium-large/scale problems ($p < 100\,000$). Solving larger scale problems can, however, be more difficult. Algorithm 2 is an active-set strategy that can overcome this issue when the

solution is very sparse. It consists of solving a sequence of smaller instances of Equation (1) on subgraphs $\tilde{G} = (\tilde{V}, \tilde{E})$, with $\tilde{V} \subseteq V$ and $\tilde{E} \subseteq E$, which we call *active graphs*. It is based on the computation of the dual-norm $\psi^*_{\mathcal{G}_p}$, which we have observed can empirically be obtained in a time linear or close to linear in $|E'|$. Given such a subgraph $\tilde{G}$, we denote by $\tilde{\mathcal{G}}_p$ the set of paths in $\tilde{G}$. The subproblems the active set strategy involve are the following:

$$\min_{\mathbf{w} \in \mathbb{R}^p} \left\{ L(\mathbf{w}) + \lambda \psi_{\tilde{\mathcal{G}}_p}(\mathbf{w}) \ \text{ s.t. } \ \mathbf{w}_j = 0 \ \text{ for all } j \notin \tilde{V} \right\}. \tag{13}$$

The key observations are that (i) when $\tilde{G}$ is small, subproblem (13) is easy to solve; (ii) after solving (13), one can check optimality conditions for problem (1) using Lemma 9, and update $\tilde{G}$ accordingly. Algorithm 2 presents the full algorithm, and the next proposition ensures that it is correct.

**Algorithm 2** Active-set Algorithm for Solving Equation (1) with $\Omega = \psi_{\mathcal{G}_p}$.

**Initialization** $\mathbf{w} \leftarrow 0$; $\tilde{G} \leftarrow (\emptyset, \emptyset)$ (active graph);
**loop**
  Update $\mathbf{w}$ by solving subproblem (13) (using the current value of $\mathbf{w}$ as a warm start);
  Compute $\tau \leftarrow \psi^*_{\mathcal{G}_p}(\nabla L(\mathbf{w}))$ (using Algorithm 1);
  **if** $\tau \leq \lambda$ **then**
    **exit** the loop;
  **else**
    $g \leftarrow \arg\min_{g \in \mathcal{G}_p} l_\tau(g)$ (shortest path problem in a directed acyclic graph);
    $\tilde{V} \leftarrow \tilde{V} \cup g$; $\tilde{E} \leftarrow \tilde{E} \cup \{(u,v) \in E : u \in g \ \text{ and } \ v \in g\}$ (update of the active graph);
  **end if**
**end loop**
**Return:** $\mathbf{w}^\star \leftarrow \mathbf{w}$, solution to Equation (1).

**Proposition 10 (Correctness of Algorithm 2.)**
*Algorithm 2 solves Equation (1) when* $\Omega = \psi_{\mathcal{G}_p}$.

The proof is presented in Appendix D. It mainly relies on Lemma 9, which requires computing the quantity $\psi^*_{\mathcal{G}_p}(\nabla L(\mathbf{w}))$. More precisely, it can be shown that when $\mathbf{w}$ is a solution to subproblem (13) for a subgraph $\tilde{G}$, whenever $\psi^*_{\mathcal{G}_p}(\nabla L(\mathbf{w})) \leq \lambda$, it is also a solution to the original large problem (1). Note that variants of Algorithm 2 can be considered: one can select more than a single path $g$ to update the subgraph $\tilde{G}$, or one can approximately solve the subproblems (13). In the latter case, the stopping criterion could be relaxed in practice. One could use the criterion $\tau \leq \lambda + \varepsilon$, where $\varepsilon$ is a small positive constant, or one could use a duality gap to stop the optimization when the solution is guaranteed to be optimal enough.

In the next section, we present various experiments, illustrating how the different penalties and algorithms behave in practice.

## 4. Experiments and Applications

We now present experiments on synthetic, genomic and image data. Our algorithms have been implemented in C++ with a Matlab interface, they have been made available as part of the open-source

software package SPAMS, originally accompanying Mairal et al. (2010).[5] We have implemented the proximal gradient algorithm FISTA (Beck and Teboulle, 2009) for convex regularization functions and ISTA for non-convex ones. When available, we use a relative duality gap as a stopping criterion and stop the optimization when the relative duality gap is smaller than $10^{-4}$. In our experiments, we often need to solve Equation (1) for several values of the parameter $\lambda$, typically chosen on a logarithmic grid. We proceed with a continuation strategy: first we solve the problem for the largest value of $\lambda$, whose solution can be shown to be 0 when $\lambda$ is large enough; then we decrease the value of $\lambda$, and use the previously obtained solution as initialization. This warm-restart strategy allows us to quickly follow a regularization path of the problem. For non-convex problems, it provides us with a good initialization for a given $\lambda$. The algorithm ISTA with the non-convex penalty $\varphi_{\mathcal{G}_p}$ is indeed only guaranteed to iteratively decrease the value of the objective function. As often the case with non-convex problems, the quality of the optimization is subject to a good initialization, and this strategy has proven to be important to obtain good results.

As far as the choice of the parameters is concerned, we have observed that all penalties we have considered in our experiments are very sensitive to the regularization parameter $\lambda$. Thus, we use in general a fine grid to choose $\lambda$ using cross-validation or a validation set. Some of the penalties involve an extra parameter, $\gamma$ in the case of $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$. This parameter offers some flexibility, for example it promotes the connectivity of the solution for $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$, but it also requires to be tuned correctly to prevent overfitting. In practice, we have found the choice of the second parameter always less critical than $\lambda$ to obtain a good prediction performance, and thus we use a coarse grid to choose this parameter. All other implementation details are provided in each experimental section.

### 4.1 Synthetic Experiments

In this first experiment, we study our penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ in a controlled setting. Since generating synthetic graphs reflecting similar properties as real-life networks is difficult, we have considered three "real" graphs of different sizes, which are part of the 10th DIMACS graph partitioning and graph clustering challenge:[6]

- the graph jazz was compiled by Gleiser and Danon (2003) and represents a community network of jazz musicians. It contains $p = 198$ vertices and $m = 2\,742$ edges;
- the graph email was compiled by Guimerà et al. (2003) and represents e-mail exchanges in a university. It contains $p = 1\,133$ vertices and $m = 5\,451$ edges;
- the graph PGP was compiled by Boguñá et al. (2004) and represents information interchange among users in a computer network. It contains $p = 10\,680$ vertices and $m = 24\,316$ edges.

We choose an arbitrary topological ordering for all of these graphs, orient the arcs according to this ordering, and obtain DAGs.[7] We generate structured sparse linear models with measurements corrupted by noise according to different scenarii, and compare the ability of different regularization functions to recover the noiseless model. More precisely, we consider a design matrix $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ with less observations than predictors ($n \triangleq \lfloor p/2 \rfloor$), and whose entries are i.i.d. samples from a normal distribution $\mathcal{N}(0,1)$. For simplicity, we preprocess each column of $\mathbf{X}$ by removing its

5. The source code is available here: `http://spams-devel.gforge.inria.fr/`.
6. The data sets are available here: `http://www.cc.gatech.edu/dimacs10/archive/clustering.shtml`.
7. A topological ordering $\preceq$ of vertices in a directed graph is such that if there is an arc from vertex $u$ to vertex $v$, then $u \prec v$. A directed graph is acyclic is and only if it possesses a topological ordering (see Ahuja et al., 1993).

mean component and normalize it to have unit $\ell_2$-norm. Then, we generate sparse vectors $\mathbf{w}_0$ with $k$ non-zero entries, according to different strategies described in the sequel. We synthesize an observation vector $\mathbf{y} = \mathbf{X}\mathbf{w}_0 + \varepsilon$ in $\mathbb{R}^n$, where the entries of $\varepsilon$ are i.i.d. draws from a normal distribution $\mathcal{N}(0, \sqrt{k/n}\sigma)$, with different noise levels:

- high SNR: we choose $\sigma = 0.2$; this yields a signal noise ratio (SNR) $\|\mathbf{X}\mathbf{w}_0\|_2^2/\|\varepsilon\|_2^2$ of about 26. We note that for $\sigma \leq 0.1$ almost all penalties almost perfectly recover the true pattern;
- medium SNR: for $\sigma = 0.4$, the SNR is about 6;
- low SNR: for $\sigma = 0.8$, the SNR is about 1.6.

Choosing a good criterion for comparing different penalties is difficult, and a conclusion drawn from an experiment is usually only valid for a given criterion. For example, we present later an image denoising experiment in Section 4.2, where non-convex penalties outperform convex ones according to one performance measure, while being the other way around for another one. In this experiment, we choose the relative mean square error $\|\mathbf{X}\hat{\mathbf{w}} - \mathbf{X}\mathbf{w}_0\|_2^2$ as a criterion, and use ordinary least square (OLS) to refit the models learned using the penalties. Whereas OLS does not change the results obtained with the non-convex penalties we consider, it changes significantly the ones obtained with the convex ones. In practice, OLS counterbalances the "shrinkage" effect of convex penalties, and empirically improves the results quality for low noise regimes (high SNR), but deteriorates it for high noise regimes (low SNR).

For simplicity, we also assume (in this experiment only) that an oracle gives us the optimal regularization parameter $\lambda$, and therefore the conclusions we draw from the experiment are only the existence or not of good solutions on the regularization path for every penalty. A more exhaustive comparison would require testing different combinations (with OLS, without OLS) and different criteria, and using internal cross-validation to select the regularization parameters. This would require a much heavier computational setting, which we have chosen not to implement in this experiment. After obtaining the matrix $\mathbf{X}$, we propose several strategies to generate "true" models $\mathbf{w}_0$:

- in the scenario flat we randomly select $k$ entries without exploiting the graph structure;
- the scenario graph consists of randomly selecting 5 entries, and iteratively selecting new vertices that are connected in $G$ to at least one previously selected vertex. This produces fairly connected sparsity patterns, but does not exploit arc directions;
- the scenario path is similar to graph, but we iteratively add new vertices following single paths in $G$. It exploits arc directions and produces sparsity patterns that can be covered by a small number of paths, which is the sort of patterns that our path-coding penalties encourage.

The number of non-zero entries in $\mathbf{w}_0$ is chosen to be $k \triangleq \lfloor 0.1p \rfloor$ for the different graphs, resulting in a fairly sparse vector. The values of the non-zero entries are randomly chosen in $\{-1, +1\}$. We consider the formulation (1) where $L$ is the square loss: $L(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2$ and $\Omega$ is one of the following penalties:

- the classical $\ell_0$- and $\ell_1$-penalties;
- the penalty $\psi_{\mathcal{G}}$ of Jacob et al. (2009) where the groups $\mathcal{G}$ are pairs of vertices linked by an arc;
- our path-coding penalties $\varphi_{\mathcal{G}_p}$ or $\psi_{\mathcal{G}_p}$ with the weights $\eta_g$ defined in (7).

- the penalty of Huang et al. (2011), and their strategy to encourage sparsity pattern with a small number of connected components. We use their implementation of the greedy algorithm StructOMP.[8] This algorithm uses a strategy dubbed "block-coding" to approximately deal with this penalty (see Huang et al., 2011), and has an additional parameter, which we also denote by $\mu$, to control the trade-off between sparsity and connectivity.

For every method except StructOMP, the regularization parameter $\lambda$ is chosen among the values $2^{i/4}$, where $i$ is an integer. We always start from a large value for $i$, and decrease its value by one, following the regularization path. For the penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$, the parameter $\gamma$ is simply chosen in $\{1/4, 1/2, 1, 2, 4\}$. Since the algorithm StructOMP is greedy and iteratively increases the model complexity, we record every solution obtained on the regularization path during one pass of the algorithm. Based on information-theoretic arguments, Huang et al. (2011) propose a default value for their parameter $\mu = 1$. Since changing this parameter value empirically improves the results quality, we try the values $\{1/4, 1/2, 1, 2, 4\}$ for a fair comparison with our penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$.

We report the results for the three graphs, three scenarii for generating $\mathbf{w}_0$, three noise levels and the five penalties in Figure 4. We report on these graphs the ratio between the prediction mean square error and the best achievable error if the sparsity pattern was given by an oracle. In other words, denoting by $\hat{\mathbf{w}}^{\text{oracle}}$ the OLS estimate if an oracle gives us the sparsity pattern, we report the value $\|\mathbf{X}\hat{\mathbf{w}} - \mathbf{X}\mathbf{w}_0\|_2^2 / \|\mathbf{X}\hat{\mathbf{w}}^{\text{oracle}} - \mathbf{X}\mathbf{w}_0\|_2^2$. The best achievable value for this criterion is therefore 1, which is represented by a dotted line on all graphs. We reproduce the experiment 20 times, randomizing every step, including the way the vector $\mathbf{w}_0$ is generated to obtain error bars representing one standard deviation.

We make pairwise comparisons and statistically assess our conclusions using error bars or, when needed, paired one-sided T-tests with a 1% significance level. The comparisons are the following:

- **convex vs non-convex ($\ell_0$ vs $\ell_1$ and $\varphi_{\mathcal{G}_p}$ vs $\psi_{\mathcal{G}_p}$)**: *For high SNR, non-convex penalties do significantly better than convex ones, whereas it is the other way around for low SNR.* The differences are highly significant for the graphs email and PGP. For medium SNR, conclusions are mixed, either the difference between a convex penalty and its non-convex counterpart are not significant or one is better than another.

- **unstructured vs path-coding ($\ell_0$ vs $\varphi_{\mathcal{G}_p}$ and $\ell_1$ vs $\psi_{\mathcal{G}_p}$)**: *In the structured scenarii graph and path, the structured penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ respectively do better than $\ell_0$ and $\ell_1$. In the unstructured flat scenario, $\ell_0$ and $\ell_1$ should be preferred.* More precisely, for the scenarii graph and path, $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ respectively outperform $\ell_0$ and $\ell_1$ with statistically significant differences, except (i) for high SNR, both $\varphi_{\mathcal{G}_p}$ and $\ell_0$ achieve perfect recovery; (ii) with the smallest graph jazz, the $p$-values obtained to compare $\psi_{\mathcal{G}_p}$ vs $\ell_1$ are slightly above our 1% significance level. In the flat scenario, $\ell_0$ and $\varphi_{\mathcal{G}_p}$ give similar results, whereas $\psi_{\mathcal{G}_p}$ performs slightly worse than $\ell_1$ in general even though they are very close.

- **Jacob et al. (2009) vs path-coding ($\psi_{\mathcal{G}}$ with pairs vs $\psi_{\mathcal{G}_p}$)**: *in the scenario path, $\psi_{\mathcal{G}_p}$ outperforms $\psi_{\mathcal{G}}$ (pairs). It is generally also the case in the scenario graph.* The differences are always significant in the low SNR regime and with the largest graph PGP.

- **Huang et al. (2011) vs path-coding (StructOMP vs $\varphi_{\mathcal{G}_p}, \psi_{\mathcal{G}_p}$)**: *For the scenario path, either $\varphi_{\mathcal{G}_p}$ (for high SNR) or $\psi_{\mathcal{G}_p}$ (for low SNR) outperform StructOMP. For the scenario graph,*

8. The source code is available here: `http://ranger.uta.edu/~huang/R_StructuredSparsity.htm`.

*the best results are shared between StructOMP and our penalties for high and medium SNR, and our penalties do better for low SNR*. More precisely in the scenario graph: (i) there is no significant difference for high SNR between $\varphi_{\mathcal{G}_p}$ and StructOMP; (ii) for medium SNR, StructOMP does slightly better with the graph PGP and similarly as $\varphi_{\mathcal{G}}$ for the two other graphs; (iii) for low SNR, our penalties do better than StructOMP with the two largest graphs email and PGP and similarly with the smallest graph jazz.

To conclude this experiment, we have shown that our penalties offer a competitive alternative to StructOMP and the penalty of Jacob et al. (2009), especially when the "true" sparsity pattern is exactly a union of a few paths in the graph. We have also identified different noise and size regimes, where a penalty should be preferred to another. Our experiment also shows that having both a non-convex and convex variant of a penalty can be interesting. In low SNR, convex penalties are indeed better behaved than non-convex ones, whereas it is the other way around when the SNR is high.

### 4.2 Image Denoising

State-of-the-art image restoration techniques are often exploiting a good model for small image patches (Elad and Aharon, 2006; Dabov et al., 2007; Mairal et al., 2009). We consider here the task of denoising natural images corrupted by white Gaussian noise, following an approach introduced by Elad and Aharon (2006). It consists of the following steps:

1. extract all overlapping patches $(\mathbf{y}^i)_{i=1,\ldots,m}$ from a noisy image;

2. compute a sparse approximation of every individual patch $\mathbf{y}^i$:

$$\min_{\mathbf{w}^i \in \mathbb{R}^p} \Big[\frac{1}{2}\|\mathbf{y}^i - \mathbf{X}\mathbf{w}^i\|_2^2 + \lambda\Omega(\mathbf{w}^i)\Big], \tag{14}$$

   where the matrix $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^p]$ in $\mathbb{R}^{n \times p}$ is a predefined "dictionary", $\lambda\Omega$ is a sparsity-inducing regularization and the term $\mathbf{X}\mathbf{w}^i$ is the clean estimate of the noisy patch $\mathbf{y}^i$;

3. since the patches overlap, each pixel admits several estimates. The last step consists of averaging the estimates of each pixel to reconstruct the full image.

Whereas Elad and Aharon (2006) learn an overcomplete basis set to obtain a "good" matrix $\mathbf{X}$ in the step 2 above, we choose a simpler approach and use a classical orthogonal discrete cosine transform (DCT) dictionary $\mathbf{X}$ (Ahmed et al., 1974). We present such a dictionary in Figure 5 for $8 \times 8$ image patches. As shown in the figure, DCT elements can be organized on a two-dimensional grid and ordered by horizontal and vertical frequencies. We use the DAG structure connecting neighbors on the grid, going from low to high frequencies. In this experiment, we address the following questions:

**(A)** *In terms of optimization, is our approach efficient for this experiment?* Because the number of problems to solve is large (several millions), the task is difficult.

**(B)** *Do we get better results by using the graph structure than with classical $\ell_0$- and $\ell_1$-penalties?*

**(C)** *How does the method compare with the state of the art?*

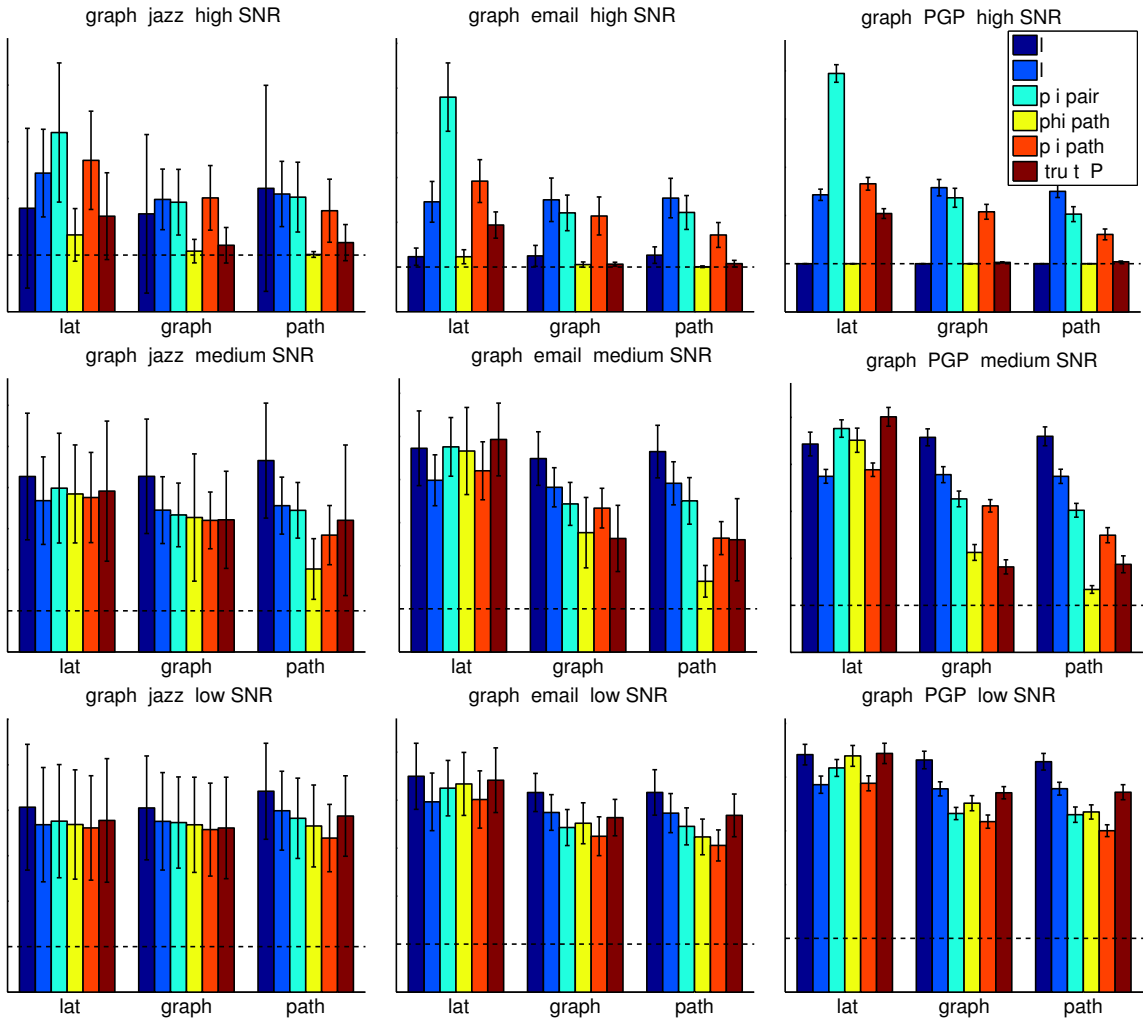



Figure 4: Every bar represents the ratio between the mean square error estimate and the oracle mean square error estimate (see main text for an explicit formula and the full experimental setting). The error bars represent one standard deviation. Each row corresponds to a specific noise level, and every column to a different graph. For a specific noise level and specific graph, the results for three scenarii, flat, graph and path are reported. Each group of six bars represents the results obtained with six penalties, from left to right: $\ell_0$, $\ell_1$, $\psi_{\mathcal{G}}$ (with $\mathcal{G}$ being the pairs of vertices linked by an arc), $\varphi_{\mathcal{G}_p}$, $\psi_{\mathcal{G}_p}$, and the method StructOMP. A legend is presented in the top right figure.

Note that since the dictionary $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ is orthogonal, the non-convex problems we address here are solved exactly. Indeed, it can be shown that Equation (14) is equivalent to

$$\min_{\mathbf{w}^i\in\mathbb{R}^p}\left[\frac{1}{2}\|\mathbf{X}^\top\mathbf{y}^i-\mathbf{w}^i\|_2^2+\lambda\Omega(\mathbf{w}^i)\right],$$

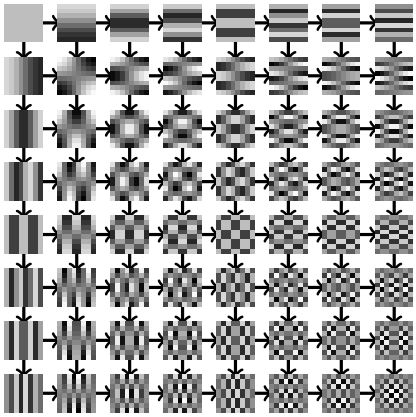


Figure 5: Orthogonal DCT dictionary with $n = 8 \times 8$ image patches. The dictionary elements are organized by horizontal and vertical frequencies.

and therefore the solution admits a closed form $\mathbf{w}^{i\star} \triangleq \mathrm{Prox}_{\lambda\Omega}(\mathbf{X}^\top \mathbf{y}^i)$. For $\ell_0$ and $\ell_1$, the solution is respectively obtained by hard and soft-thresholding, and we have introduced some tools in Section 3 to compute the proximal operators of $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$. We consider $e \times e$ image patches, with $e \in \{6, 8, 10, 12, 14, 16\}$, and a parameter $\lambda$ on a logarithmic scale with step $2^{1/8}$. We also exploit the variant of our penalties presented in Section 3 that allows us to choose the costs on the arcs of the graph $G'$. We choose here a small cost on the arc $(s, 1)$ of the graph $G'$, and a large one for every arc $(s, j)$, for $j$ in $\{2, \ldots, p\}$, such that all paths selected by our approach are encouraged to start by the variable 1 (equivalently the dictionary element $\mathbf{x}^1$ with the lowest frequencies). We use a data set of 12 classical high-quality images (uncompressed and free of artifact). We optimize the parameters $\lambda$ and $e$ on the first 3 images, keeping the 9 last images as a test set and report denoising results on Table 1. Even though this data set is relatively small, it is standard in the image processing literature, making the comparison easy with other approaches.[9]

We start by answering question **(A)**: we have observed that the time of computation depends on several factors, including the problem size and the sparsity of the solution (the sparser, the faster). In the setting $\sigma = 10$ and $e = 8$, we are able to denoise approximately 4 000 patches per second using $\varphi_{\mathcal{G}_p}$, and 1 800 in the setting $\sigma = 50$ and $e = 14$ with our laptop 1.2Ghz CPU (core i3 330UM). The penalty $\psi_{\mathcal{G}_p}$ requires solving quadratic minimum cost flow problems, and is slower to use in practice: the numbers 4 000 and 1 800 above respectively become 70 and 130. Our approach with $\varphi_{\mathcal{G}_p}$ proves therefore to be fairly efficient for our task, allowing us to process an image with about 250 000 patches in between one and three minutes. As expected, simple penalties are faster to use: about 65 000 patches per second can be processed using $\ell_0$.

Moving to question **(B)**, the best performance among the penalties $\ell_0$, $\ell_1$, $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ is obtained by $\varphi_{\mathcal{G}_p}$. This difference is statistically significant: we measure for instance an average improvement of $0.38 \pm 0.21$ dB over $\ell_0$ for $\sigma \geq 20$. For this denoising task, it is indeed typical to have non-convex penalties outperforming convex ones (see Mairal, 2010, Section 1.6.5, for a benchmark between $\ell_0$ and $\ell_1$-penalties), and this is why the original method of Elad and Aharon (2006) uses

9. This data set can be found for example in Mairal et al. (2009).

the $\ell_0$-penalty. Interestingly, this superiority of non-convex penalties in this denoising scheme based on overlapping patches is usually only observed after the averaging step 3. When one evaluates the quality of the denoising of individual patches without averaging—that is, after step 2, opposite conclusions are usually drawn (see again Mairal, 2010, Section 1.6.5). We therefore report mean-square error results for individual patches without averaging in Table 2 when $e = 10$. As expected, the penalty $\psi_{\mathcal{G}_p}$ turns out to be the best at this stage of the algorithm. Note that the bad results obtained by convex penalties after the averaging step are possibly due to the shrinkage effect of these penalties. It seems that the shrinkage is helpful for denoising individual patches, but hurts after the averaging process.

We also present the performance of state-of-the-art image denoising approaches in Table 1 to address question **(C)**. We have chosen to include in the comparison several methods that have successively been considered as the state of the art in the past: the Gaussian Scale Mixture (GSM) method of Portilla et al. (2003), the K-SVD algorithm of Elad and Aharon (2006), the BM3D method of Dabov et al. (2007) and the sparse coding approach of Mairal et al. (2009) (LSSC). We of course do not claim to do better than the most recent approaches of Dabov et al. (2007) or Mairal et al. (2009), which in addition to sparsity exploit non-local self similarities in images (Buades et al., 2005). Nevertheless, given the fact that we use a simple orthogonal DCT dictionary, unlike Elad and Aharon (2006) who learn overcomplete dictionaries adapted to the image, our approach based on the penalty $\varphi_{\mathcal{G}_p}$ performs relatively well. We indeed obtain similar results as Elad and Aharon (2006) and Portilla et al. (2003), and show that structured parsimony could be a promising tool in image processing.

| $\sigma$ | 5 | 10 | 15 | 20 | 25 | 50 | 100 |
|---|---|---|---|---|---|---|---|
| Our approach | | | | | | | |
| $\ell_0$ | **37.04** | 33.15 | 31.03 | 29.59 | 28.48 | 25.26 | 22.44 |
| $\ell_1$ | 36.42 | 32.28 | 30.06 | 28.59 | 27.51 | 24.48 | 21.96 |
| $\varphi_{\mathcal{G}_p}$ | 37.01 | **33.22** | **31.21** | **29.82** | **28.77** | **25.73** | **22.97** |
| $\psi_{\mathcal{G}_p}$ | 36.32 | 32.17 | 29.99 | 28.54 | 27.49 | 24.54 | 22.12 |
| State-of-the-art approaches | | | | | | | |
| Portilla et al., 2003 (GSM) | 36.96 | 33.19 | 31.17 | 29.78 | 28.74 | 25.67 | 22.96 |
| Elad and Aharon, 2006 (K-SVD) | 37.11 | 33.28 | 31.22 | 29.81 | 28.72 | 25.29 | 22.02 |
| Dabov et al., 2007 (BM3D) | 37.24 | 33.60 | 31.68 | **30.36** | **29.36** | 26.11 | 23.11 |
| Mairal et al., 2009 (LSSC) | **37.29** | **33.64** | **31.70** | **30.36** | 29.33 | **26.20** | **23.20** |

Table 1: Denoising results for 9 test images. The numbers represent the average PSNR in dB (higher is better). Denoting by MSE the mean-squared-error for images whose intensities are between 0 and 255, the PSNR is defined as $\text{PSNR} = 10\log_{10}(255^2/\text{MSE})$. Pixel values are scaled between 0 and 255 and $\sigma$ (the standard deviation of the noise) is between 5 and 100. The top part of the table presents the results of the denoising scheme obtained with different penalties. The bottom part presents the results obtained with various state-of-the-art denoising methods (see main text for more details). Best results are in bold for both parts of the table.

| $\sigma$ | 5 | 10 | 15 | 20 | 25 | 50 | 100 |
|---|---|---|---|---|---|---|---|
| $\ell_0$ | 3.60 | 10.00 | 16.65 | 23.22 | 29.58 | 57.97 | 95.79 |
| $\ell_1$ | 2.68 | 7.65 | 13.42 | 19.22 | 25.23 | 52.38 | 87.90 |
| $\varphi_{\mathcal{G}_p}$ | 3.26 | 8.36 | 13.62 | 18.83 | 23.99 | 47.66 | 84.74 |
| $\psi_{\mathcal{G}_p}$ | **2.66** | **7.27** | **12.29** | **17.35** | **22.65** | **45.04** | **76.85** |

Table 2: Denoising results for individual $10 \times 10$ image patches on the 9 test images. The numbers represent the mean-squared error for the image patches (lower the better). Best results are in bold.

### 4.3 Breast Cancer Data

One of our goal was to develop algorithmic tools improving the approach of Jacob et al. (2009). It is therefore natural to try one of the data set they used to obtain an empirical comparison. On the one hand, we have developed tools to enrich the group structure that the penalty $\psi_{\mathcal{G}}$ could handle, and thus we expect better results. On the other hand, the graph in this experiment is undirected and we need to use heuristics to transform it into a DAG.

We use in this task the breast cancer data set of Van De Vijver et al. (2002). It consists of gene expression data from 8 141 genes in $n=295$ breast cancer tumors and the goal is to classify metastatic samples versus non-metastatic ones. Following Jacob et al. (2009), we use the gene network compiled by Chuang et al. (2007), obtained by concatenating several known biological networks. As argued by Jacob et al. (2009), taking into account the graph structure into the regularization has two objectives: (i) possibly improving the prediction performance by using a better prior; (ii) identifying connected subgraphs of genes that might be involved in the metastatic form of the disease, leading to results that yield better interpretation than the selection of isolated genes. Even though prediction is our ultimate goal in this task, interpretation is equally important since it is necessary in practice to design drug targets. In their paper, Jacob et al. (2009) have succeeded in the sense that their penalty is able to extract more connected patterns than the $\ell_1$-regularization, even though they could not statistically assess significant improvements in terms of prediction. Following Jacob et al. (2009), we also assume that connectivity of the solution is an asset for interpretation. The questions we address here are the following:

**(A)** *Despite the heuristics described below to transform the graph into a DAG, does our approach lead to well-connected solutions in the original (undirected) graph? Do our penalties lead to better-connected solutions than other penalties?.*

**(B)** *Do our penalties lead to better classification performance than Jacob et al. (2009) and other classical unstructured and structured regularization functions? Is the graph structure useful to improve the prediction? Does sparsity help prediction?*

**(C)** *How efficient is our approach for this task?* The problem here is of medium/large scale but should be solved a large number of times (several thousands of times) because of the internal cross-validation procedure.

The graph of genes, which we denote by $G_0$, contains 42 587 edges, and, like Jacob et al. (2009), we keep the $p=7\,910$ genes that are present in $G_0$. In order to obtain interpretable results and select

connected components of $G_0$, Jacob et al. (2009) use their structured sparsity penalty $\psi_{\mathcal{G}}$ where the groups $\mathcal{G}$ are all pairs of genes linked by an arc. Our approach requires a DAG, but we will show in the sequel that we nevertheless obtain good results after heuristically transforming $G_0$ into a DAG. To do so, we first treat $G_0$ as directed by choosing random directions on the arcs, and second we remove some arcs along cycles in the graph. It results in a DAG containing 33 303 arcs, which we denote by $G$. This pre-processing step is of course questionable since our penalties are originally not designed to deal with the graph $G_0$. We of course do not claim to be able to individually interpret each path selected by our method, but, as we show, it does not prevent our penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ to achieve their ultimate goal—that is connectivity in the original graph $G_0$.

We consider the formulation (1) where $L$ is a weighted logistic regression loss:

$$L(\mathbf{w}) \triangleq \sum_{i=1}^{n} \frac{1}{n_{y_i}} \log(1 + e^{-y_i \mathbf{w}^\top \mathbf{x}_i}),$$

where the $y_i$'s are labels in $\{-1,+1\}$, the $\mathbf{x}_i$'s are gene expression vectors in $\mathbb{R}^p$. The weight $n_1$ is the number of positive samples, whereas $n_{-1}$ the number of negative ones. This model does not include an intercept, but the gene expressions are centered. The regularization functions included in the comparison are the following:

- our path-coding penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ with the weights $\eta_g$ defined in (7);
- the squared $\ell_2$-penalty (ridge logistic regression);
- the $\ell_1$-norm (sparse logistic regression);
- the elastic-net penalty of Zou and Hastie (2005), which has the form $\mathbf{w} \to \|\mathbf{w}\|_1 + (\mu/2)\|\mathbf{w}\|_2^2$, where $\mu$ is an additional parameter;
- the penalty $\psi_{\mathcal{G}}$ of Jacob et al. (2009) where the groups $\mathcal{G}$ are pairs of vertices linked by an arc;
- a variant of the penalty $\psi_{\mathcal{G}}$ of Jacob et al. (2009) whose form is given in Equation (16) of Appendix B, where the $\ell_2$-norm is used in place of the $\ell_\infty$-norm;
- the penalty $\zeta_{\mathcal{G}}$ of Jenatton et al. (2011) given in Appendix A where the groups are all pairs of vertices linked by an arc;
- the penalty $\zeta_{\mathcal{G}}$ of Jenatton et al. (2011) using the group structure adapted to DAGs described in Appendix A. This penalty has shown to be empirically problematic to use directly. The number of groups each variable belongs to significantly varies from a variable to another, resulting in overpenalization for some variables and underpenalization for some others. To cope with this issue, we have tried different strategies to choose the weights $\eta_g$ for every group in the penalty, similarly as those described by Jenatton et al. (2011), but we have been unable to obtain sparse solutions in this setting (typically the penalty selects here more than a thousand variables). A heuristic that has proven to be much better is to add a weighted $\ell_1$-penalty to $\zeta_{\mathcal{G}}$ to correct the over/under-penalization issue. Denoting for a variable $j$ in $\{1,\ldots,p\}$ by $d_j$ the number of groups the variable $j$ belongs to—in other words $d_j \triangleq \sum_{g \in \mathcal{G}: g \ni j} 1$, we add the term $\sum_{j=1}^{p} (\max_k d_k - d_j)|\mathbf{w}_j|$ to the penalty $\zeta_{\mathcal{G}}$.

The parameter $\lambda$ in Equation (1) is chosen on a logarithmic scale with steps $2^{1/4}$. The elastic-net parameter $\mu$ is chosen in $\{1, 10, 100\}$. The parameter $\gamma$ for the penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ is chosen in $\{2, 4, 8, 16\}$. We proceed by randomly sampling 20% of the data as a test set, keeping 80% for

training, selecting the parameters $\lambda, \mu, \gamma$ using internal 5-fold cross-validation on the training set, and we measure the average balanced error rate between the two classes on the test set. We have repeated this experiment 20 times and we report the averaged results in Table 3.

We start by answering question **(A)**. We remark that our penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$ succeed in selecting very few connected components of $G_0$, on average 1.3 for $\psi_{\mathcal{G}_p}$ and 1.6 for $\varphi_{\mathcal{G}_p}$ while providing sparse solutions. This significantly improves the connectivity of the solutions obtained using the approach of Jacob et al. (2009) or Jenatton et al. (2011). To claim interpretable results, one has of course to trust the original graph. Like Jacob et al. (2009), we assume that connectivity in $G_0$ is good a priori. We also study the effect of the preprocessing step we have used to obtain a directed acyclic graph $G$ from $G_0$. We report in the row "$\psi_{\mathcal{G}_p}$-random" in Table 3 the results obtained when randomizing the pre-processing step between every experimental run (providing us a different graph $G$ for every run). We observe that the outcome $G$ does not significantly change the sparsity and connectivity in $G_0$ of the sparsity patterns our penalty selects.

As far as the prediction performance is concerned, $\psi_{\mathcal{G}_p}$ seems to be the only penalty that is able to produce sparse and connected solutions while providing a similar average error rate as the $\ell_2$-penalty. The non-convex penalty $\varphi_{\mathcal{G}_p}$ produces a very sparse solution which is connected as well, but with an approximately 6% higher classification error rate. Note that because of the high variability of this performance measure, clearly assessing the statistical significance of the observed difference is difficult. As it was previously observed by Jacob et al. (2009), the data is very noisy and the number of samples is small, resulting in high variability. As Jacob et al. (2009), we have been unable to test the statistical significance rigorously—that is, without assuming independence of the different experimental runs. We can therefore not clearly answer the first part of question **(B)**. The second part of the question is however clearer: neither sparsity, nor the graph structure seems to help prediction in this experiment. We have for example tried to use the same graph $G$, but where we randomly permute the $p$ predictors (genes) at every run, making the graph structure irrelevant to the data. We report in Table 3 at the row "$\psi_{\mathcal{G}_p}$-permute" the average classification error rate, which is not significantly different than without permutation.

Our conclusions about the use of structured sparse estimation for this task are therefore mixed. On the one hand, none of the tested method are shown to perform statistically better in prediction than simple ridge regularization. On the other hand, our penalty $\psi_{\mathcal{G}_p}$ is the only one that performs as well as ridge while selecting a few predictive genes forming a a well-connected sparsity pattern. According to Jacob et al. (2009), this is a significant asset for biologists, assuming the original graph should be trusted.

Another aspect we would like to study is the stability properties of the selected sparsity patterns, which is often an issue with features selection methods (Meinshausen and Bühlmann, 2010). By introducing strong prior knowledge in the regularization, structured sparse estimation seems to provide more stable solutions than $\ell_1$. For instance, 5 genes are selected by $\ell_1$ in more than half of the experimental runs, whereas this number is 10 and 14 for the penalties of Jacob et al. (2009), and 33 for $\psi_{\mathcal{G}_p}$. Whereas we believe that stability is important, it is however hard to claim direct benefits of having a "stable" penalty without further study. By encouraging connectivity of the solution, variables that are highly connected in the graph tend to be more often selected, improving the stability of the solution, but not necessarily its interpretation in the absence of biological prior knowledge that prefers connectivity.

To answer question **(C)**, we study the computational efficiency of our implementation. One iteration of the proximal gradient method for the selected parameters is relatively fast, approximately

| | test error (%) | sparsity | connected components |
|---|---|---|---|
| $\ell_2^2$ | $31.0 \pm 6.1$ | 7910 | 58 |
| $\ell_1$ | $36.0 \pm 6.5$ | 32.6 | 30.9 |
| $\ell_2^2 + \ell_1$ | $31.5 \pm 6.7$ | 929.6 | 355.2 |
| Jacob et al. (2009)-$\ell_\infty$ | $35.9 \pm 6.8$ | 68.4 | 13.2 |
| Jacob et al. (2009)-$\ell_2$ | $36.0 \pm 7.2$ | 58.5 | 11.1 |
| Jenatton et al. (2011) (pairs) | $34.5 \pm 5.2$ | 33.4 | 28.8 |
| Jenatton et al. (2011) (DAG)+weighted $\ell_1$ | $35.6 \pm 7.0$ | 54.6 | 28.4 |
| $\varphi_{\mathcal{G}_p}$ | $36.0 \pm 6.8$ | 10.2 | 1.6 |
| $\psi_{\mathcal{G}_p}$ | $30.2 \pm 6.8$ | 69.9 | 1.3 |
| $\psi_{\mathcal{G}_p}$-permute | $33.2 \pm 7.6$ | 143.2 | 1.7 |
| $\psi_{\mathcal{G}_p}$-random | $31.6 \pm 6.0$ | 78.5 | 1.4 |

Table 3: Experimental results on the breast cancer data set. Column "test error": average balanced classification error rate on the test set in percents with standard deviations; the results are averaged over 20 runs and the parameters are selected for each run by internal 5-fold cross-validation. Column "sparsity": average number of selected genes. Column "connected components": average number of selected connected components in $G_0$.

0.17s for $\psi_{\mathcal{G}_p}$ and 0.15s for $\varphi_{\mathcal{G}_p}$ on a 1.2GHz laptop CPU (Intel core i3 330UM), but it tends to be significantly slower when the solution is less sparse, for instance with small values for $\lambda$. Since solving an instance of problem (1) requires computing about 500 proximal operators to obtain a reasonably precise solution, our method is fast enough to conduct this experiment in a reasonable amount of time. Of course, simpler penalties are faster to use. An iteration of the proximal gradient method takes about 0.15s for $\zeta_{\mathcal{G}}$, 0.05s for Jacob et al. (2009), 0.01s for $\ell_2$ and 0.003s for $\ell_1$.

## 5. Conclusion

Our paper proposes a new form of structured penalty for supervised learning problems where predicting features are sitting on a DAG, and where one wishes to automatically select a few connected subgraphs of the DAG. The computational feasibility of this form of penalty is established by making a new link between supervised path selection problems and network flows. Our penalties admit non-convex and convex variants, which can be used within the same network flow optimization framework. These penalties are flexible in the sense that they can control the connectivity of a problem solution, whether one wishes to encourage large or small connected components, and are able to model long-range interactions between variables.

Some of our conclusions show that being able to provide both non-convex and convex variants of the penalties is valuable. In various contexts, we have been able to find situations where convexity is helpful, and others where the non-convex approach leads to better solutions than the convex one. Our experiments show that when connectivity of a sparsity pattern is a good prior knowledge our approach is fast and effective for solving different prediction problems.

Interestingly, our penalties seem to perform empirically well on more general graphs than DAGs, when heuristically removing cycles, and we would like in the future to find a way to better

handle them. We also would like to make further connections with image segmentation techniques, which exploit different but related optimization techniques (see Boykov et al., 2001; Couprie et al., 2011), and kernel methods, where other types of feature selection in DAGs occur (Bach, 2008).

Finally, we are also interested in applying our techniques to sparse estimation problems where the sparsity pattern is expected to be exactly a combination of a few paths in a DAG. While the first version of this manuscript was under review, the first author started a collaboration with computational biologists to address the problem of isoform detection in RNA-Seq data. In a nutshell, a mixture of small fragments of mRNA is observed and the goal is to find a few mRNA molecules that explain the observed mixture. In mathematical terms, it corresponds to selecting a few paths in a directed acyclic graph in a penalized maximum likelihood formulation. Preliminary results obtained by Bernard et al. (2013) confirm that one could achieve state-of-the-art results for this task by adapting part of the methodology we have developed in this paper.

## Acknowledgments

This paper was supported in part by NSF grants SES-0835531, CCF-0939370, DMS-1107000, DMS-0907632, and by ARO-W911NF-11-1-0114. Julien Mairal would like to thank Laurent Jacob, Rodolphe Jenatton, Francis Bach, Guillaume Obozinski and Guillermo Sapiro for interesting discussions and suggestions leading to improvements of this manuscript, and his former research lab, the INRIA WILLOW and SIERRA project-teams, for letting him use computational resources funded by the European Research Council (VideoWorld and Sierra projects). He would also like to thank Junzhou Huang for providing the source code of his StructOMP software.

## Appendix A. The Penalty of Jenatton et al. (2011) for DAGs

The penalty of Jenatton et al. (2011) requires a pre-defined set of possibly overlapping groups $\mathcal{G}$ and is defined as follows:

$$\zeta_{\mathcal{G}}(\mathbf{w}) \triangleq \sum_{g \in \mathcal{G}} \eta_g \|\mathbf{w}_g\|_\nu, \tag{15}$$

where the vector $\mathbf{w}_g$ in $\mathbb{R}^{|g|}$ records the coefficients of $\mathbf{w}$ indexed by $g$ in $\mathcal{G}$, the scalars $\eta_g$ are positive weights, and $\nu$ typically equals 2 or $\infty$. This penalty can be interpreted as the $\ell_1$-norm of the vector $[\eta_g \|\mathbf{w}_g\|_\nu]_{g \in \mathcal{G}}$, therefore inducing sparsity at the group level. It extends the Group Lasso (Turlach et al., 2005; Yuan and Lin, 2006; Zhao et al., 2009) by allowing the groups to overlap.

Whereas the penalty $\psi_{\mathcal{G}}$ of Jacob et al. (2009) encourages solutions whose set of non-zero coefficients is a *union* of a few groups, the penalty $\zeta_{\mathcal{G}}$ promotes solutions whose sparsity pattern is in the *intersection* of some selected groups. This subtlety makes these two lines of work significantly different. It is for example unnatural to use the penalty $\zeta_{\mathcal{G}}$ to encourage connectivity in a graph. When the groups are defined as the pairs of vertices linked by an arc, it is indeed not clear that sparsity patterns defined as the intersection of such groups would lead to a well-connected subgraph. As shown experimentally in Section 4, this setting indeed performs poorly for this task.

However, when the graph is a DAG, there exists an appropriate group setting $\mathcal{G}$ *when the sparsity pattern of the solution is expected to be a single connected component of the DAG*. Let us indeed define the groups to be the sets of ancestors, and sets of descendents for every vertex; the set of descendents of a vertex $u$ in a DAG are defined as all vertices $v$ such that there exists a path from $u$

to $v$. Similarly the set of ancestors contains all vertices such that there is a path from $v$ to $u$. The corresponding penalty $\zeta_{\mathcal{G}}$ encourages sparsity patterns which are intersections of groups in $\mathcal{G}$, which can be shown to be exactly the connected subgraphs of the DAG.[10] This penalty is tractable since the number of groups is linear in the number of vertices, but as soon as the sparsity pattern of the solution is not connex (contains more than one connected component), it is unable to recover it, making it useful to seek for a more flexible approach. For this group structure $\mathcal{G}$, the penalty $\zeta_{\mathcal{G}}$ also suffers from other practical issues concerning the overpenalization of variables belonging to many different groups. These issues are empirically discussed in Section 4 on concrete examples.

Interestingly, Mairal et al. (2011) have shown that the penalty $\zeta_{\mathcal{G}}$ with $\nu = \infty$ and any arbitrary group structure $\mathcal{G}$ is related to network flows, but for different reasons than the penalties $\varphi_{\mathcal{G}_p}$ and $\psi_{\mathcal{G}_p}$. The penalty $\zeta_{\mathcal{G}}$ is indeed unrelated to the concept of graph sparsity since it does not require the features to have any graph structure. Solving regularized problems with $\zeta_{\mathcal{G}}$ is however challenging, and Mairal et al. (2011) have shown that the proximal operator of $\zeta_{\mathcal{G}}$ could be computed by means of a parametric maximum flow formulation. It involves a bipartite graph where the nodes represent variables and groups, and arcs represent inclusion relations between a variable and a group. Mairal et al. (2011) address thus a significantly different problem than ours, even though there is a common terminology between their work and ours.

## Appendix B. Links Between Huang et al. (2011) and Jacob et al. (2009)

Similarly as the penalty of $\varphi_{\mathcal{G}}$ of Huang et al. (2011), the penalty of Jacob et al. (2009) encourages the sparsity pattern of a solution to be the union of a small number of predefined groups $\mathcal{G}$. Unlike the function $\varphi_{\mathcal{G}}$, it is convex (it can be shown to be a norm), and is defined as follows:

$$\psi'_{\mathcal{G}}(\mathbf{w}) \triangleq \min_{(\xi^g \in \mathbb{R}^p)_{g\in\mathcal{G}}} \left\{ \sum_{g\in\mathcal{G}} \eta_g \|\xi^g\|_\nu \text{ s.t. } \mathbf{w} = \sum_{g\in\mathcal{G}} \xi^g \quad \text{and} \quad \forall\, g \in \mathcal{G},\ \text{Supp}(\xi^g) \subseteq g \right\}, \tag{16}$$

where $\|.\|_\nu$ typically denotes the $\ell_2$-norm ($\nu = 2$) or $\ell_\infty$-norm ($\nu = \infty$). In this equation, the vector $\mathbf{w}$ is decomposed into a sum of latent vectors $\xi^g$, one for every group $g$ in $\mathcal{G}$, with the constraint that the support of $\xi^g$ is itself in $g$. The objective function is a group Lasso penalty (Yuan and Lin, 2006; Turlach et al., 2005) as presented in Equation (15) which encourages the vectors $\xi^g$ to be zero. As a consequence, the support of $\mathbf{w}$ is contained in the union of a few groups $g$ corresponding to non-zero vectors $\xi^g$, which is exactly the desired regularization effect. We now give a proof of Lemma 2 relating this penalty to the convex relaxation of $\varphi_{\mathcal{G}}$ given in Equation (6) when $\nu = \infty$.

**Proof** We start by showing that $\psi'_{\mathcal{G}}$ is equal to the penalty $\psi_{\mathcal{G}}$ defined in Equation (6) on $\mathbb{R}^p_+$. We consider a vector $\mathbf{w}$ in $\mathbb{R}^p_+$ and introduce for all groups $g$ in $\mathcal{G}$ appropriate variables $\xi^g$ in $\mathbb{R}^p$. The linear program defining $\psi_{\mathcal{G}}$ can be equivalently rewritten

$$\psi_{\mathcal{G}}(\mathbf{w}) = \min_{\substack{\mathbf{x}\in\mathbb{R}_+^{|\mathcal{G}|} \\ (\xi^g\in\mathbb{R}^p)_{g\in\mathcal{G}}}} \left\{ \eta^\top \mathbf{x} \text{ s.t. } \sum_{g\in\mathcal{G}} \xi^g = \mathbf{w},\ \mathbf{N}\mathbf{x} \geq \sum_{g\in\mathcal{G}} \xi^g \text{ and } \forall\, g \in \mathcal{G},\ \text{Supp}(\xi^g) \subseteq g \right\},$$

where we use the assumption that for all vector $\mathbf{w}$ in $\mathbb{R}^p_+$, there exist vectors $\xi^g$ such that $\sum_{g\in\mathcal{G}} \xi^g = \mathbf{w}$. Let us consider an optimal pair $(\mathbf{x}, (\xi^g)_{g\in\mathcal{G}})$. For all indices $j$ in $\{1,\dots,p\}$, the constraint $\mathbf{N}\mathbf{x} \geq$

10. This setting was suggested to us by Francis Bach, Rodolphe Jenatton and Guillaume Obozinski in a private discussion. Note that we have assumed here for simplicity that the DAG is connected—that is, has a single connected component.

$\sum_{g\in\mathcal{G}}\xi^g$ leads to the following inequality

$$\underbrace{\sum_{g\ni j:x_g\geq\xi_j^g} x_g-\xi_j^g}_{\tau_j^+\geq 0}+\underbrace{\sum_{g\ni j:x_g<\xi_j^g} x_g-\xi_j^g}_{\tau_j^-\leq 0}\geq 0,$$

where $x_g$ denotes the entry of $\mathbf{x}$ corresponding to the group $g$, and two new quantities $\tau_j^+$ and $\tau_j^-$ are defined. For all $g$ in $\mathcal{G}$, we define a new vector $\xi'^g$ such that for every pair $(g,j)$ in $\mathcal{G}\times\{1,\ldots,p\}$:

1. if $j\notin g$, $\xi_j'^g\triangleq 0$;
2. if $j\in g$ and $x_g\geq\xi_j^g$, then $\xi_j'^g\triangleq x_g$;
3. if $j\in g$ and $x_g<\xi_j^g$, then $\xi_j'^g\triangleq\xi_j^g-(x_g-\xi_j^g)\frac{\tau_j^+}{\tau_j^-}$.

Note that if there exists $j$ and $g$ such that $x_g<\xi_j^g$, then $\tau_j^-$ is nonzero and the quantity $\tau_j^+/\tau_j^-$ is well defined. Simple verifications show that for all indices $j$ in $\{1,\ldots,p\}$, we have $\sum_{g\ni j}x_g-\xi_j'^g=\tau_j^++\tau_j^-=\sum_{g\ni j}x_g-\xi_j^g$, and therefore $\sum_{g\in\mathcal{G}}\xi'^g=\sum_{g\in\mathcal{G}}\xi^g=\mathbf{w}$. The pair $(\mathbf{x},(\xi'^g)_{g\in\mathcal{G}})$ is therefore also optimal. In addition, for all groups $g$ in $\mathcal{G}$ and index $j$ in $\{1,\ldots,p\}$, it is easy to show that $x_g-\xi_j'^g\geq 0$ and that we have at optimality $\operatorname{sign}(\xi_j^g)=\operatorname{sign}(\mathbf{w}_j)=1$ for any nonzero $\xi_j^g$. Therefore, the condition $\|\xi'^g\|_\infty\leq x_g$ is satisfied, which is stronger than the original constraint $\mathbf{N}\mathbf{x}\geq\sum_{g\in\mathcal{G}}\xi'^g$. Moreover, it is easy to show that $\|\xi'^g\|_\infty$ is necessary equal to $x_g$ at optimality (otherwise, one could decrease the value of $x_g$ to decrease the value of the objective function). We can now rewrite $\psi_\mathcal{G}(\mathbf{w})$ as

$$\psi_\mathcal{G}(\mathbf{w})=\left\{\min_{(\xi^g\in\mathbb{R}^p)_{g\in\mathcal{G}}}\sum_{g\in\mathcal{G}}\eta_g\|\xi^g\|_\infty\ \text{ s.t. }\ \sum_{g\in\mathcal{G}}\xi^g=\mathbf{w},\ \text{and}\ \forall\, g\in\mathcal{G},\ \operatorname{Supp}(\xi^g)\subseteq g\right\},$$

and we have shown that $\psi'_\mathcal{G}=\psi_\mathcal{G}$ on $\mathbb{R}_+^p$. By noticing that in Equation (6) a solution $(\xi^g)_{g\in\mathcal{G}}$ necessary satisfies $\operatorname{sign}(\xi_j^g)=\operatorname{sign}(\mathbf{w}_j)$ for every group $g$ and index $j$ such that $\xi_j^g\neq 0$, we can extend the proof from $\mathbb{R}_+^p$ to $\mathbb{R}^p$. ■

## Appendix C. Interpretation of the Weights $\eta_g$ with Coding Lengths

Huang et al. (2011) have given an interpretation of the penalty $\varphi_\mathcal{G}$ defined in Equation (4) in terms of coding length. We use similar arguments to interpret the path-coding penalty $\varphi_{\mathcal{G}_p}$ from an information-theoretic point of view. For appropriate weights $\eta_g$, the quantity $\varphi_{\mathcal{G}_p}(\mathbf{w})$ for a vector $\mathbf{w}$ in $\mathbb{R}^p$ can be seen as a coding length for the sparsity pattern of $\mathbf{w}$—that is, the following Kraft-MacMillan inequality (see Cover and Thomas, 2006; MacKay, 2003) is satisfied:

$$\sum_{S\in\{0,1\}^p}2^{-\varphi_{\mathcal{G}_p}(S)}\leq 1.$$

It is indeed well known in the information theory literature that there exists a binary uniquely decodeable code over $\{0,1\}^p$ with code length $\varphi_{\mathcal{G}_p}(S)$ for every pattern $S$ in $\{0,1\}^p$ if and only if the

above inequality is satisfied (see Cover and Thomas, 2006). We now show that a particular choice for the weights $\eta_g$ leads to an interesting interpretation.

Let us consider the graph $G'$ with source and sink vertices $s$ and $t$ defined in Section 3. We assume that a probability matrix transition $\pi(u,v)$ for all $(u,v)$ in $E'$ is given. With such a matrix transition, it is easy to obtain a coding length for the set of paths $\mathcal{G}_p$:

**Lemma 11 (Coding Length for Paths.)**
*Let $cl_g$ for a path $g = (v_1,\dots,v_k)$ in $\mathcal{G}_p$ be defined as*

$$cl_g \triangleq -\log_2 \pi(s,v_1) - \Big(\sum_{i=1}^{k-1} \log_2 \pi(v_i,v_{i+1})\Big) - \log_2 \pi(v_k,t).$$

*Then $cl_g$ is a coding length on $\mathcal{G}_p$.*

**Proof** We observe that for every path $(v_1,\dots,v_k)$ in $\mathcal{G}_p$ corresponds a unique walk of length $|V'|$ of the form $(s,v_1,\dots,v_b,t,t,\dots,t)$, and vice versa. Denoting by $\pi^t(s,t)$ the probability that a Markov chain associated to the probability transition matrix $\pi$ starting at the vertex $u$ is at the vertex $v$ at time $t$, it is easy to show that

$$\sum_{g\in\mathcal{G}_p} 2^{-\mathrm{cl}_g} = \pi^{|V'|}(s,t) = 1,$$

and therefore $\mathrm{cl}_g$ is a coding length on $\mathcal{G}_p$. ■

the term $-\log_2 \pi(s,v_1)$ represents the number of bits used to indicate that a path $g$ starts with the vertex $v_1$, whereas the bits corresponding to the terms $-\log_2 \pi(v_i,v_{i+1})$ indicate that the vertex following $v_i$ is $v_{i+1}$. The bits corresponding to last term $-\log_2 \pi(v_k,t)$ indicate the end of the path. To define the weights $\eta_g$, we now define the following costs:

$$c_{uv} \triangleq \begin{cases} 1-\log_2 \pi(u,v) & \text{if } u = s \\ -\log_2 \pi(u,v) & \text{otherwise.} \end{cases}$$

The weight $\eta_g$ therefore satisfies $\eta_g = \sum_{(u,v)\in E'} c_{uv} = \mathrm{cl}_g + 1$, and as shown by Huang et al. (2011), this is a sufficient condition for $\varphi_{\mathcal{G}_p}(\mathbf{w})$ to be a coding length for $\{0,1\}^p$.

We have therefore shown that (i) the different terms composing the weights $\eta_g$ can be interpreted as the number of bits used to encode the paths in the graph; (ii) it is possible to use probability transition matrices (or random walks) on the graph to design the weights $\eta_g$.

## Appendix D. Proofs of the Propositions

In this section, we provide the proofs of our main results.

### D.1 Proofs of Propositions 3 and 4

**Proof** We start by proving Proposition 3. Let us consider the alternative definition of $\varphi_{\mathcal{G}_p}$ given in Equation (5). This is an optimization problem over all paths in $G$, or equivalently all $(s,t)$-paths in $G'$ (since these two sets are in bijection). We associate to a vector $\mathbf{x}$ in $\{0,1\}^p$ a flow $f$ on $G'$, obtained by sending one unit of flow on every $(s,t)$-path $g$ satisfying $x^g = 1$ ($x^g$ denotes the entry of $\mathbf{x}$ associated to the group/path $g$). Each of these $(s,t)$-path flow has a cost $\eta_g$ and the total cost of $f$ is exactly $\eta^\top \mathbf{x}$.

We also observe that within this network flow framework, the constraint $\mathbf{N}\mathbf{x} \geq \text{Supp}(\mathbf{w})$ in Equation (5) is equivalent to saying that for all $j$ in $\{1,\dots,p\}$ the amount of flow going through the vertex $j$ (denoted by $s_j(f)$) is greater than one if $\mathbf{w}_j \neq 0$. We have therefore shown that $\varphi_{\mathcal{G}_p}(\mathbf{w})$ is the minimum cost achievable by a flow $f$ such that the constraint $s_j(f) \geq 1$ is satisfied for all $j$ in $\text{Supp}(\mathbf{w})$ and such that $f$ can be decomposed into binary $(s,t)$-path flows.

To conclude the proof of Proposition 3, we show that there exists an optimal flow that admits a decomposition into binary $(s,t)$-path flows. We notice that all arc capacities in Equation (9) are integers. A classical result (Ahuja et al., 1993, Theorem 9.10) says that there exists an optimal integer minimum-cost flow (a flow whose values on arcs are integers). We denote by $f$ such a solution. Then, the flow decomposition theorem (Bertsekas, 1998, Proposition 1.1) tells us that $f$ can be decomposed into $(s,t)$-path flows, but it also says that if $f$ is integer, then $f$ can be decomposed into integer $(s,t)$-path flows. We conclude the proof by noticing that sending more than one unit of flow on a path is not optimal (one can reduce the cost by sending only one unit of flow, while keeping the capacity constraints satisfied), and therefore there exists in fact a decomposition of $f$ into binary $(s,t)$-path flows. The quantity presented in Equation (9) is therefore equal to $\varphi_{\mathcal{G}_p}(\mathbf{w})$.

The proof of Proposition 4 builds upon the definition of $\psi_{\mathcal{G}}$ given in Equation (6) and is similar to the one of Proposition 3. ■

### D.2 Proof of Proposition 5

**Proof** Using the definition of the proximal operator in Equation (3) and the definition of $\varphi_{\mathcal{G}}$ in Equation (5), there exists a pattern $S$ in $\{0,1\}^p$ such that the solution $\mathbf{w}^\star$ of the proximal problem satisfies for all $j$, $\mathbf{w}^\star_j = \mathbf{u}_j$ if $j$ is in $S$, and $\mathbf{w}^\star_j = 0$ otherwise. We therefore rewrite Equation (3) by using the result of Proposition 3

$$\min_{S\in\{0,1\}^p, f\in\mathcal{F}} \left\{ \frac{1}{2}\sum_{j\notin S} \mathbf{u}_j^2 + \sum_{(u,v)\in E'} f_{uv}c_{uv} \text{ s.t. } s_j(f) \geq 1, \forall j \in S \right\}.$$

When $S$ is fixed, the above expression is a minimum cost flow problem with integer capacity constraints. Thus, there exists an integer flow solution, and we can, without loss of generality, constrain $f$ to be integer, and replace the constraints $s_j(f) \geq 1$ by $s_j(f) > 0$. After this modification, for $f$ is fixed, minimizing with respect to $S$ gives us the following closed form: for all $j$ in $\{1,\dots,p\}$, $S_j = 1$ if $s_j(f) > 0$ and 0 otherwise. With this choice for $S$, we have in addition $\sum_{j\notin S} \mathbf{u}_j^2 = \sum_{j=1}^p \max\left(\mathbf{u}_j^2(1-s_j(f)),0\right)$, and denoting by $\mathcal{F}_{\text{int}}$ the set of integer flows, we can equivalently rewrite the optimization problem

$$\min_{f\in\mathcal{F}_{\text{int}}} \left\{ \sum_{(u,v)\in E'} f_{uv}c_{uv} + \sum_{j=1}^p \frac{1}{2}\max\left(\mathbf{u}_j^2(1-s_j(f)),0\right) \right\}.$$

It is easy to transform this minimum cost flow problem with piecewise linear costs to a classical minimum cost flow problem (see Bertsekas, 1998, Exercise 1.19) with integral constraints. Therefore, it is possible to remove the constraint $f \in \mathcal{F}_{\text{int}}$ and replace it by $f \in \mathcal{F}$ without changing the optimal value of the cost function, leading to the formulation proposed in Equation (11). ■

### D.3 Proof of Proposition 6

**Proof** Without loss of generality, let us suppose that $\mathbf{u}$ is in $\mathbb{R}_+^p$. Let us indeed denote by $\mathbf{w}^\star \triangleq \text{Prox}_{\psi_{\mathcal{G}_p}}[\mathbf{u}]$. It is indeed easy to see that the signs of the entries of $\mathbf{w}^\star$ are necessary the same as those of $\mathbf{u}$, and flipping the signs of some entries of $\mathbf{u}$ results in flipping the signs of the corresponding entries in $\mathbf{w}^\star$. According to Proposition 4, we can write the proximal problem as

$$\min_{\mathbf{w}\in\mathbb{R}_+^p, f\in\mathcal{F}} \left\{ \frac{1}{2}\sum_{j=1}^p (\mathbf{u}_j - \mathbf{w}_j)^2 + \sum_{(u,v)\in E'} f_{uv} c_{uv} \;\text{ s.t. }\; s_j(f) \geq \mathbf{w}_j, \forall j \in \{1,\ldots,p\} \right\}.$$

When $f$ is fixed, minimizing with respect to $\mathbf{w}$ yields for all $j$, $\mathbf{w}_j^\star = \min(\mathbf{u}_j, s_j(f^\star))$. Plugging this closed form in the above equation yields the desired formulation. ■

### D.4 Proof of Proposition 7

**Proof** We recall that according to Lemma 2 we have for all $\mathbf{w}$ in $\mathbb{R}_+^p$

$$\psi_{\mathcal{G}_p}(\mathbf{w}) = \min_{\mathbf{x}\in\mathbb{R}_+^{|\mathcal{G}_p|}} \left\{ \eta^\top \mathbf{x} \;\text{ s.t. }\; \mathbf{N}\mathbf{x} \geq \mathbf{w} \right\}.$$

This is a linear program, whose dual (see Nocedal and Wright, 2006) gives us another definition for $\psi_{\mathcal{G}_p}$ on $\mathbb{R}_+^p$. Since strong duality holds here, we have

$$\psi_{\mathcal{G}_p}(\mathbf{w}) = \max_{\kappa\in\mathbb{R}_+^p} \left\{ \mathbf{w}^\top \kappa \;\text{ s.t. }\; \mathbf{N}^\top \kappa \leq \eta \right\}.$$

It is easy to show that one can extend this definition on $\mathbb{R}^p$ such that we have

$$\psi_{\mathcal{G}_p}(\mathbf{w}) = \max_{\kappa\in\mathbb{R}^p} \left\{ \mathbf{w}^\top \kappa \;\text{ s.t. }\; \max_{g\in\mathcal{G}_p} \frac{\|\kappa_g\|_1}{\eta_g} \leq 1 \right\}, \tag{17}$$

where $\kappa_g$ denotes the vector of size $|g|$ containing the entries of $\kappa$ corresponding to the indices in the group $g$. Note that a similar formula appears in Jacob et al. (2009, Lemma 2), when the $\ell_2$-norm is used in place of the $\ell_\infty$. We now define for a vector $\kappa$ in $\mathbb{R}^p$,

$$\psi_{\mathcal{G}_p}^*(\kappa) \triangleq \max_{g\in\mathcal{G}_p} \frac{\|\kappa_g\|_1}{\eta_g}.$$

It is easy to see that it is a norm, and by Equation (17), this is in fact the dual norm of the norm $\psi_{\mathcal{G}_p}$. We can now rewrite it as

$$\begin{aligned}
\psi_{\mathcal{G}_p}^*(\kappa) &= \min_{\tau\in\mathbb{R}_+} \left\{ \tau \;\text{ s.t. }\; \max_{g\in\mathcal{G}_p} \frac{\|\kappa_g\|_1}{\eta_g} \leq \tau \right\}, \\
&= \min_{\tau\in\mathbb{R}_+} \left\{ \tau \;\text{ s.t. }\; \max_{g\in\mathcal{G}_p} \frac{\|\kappa_g\|_1}{\tau} - \eta_g \leq 0 \right\}, \\
&= \min_{\tau\in\mathbb{R}_+} \left\{ \tau \;\text{ s.t. }\; \min_{g\in\mathcal{G}_p} l_\tau(g) \geq 0 \right\},
\end{aligned}$$

where we have identified the groups in $\mathcal{G}_p$ to their corresponding $(s,t)$-paths in $G'$. ■

### D.5 Proof of Proposition 8

**Proof**

**Correctness:**

We start by showing that when the algorithm converges, it returns the correct solution. We remark that the choice of $\tau$ in the algorithm ensures that there always exists a group $h$ in $\mathcal{G}_p$ such that $l_\tau(h) = 0$ and therefore we always have $\delta \leq 0$. Thus, when the algorithm converges, $\delta$ is equal to zero. Moreover, the function $G : \tau \to \min_{h \in \mathcal{G}_p} l_\tau(h)$ is non-increasing with $\tau$ since the functions $\tau \to l_\tau(h)$ are themselves non-increasing for all $h$ in $\mathcal{G}_p$. It is also easy to show that there exists a unique $\tau$ such that $G(\tau) = 0$, which is the desired solution. We conclude by noticing that at convergence, we have $G(\tau) = \delta = 0$.

**Convergence and complexity:**

We now show that the algorithm converges and give a worst-case complexity. We denote by $\tau_k$, $g_k$ and $\delta_k$ the respective values of $\tau, g$ and $\delta$ at the iteration $k$ of the algorithm. The definition of $\tau_{k+1}$ implies that

$$l_{\tau_{k+1}}(g_k) = 0 = \underbrace{l_{\tau_k}(g_k)}_{\delta_k \leq 0} + \underbrace{\|\kappa_{g_k}\|_1 \Big(\frac{1}{\tau_k} - \frac{1}{\tau_{k+1}}\Big)}_{-\delta_k \geq 0}.$$

Moreover,

$$\delta_{k+1} = l_{\tau_{k+1}}(g_{k+1}) = l_{\tau_k}(g_{k+1}) + \|\kappa_{g_{k+1}}\|_1 \Big(\frac{1}{\tau_k} - \frac{1}{\tau_{k+1}}\Big).$$

Since $l_{\tau_k}(g_{k+1}) \geq \delta_k$ ($\delta_k$ is the length of the shortest path), we can show that

$$\delta_{k+1} \geq \delta_k \Big(1 - \frac{\|\kappa_{g_{k+1}}\|_1}{\|\kappa_{g_k}\|_1}\Big).$$

Since $\delta_{k+1} \leq 0$, we remark that necessarily $\|\kappa_{g_{k+1}}\|_1 \leq \|\kappa_{g_k}\|_1$, and we have two possibilities

1. either $\|\kappa_{g_{k+1}}\|_1 = \|\kappa_{g_k}\|_1$ and $\delta_{k+1} = 0$, meaning that the algorithm has converged.
2. either $\|\kappa_{g_{k+1}}\|_1 < \|\kappa_{g_k}\|_1$ and it is easy to show that is implies that $\eta_{g_{k+1}} < \eta_{g_k}$.

Since $\eta_h = \gamma + |h|$, we obtain that $\eta_{g_k}$ is strictly decreasing with $k$ before the convergence of the algorithm. Since it can have at most $p$ different values, the algorithm converges in at most $p$ iterations. Updating the path $g$ in the algorithm can be done by solving a shortest path problem in the graph $G'$, which can be done in $O(|E|)$ operations since the graph is acyclic (Ahuja et al., 1993), and the total worst-case complexity is $O(p|E|)$, which concludes the proof. ■

### D.6 Proof of Proposition 10

**Proof** We denote by $\kappa$ the quantity $\kappa \triangleq \nabla L(\mathbf{w})$, and respectively by $\tilde{\kappa}$ and $\tilde{\mathbf{w}}$ the vectors recording the entries of $\kappa$ and $\mathbf{w}$ that are in $\tilde{V}$.

**Convergence of the algorithm:**

Convergence of the algorithm is easy to show and consists of observing that $\tilde{G}$ is strictly increasing. After solving subproblem (13), we have from the optimality conditions of Lemma 9 that $\psi^*_{\tilde{\mathcal{G}}_p}(\tilde{\kappa}) \leq \lambda$. By definition of the dual-norm given in Proposition 7, and using the same notation, we have that

for all $g$ in $\tilde{\mathcal{G}}_p$, $l_\lambda(g) \geq 0$. We now denote by $\tau$ the quantity $\tau = \psi^*_{\mathcal{G}_p}(\kappa)$; if $\tau \leq \lambda$, the algorithm stops. If not, we have that for all $g$ in $\tilde{\mathcal{G}}_p$, $l_\tau(g) > 0$ (since $\tau > \lambda$ and $l_\lambda(g) \geq 0$ for all $g$ in $\tilde{\mathcal{G}}_p$). The step $g \leftarrow \arg\min_{g \in \mathcal{G}_p} l_\tau(g)$ then selects a group $g$ such that $l_\tau(g) = 0$ (which is easy to check given the definition of $\psi^*_{\mathcal{G}_p}$ in Proposition 7. Therefore, the selected path $g$ is not in $\tilde{G}$, and the size of $\tilde{G}$ strictly increases, which guarantees the convergence of the algorithm.

**Correctness:**

We want to show that when the algorithm stops, it returns the correct solution. First, if we have $\tilde{G} = G$, it is trivially correct. If it stops with $\tilde{G} \neq G$, we have that $\psi^*_{\mathcal{G}_p}(\kappa) \leq \lambda$, and according to Lemma 9, we only need to check that $-\kappa^\top \mathbf{w} = \lambda \psi_{\mathcal{G}_p}(\mathbf{w})$. We remark that we have $\lambda \psi_{\mathcal{G}_p}(\mathbf{w}) \leq \lambda \psi_{\tilde{\mathcal{G}}_p}(\mathbf{w}) = -\tilde{\kappa}^\top \tilde{\mathbf{w}} = -\kappa^\top \mathbf{w} \leq \psi^*_{\mathcal{G}_p}(\kappa) \psi_{\mathcal{G}_p}(\mathbf{w})$,where the first inequality is easy to show when observing that $\tilde{\mathcal{G}}_p \subseteq \mathcal{G}_p$, and the last inequality is the generalized Hölder inequality for a norm and its dual-norm. Since $\psi^*_{\mathcal{G}_p}(\kappa) \psi_{\mathcal{G}_p}(\mathbf{w}) \leq \lambda \psi_{\mathcal{G}_p}(\mathbf{w})$ we have in fact equality, and we conclude the proof. ■

## References


N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows*. Prentice Hall, 1993.

H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, volume 1, pages 267–281, 1973.

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundation and Trends in Machine Learning*, 4:1–106, 2012.

A. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760, 1998.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

E. Bernard, L. Jacob, J. Mairal, and J.-P. Vert. Efficient RNA isoform identification and quantification from RNA-Seq data with network flows. Technical Report hal-00803134, 2013.

D.P. Bertsekas. *Network Optimization: Continuous and Discrete Models*. Athena Scientific, 1998.

T. Blumensath and M.E. Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265–274, 2009.

M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas. Models of social networks based on social distance attachment. *Physical Review E*, 70(5):056122, 2004.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

A. Buades, B. Coll, and J.M. Morel. A review of image denoising algorithms, with a new one. *SIAM Multiscale Modelling and Simulation*, 4(2):490, 2005.

V. Cehver, M. Duarte, C. Hedge, and R. G. Baraniuk. Sparse signal recovery using Markov random fields. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

A. Chambolle and J. Darbon. On total variation minimization and surface evolution using parametric maximal flows. *International Journal of Computer Vision*, 84(3):288–307, 2009.

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.

X. Chen, Q. Lin, S. Kim, J. Pena, J.G. Carbonell, and E.P. Xing. Smoothing proximal gradient method for general structured sparse learning. In *Proceedings of the Twenty-Seven Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011.

H.Y. Chuang, E. Lee, Y.T. Liu, D. Lee, and T. Ideker. Network-based classification of breast cancer metastasis. *Molecular Systems Biology*, 3(140), 2007.

T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001.

C. Couprie, L. Grady, H. Talbot, and L. Najman. Combinatorial continuous maximum flow. *SIAM Journal on Imaging Sciences*, 4:905–930, 2011.

T.M. Cover and J.A Thomas. *Elements of Information Theory*. Wiley, 2006. 2nd edition.

K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11): 1413–1457, 2004.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 54(12):3736–3745, 2006.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

L.R. Ford and D.R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399–404, 1956.

P. Gleiser and L. Danon. Community structure in jazz. *Advances in Complex Systems*, 6(4):565–573, 2003.

A.V. Goldberg. An Efficient Implementation of a Scaling Minimum-Cost Flow Algorithm. *Journal of Algorithms*, 22(1):1–29, 1997.

A.V. Goldberg and R.E. Tarjan. A new approach to the maximum flow problem. In *Proceedings of the ACM Symposium on Theory of Computing*, 1986.

R. Guimerà, L. Danon, A. Díaz Guilera, F. Giralt, and A. Arenas. Self-similar community structure in a network of human interactions. *Physical Review E*, 68(6):065103, 2003.

D.S. Hochbaum. Complexity and algorithms for nonlinear optimization problems. *Annals of Operations Research*, 153(1):257–296, 2007.

H. Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4):984–1006, 2010.

J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. *Journal of Machine Learning Research*, 12:3371–3412, 2011.

D.R. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–37, 2004.

L. Jacob, G. Obozinski, and J.-P. Vert. Group Lasso with overlap and graph Lasso. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

R. Jenatton, J-Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. *Journal of Machine Learning Research*, 12:2777–2824, 2011.

D.J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

J. Mairal. *Sparse Coding for Machine Learning, Image Processing and Computer Vision*. PhD thesis, Ecole Normale Supérieure de Cachan, 2010. `http://tel.archives-ouvertes.fr/tel-00595312`.

J. Mairal. Optimization with first-order surrogate functions. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. *Journal of Machine Learning Research*, 12:2649–2689, 2011.

S. Mallat and Z. Zhang. Matching pursuit in a time-frequency dictionary. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

A. Nemirovsky and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley, 1983.

Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, CORE Discussion paper, 2007.

J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, 2006. 2nd edition.

G. Obozinski and F. Bach. Convex relaxations for combinatorial penalties. Technical Report arXiv:1205.1240v1, 2012.

M. R. Osborne, B. Presnell, and B. A. Turlach. On the Lasso and its dual. *Journal of Computational and Graphical Statistics*, 9(2):319–37, 2000.

J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, 2003.

F. Rapaport, A. Zinovyev, M. Dutreix, E. Barillot, and J.-P. Vert. Classification of microarray data using gene networks. *BMC Bioinformatics*, 8(1):35, 2007.

J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

M. Schmidt, N. Le Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

B.A. Turlach, W.N. Venables, and S.J. Wright. Simultaneous variable selection. *Technometrics*, 47 (3):349–363, 2005.

M.H. Van De Vijver et al. A gene-expression signature as a predictor of survival in breast cancer. *The New England Journal of Medicine*, 347(25):1999–2009, 2002.

S. Wright, R. Nowak, and M. 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*, 68:49–67, 2006.

P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37(6A):3468–3497, 2009.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67(2):301–320, 2005.

Journal of Machine Learning Research 14 (2013) 2487-2517 Submitted 4/12; Revised 3/13; Published 9/13

# Greedy Feature Selection for Subspace Clustering

**Eva L. Dyer** E.DYER@RICE.EDU
*Department of Electrical & Computer Engineering*
*Rice University*
*Houston, TX, 77005, USA*

**Aswin C. Sankaranarayanan** SASWIN@ECE.CMU.EDU
*Department of Electrical & Computer Engineering*
*Carnegie Mellon University*
*Pittsburgh, PA, 15213, USA*

**Richard G. Baraniuk** RICHB@RICE.EDU
*Department of Electrical & Computer Engineering*
*Rice University*
*Houston, TX, 77005, USA*

**Editor:** Tong Zhang

## Abstract

Unions of subspaces provide a powerful generalization of single subspace models for collections of high-dimensional data; however, learning multiple subspaces from data is challenging due to the fact that segmentation—the identification of points that live in the same subspace—and subspace estimation must be performed simultaneously. Recently, sparse recovery methods were shown to provide a provable and robust strategy for *exact feature selection* (EFS)—recovering subsets of points from the ensemble that live in the same subspace. In parallel with recent studies of EFS with $\ell_1$-minimization, in this paper, we develop sufficient conditions for EFS with a greedy method for sparse signal recovery known as orthogonal matching pursuit (OMP). Following our analysis, we provide an empirical study of feature selection strategies for signals living on unions of subspaces and characterize the gap between sparse recovery methods and nearest neighbor (NN)-based approaches. In particular, we demonstrate that sparse recovery methods provide significant advantages over NN methods and that the gap between the two approaches is particularly pronounced when the sampling of subspaces in the data set is sparse. Our results suggest that OMP may be employed to reliably recover exact feature sets in a number of regimes where NN approaches fail to reveal the subspace membership of points in the ensemble.

**Keywords:** subspace clustering, unions of subspaces, hybrid linear models, sparse approximation, structured sparsity, nearest neighbors, low-rank approximation

## 1. Introduction

With the emergence of novel sensing systems capable of acquiring data at scales ranging from the nano to the peta, modern sensor and imaging data are becoming increasingly high-dimensional and heterogeneous. To cope with this explosion of high-dimensional data, one must exploit the fact that low-dimensional geometric structure exists amongst collections of data.

Linear subspace models are one of the most widely used signal models for collections of high-dimensional data, with applications throughout signal processing, machine learning, and the com-

©2013 Eva L. Dyer, Aswin C. Sankaranarayanan, and Richard G. Baraniuk.

putational sciences. This is due in part to the simplicity of linear models but also due to the fact that principal components analysis (PCA) provides a closed-form and computationally efficient solution to the problem of finding an optimal low-rank approximation to a collection of data (an ensemble of signals in $\mathbb{R}^n$). More formally, if we stack a collection of $d$ vectors (points) in $\mathbb{R}^n$ into the columns of $Y \in \mathbb{R}^{n \times d}$, then PCA finds the best rank-$k$ estimate of $Y$ by solving

$$\text{(PCA)} \qquad \min_{X \in \mathbb{R}^{n \times d}} \quad \sum_{i=1}^{n} \sum_{j=1}^{d} (Y_{ij} - X_{ij})^2 \quad \text{subject to} \quad \text{rank}(X) \leq k, \tag{1}$$

where $X_{ij}$ is the $(i, j)$ entry of $X$.

### 1.1 Unions of Subspaces

In many cases, a linear subspace model is sufficient to characterize the intrinsic structure of an ensemble; however, in many emerging applications, a single subspace is not enough. Instead, ensembles can be modeled as living on a *union of subspaces* or a union of affine planes of mixed or equal dimension. Formally, we say that a set of $d$ signals $\mathcal{Y} = \{y_1, \ldots, y_d\}$, each of dimension $n$, lives on a union of $p$ subspaces if $\mathcal{Y} \subset \mathcal{U} = \cup_{i=1}^{p} \mathcal{S}_i$, where $\mathcal{S}_i$ is a subspace of $\mathbb{R}^n$.

Ensembles ranging from collections of images taken of objects under different illumination conditions (Basri and Jacobs, 2003; Ramamoorthi, 2002), motion trajectories of point-correspondences (Kanatani, 2001), to structured sparse and block-sparse signals (Lu and Do, 2008; Blumensath and Davies, 2009; Baraniuk et al., 2010) are all well-approximated by a union of low-dimensional subspaces or a union of affine hyperplanes. Union of subspace models have also found utility in the classification of signals collected from complex and adaptive systems at different instances in time, for example, electrical signals collected from the brain's motor cortex (Gowreesunker et al., 2011).

### 1.2 Exact Feature Selection

Unions of subspaces provide a natural extension to single subspace models, but providing an extension of PCA that leads to provable guarantees for learning multiple subspaces is challenging. This is due to the fact that *subspace clustering*—the identification of points that live in the same subspace—and subspace estimation must be performed simultaneously. However, if we can sift through the points in the ensemble and identify subsets of points that lie along or near the same subspace, then a *local subspace estimate*[1] formed from any such set is guaranteed to coincide with one of the true subspaces present in the ensemble (Vidal et al., 2005; Vidal, 2011). Thus, to guarantee that we obtain an accurate estimate of the subspaces present in a collection of data, we must select a sufficient number of subsets (feature sets) containing points that lie along the same subspace; when a feature set contains points from the same subspace, we say that *exact feature selection* (EFS) occurs.

A common heuristic used for feature selection is to simply select subsets of points that lie within an Euclidean neighborhood of one another (or a fixed number of nearest neighbors (NNs)). Methods that use sets of NNs to learn a union of subspaces include: local subspace affinity (LSA) (Yan and Pollefeys, 2006), spectral clustering based on locally linear approximations (Arias-Castro et al., 2011), spectral curvature clustering (Chen and Lerman, 2009), and local best-fit flats (Zhang et al.,

1. A local subspace estimate is a low-rank approximation formed from a subset of points in the ensemble, rather than from the entire collection of data.

2012). When the subspaces present in the ensemble are non-intersecting and densely sampled, NN-based approaches provide high rates of EFS and in turn, provide accurate local estimates of the subspaces present in the ensemble. However, such approaches quickly fail as the intersection between pairs of subspaces increases and as the number of points in each subspace decreases; in both of these cases, the Euclidean distance between points becomes a poor predictor of whether points belong to the same subspace.

### 1.3 Endogenous Sparse Recovery

Instead of computing local subspace estimates from sets of NNs, Elhamifar and Vidal (2009) propose a novel approach for feature selection based upon forming sparse representations of the data via $\ell_1$-minimization. The main intuition underlying their approach is that when a sparse representation of a point is formed with respect to the remaining points in the ensemble, the representation should only consist of other points that belong to the same subspace. Under certain assumptions on both the sampling and "distance between subspaces",[2] this approach to feature selection leads to provable guarantees that EFS will occur, even when the subspaces intersect (Elhamifar and Vidal, 2010; Soltanolkotabi and Candès, 2012).

We refer to this application of sparse recovery as *endogenous sparse recovery* due to the fact that representations are not formed from an external collection of primitives (such as a basis or dictionary) but are formed "from within" the data. Formally, for a set of $d$ signals $\mathcal{Y} = \{y_1, \ldots, y_d\}$, each of dimension $n$, the sparsest representation of the $i^{\text{th}}$ point $y_i \in \mathbb{R}^n$ is defined as

$$c_i^* \;=\; \arg\min_{c \in \mathbb{R}^d} \|c\|_0 \qquad \text{subject to} \qquad y_i = \sum_{j \neq i} c(j) y_j, \tag{2}$$

where $\|c\|_0$ counts the number of non-zeroes in its argument and $c(j) \in \mathbb{R}$ denotes the contribution of the $j^{\text{th}}$ point $y_j$ to the representation of $y_i$. Let $\Lambda^{(i)} = \text{supp}(c_i^*)$ denote the subset of points selected to represent the $i^{\text{th}}$ point and $c_i^*(j)$ denote the contribution of the $j^{\text{th}}$ point to the endogenous representation of $y_i$. By penalizing representations that require a large number of non-zero coefficients, the resulting representation will be sparse.

In general, finding the sparsest representation of a signal has combinatorial complexity; thus, sparse recovery methods such as basis pursuit (BP) (Chen et al., 1998) or low-complexity greedy methods (Davis et al., 1994) are employed to obtain approximate solutions to (2).

### 1.4 Contributions

In parallel with recent studies of feature selection with $\ell_1$-minimization (Elhamifar and Vidal, 2010; Soltanolkotabi and Candès, 2012; Elhamifar and Vidal, 2013), in this paper, we study feature selection with a greedy method for sparse signal recovery known as orthogonal matching pursuit (OMP). The main result of our analysis is a new geometric condition (Theorem 1) for EFS with OMP that highlights the tradeoff between the: *mutual coherence* or similarity between points living in different subspaces and the *covering radius* of the points within the same subspace. The covering radius can be interpreted as the radius of the largest ball that can be embedded within each subspace without touching a point in the ensemble; the vector that lies at the center of this open ball, or the vector in the subspace that attains the covering radius is referred to as a *deep hole*. Theorem 1 suggests that

2. The distance between a pair of subspaces is typically measured with respect to the principal angles between the subspaces or other related distances on the Grassmanian manifold.

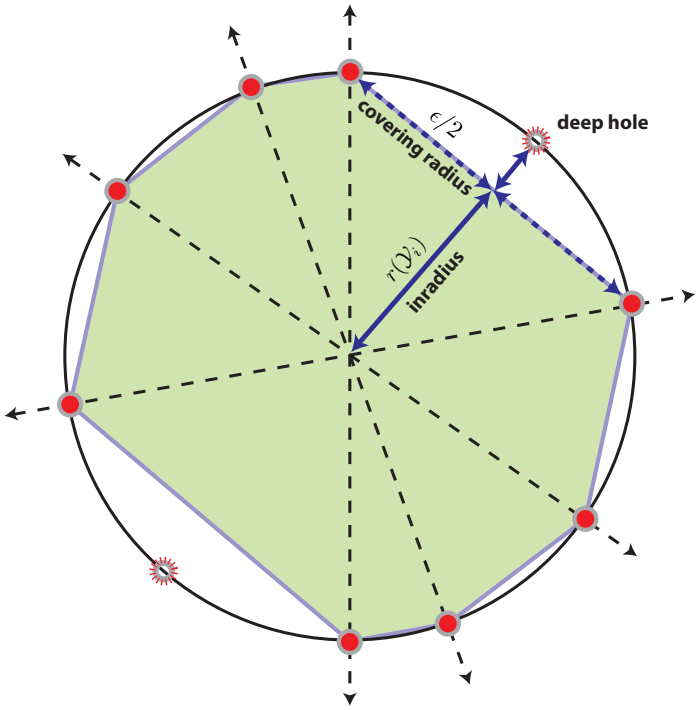



Figure 1: Covering radius of points in a normalized subspace. The interior of the antipodal convex hull of points in a normalized subspace—a subspace of $\mathbb{R}^n$ mapped to the unit $\ell_2$-sphere—is shaded. The vector in the normalized subspace (unit circle) that attains the covering radius (deep hole) is marked with a star: when compared with the convex hull, the deep hole coincides with the maximal gap between the convex hull and the set of all vectors that live in the normalized subspace.

subspaces can be arbitrarily close to one another and even intersect, as long as the data is distributed "nicely" along each subspace. By "nicely", we mean that the points that lie on each subspace do not cluster together, leaving large gaps in the sampling of the underlying subspace. In Figure 1, we illustrate the covering radius of a set of points on the sphere (the deep hole is denoted by a star).

After introducing a general geometric condition for EFS, we extend this analysis to the case where the data live on what we refer to as a *bounded union of subspaces* (Theorem 3). In particular, we show that when the points living in a particular subspace are incoherent with the principal vectors that support pairs of subspaces in the ensemble, EFS can be guaranteed, even when non-trivial intersections exist between subspaces in the ensemble. Our condition for bounded subspaces suggests that, in addition to properties related to the sampling of subspaces, one can characterize the separability of pairs of subspaces by examining the correlation between the data set and the unique set of principal vectors that support pairs of subspaces in the ensemble.

In addition to providing a theoretical analysis of EFS with OMP, the other main contribution of this work is revealing the gap between nearest neighbor-based (NN) approaches and sparse recovery methods, that is, OMP and BP, for feature selection. In both synthetic and real world experiments,

we observe that while OMP, BP, and NN provide comparable rates of EFS when subspaces in the ensemble are non-intersecting and densely sampled, sparse recovery methods provide significantly higher rates of EFS than NN sets when: (*i*) the dimension of the intersection between subspaces increases and (*ii*) the sampling density decreases (fewer points per subspace). In Section 5.4, we study the performance of OMP, BP, and NN-based subspace clustering on real data, where the goal is to cluster a collection of images into their respective illumination subspaces. We show that clustering the data with OMP-based feature selection (see Algorithm 2) provides improvements over NN and BP-based (Elhamifar and Vidal, 2010, 2013) clustering methods. In the case of very sparsely sampled subspaces, where the subspace dimension equals 5 and the number of points per subspace equals 16, we obtain a 10% and 30% improvement in classification accuracy with OMP (90%), when compared with BP (80%) and NN (60%).

### 1.5 Paper Organization

We now provide a roadmap for the rest of the paper.

*Section 2.* We introduce a signal model for unions of subspaces, detail the sparse subspace clustering (SSC) algorithm (Elhamifar and Vidal, 2010), and then go on to introduce the use of OMP for feature selection and subspace clustering (Algorithm 2); we end with a motivating example.

*Section 3 and 4.* We provide a formal definition of EFS and then develop the main theoretical results of this paper. We introduce sufficient conditions for EFS to occur with OMP for general unions of subspaces in Theorem 1, disjoint unions in Corollary 1, and bounded unions in Theorem 3.

*Section 5.* We conduct a number of numerical experiments to validate our theory and compare sparse recovery methods (OMP and BP) with NN-based feature selection. Experiments are provided for both synthetic and real data.

*Section 6.* We discuss the implications of our theoretical analysis and empirical results on sparse approximation, dictionary learning, and compressive sensing. We conclude with a number of interesting open questions and future lines of research.

*Section 7.* We supply the proofs of the results contained in Sections 3 and 4.

### 1.6 Notation

In this paper, we will work solely in real finite-dimensional vector spaces, $\mathbb{R}^n$. We write vectors $x$ in lowercase script, matrices $A$ in uppercase script, and scalar entries of vectors as $x(j)$. The standard $p$-norm is defined as

$$\|x\|_p = \left( \sum_{j=1}^{n} |x(j)|^p \right)^{1/p},$$

where $p \geq 1$. The "$\ell_0$-norm" of a vector $x$ is defined as the number of non-zero elements in $x$. The support of a vector $x$, often written as $\text{supp}(x)$, is the set containing the indices of its non-zero coefficients; hence, $\|x\|_0 = |\text{supp}(x)|$. We denote the Moore-Penrose pseudoinverse of a matrix $A$ as $A^\dagger$. If $A = U\Sigma V^T$ then $A^\dagger = V\Sigma^+ U^T$, where we obtain $\Sigma^+$ by taking the reciprocal of the entries in $\Sigma$, leaving the zeros in their places, and taking the transpose. An orthonormal basis (ONB) $\Phi$ that spans the subspace $\mathcal{S}$ of dimension $k$ satisfies the following two properties: $\Phi^T\Phi = I_k$ and $\text{range}(\Phi) = \mathcal{S}$, where $I_k$ is the $k \times k$ identity matrix. Let $P_\Lambda = X_\Lambda X_\Lambda^\dagger$ denote an ortho-projector onto the subspace spanned by the sub-matrix $X_\Lambda$.

## 2. Sparse Feature Selection for Subspace Clustering

In this section, we introduce a signal model for unions of subspaces, detail the sparse subspace clustering (SSC) method (Elhamifar and Vidal, 2009), and introduce an OMP-based method for sparse subspace clustering (SSC-OMP).

### 2.1 Signal Model

Given a set of $p$ subspaces of $\mathbb{R}^n$, $\{\mathcal{S}_1,\dots,\mathcal{S}_p\}$, we generate a "subspace cluster" by sampling $d_i$ points from the $i^{\text{th}}$ subspace $\mathcal{S}_i$ of dimension $k_i \leq k$. Let $\widetilde{\mathcal{Y}}_i$ denote the set of points in the $i^{\text{th}}$ subspace cluster and let $\widetilde{\mathcal{Y}} = \cup_{i=1}^{p} \widetilde{\mathcal{Y}}_i$ denote the union of $p$ subspace clusters. Each point in $\widetilde{\mathcal{Y}}$ is mapped to the unit sphere to generate a union of normalized subspace clusters. Let

$$\mathcal{Y} = \left\{ \frac{y_1}{\|y_1\|_2}, \frac{y_2}{\|y_2\|_2}, \cdots, \frac{y_d}{\|y_d\|_2} \right\}$$

denote the resulting set of unit norm points and let $\mathcal{Y}_i$ be the set of unit norm points that lie in the span of subspace $\mathcal{S}_i$. Let $\mathcal{Y}_{-i} = \mathcal{Y} \setminus \mathcal{Y}_i$ denote the set of points in $\mathcal{Y}$ with the points in $\mathcal{Y}_i$ excluded.

Let $Y = [Y_1\ Y_2\ \cdots\ Y_p]$ denote the matrix of normalized data, where each point in $\mathcal{Y}_i$ is stacked into the columns of $Y_i \in \mathbb{R}^{n \times d_i}$. The points in $Y_i$ can be expanded in terms of an ONB $\Phi_i \in \mathbb{R}^{n \times k_i}$ that spans $\mathcal{S}_i$ and subspace coefficients $A_i = \Phi_i^T Y_i$, where $Y_i = \Phi_i A_i$. Let $Y_{-i}$ denote the matrix containing the points in $Y$ with the sub-matrix $Y_i$ excluded.

### 2.2 Sparse Subspace Clustering

The sparse subspace clustering (SSC) algorithm (Elhamifar and Vidal, 2009) proceeds by solving the following basis pursuit (BP) problem for each point in $\mathcal{Y}$:

$$c_i^* \ = \ \arg\min_{c \in \mathbb{R}^d} \|c\|_1 \qquad \text{subject to} \qquad y_i = \sum_{j \neq i} c(j) y_j.$$

After solving BP for each point in the ensemble, each $d$-dimensional feature vector $c_i^*$ is placed into the $i^{\text{th}}$ row or column of a matrix $C \in \mathbb{R}^{d \times d}$ and spectral clustering (Shi and Malik, 2000; Ng et al., 2002) is performed on the graph Laplacian of the affinity matrix $W = |C| + |C^T|$.

In situations where points might not admit an exact representation with respect to other points in the ensemble, an inequality constrained version of BP known as basis pursuit denoising (BPDN) may be employed for feature selection (Elhamifar and Vidal, 2013). In this case, the following BPDN problem is computed for each point in $\mathcal{Y}$:

$$c_i^* \ = \ \arg\min_{c \in \mathbb{R}^d} \|c\|_1 \qquad \text{subject to} \qquad \|y_i - \sum_{j \neq i} c(j) y_j\|_2 < \kappa, \tag{3}$$

where $\kappa$ is a parameter that is selected based upon the amount of noise in the data. Recently, Wang and Xu (2013) provided an analysis of EFS for a unconstrained variant of the formulation in (3) for noisy unions of subspaces. Soltanolkotabi et al. (2013) proposed a robust procedure for subspace clustering from noisy data that extends the BPDN framework studied in Elhamifar and Vidal (2013) and Wang and Xu (2013).

**Algorithm 1** : **Orthogonal Matching Pursuit (OMP)**

**Input:** Input signal $y \in \mathbb{R}^n$, a matrix $A \in \mathbb{R}^{n \times d}$ containing $d$ signals $\{a_i\}_{i=1}^d$ in its columns, and a stopping criterion (either the sparsity $k$ or the approximation error $\kappa$).

**Output:** An index set $\Lambda$ containing the indices of all atoms selected in the pursuit and a coefficient vector $c$ containing the coefficients associated with of all atoms selected in the pursuit.

**Initialize:** Set the residual to the input signal $s = y$.

1. Select the atom that is maximally correlated with the residual and add it to $\Lambda$

$$\Lambda \leftarrow \Lambda \cup \arg\max_i |\langle a_i, s \rangle|.$$

2. Update the residual by projecting $s$ into the space orthogonal to the span of $A_\Lambda$

$$s \leftarrow (I - P_\Lambda) y.$$

3. Repeat steps (1)–(2) until the stopping criterion is reached, for example, either $|\Lambda| = k$ or the norm of the residual $\|s\|_2 \leq \kappa$.
4. Return the support set $\Lambda$ and coefficient vector $c = A_\Lambda^\dagger y$.

### 2.3 Greedy Feature Selection

Instead of solving the sparse recovery problem in (2) via $\ell_1$-minimization, we propose the use of a low-complexity method for sparse recovery known as orthogonal matching pursuit (OMP). We detail the OMP algorithm in Algorithm 1. For each point $y_i$, we solve Algorithm 1 to obtain a sparse representation of the signal with respect to the remaining points in $Y$. The output of the OMP algorithm is a *feature set*, $\Lambda^{(i)}$, which indexes the columns in $Y$ selected to form an endogenous representation of $y_i$.

After computing feature sets for each point in the data set via OMP, either a spectral clustering method or a consensus-based method (Dyer, 2011) may then be employed to cluster the data. In Algorithm 2, we outline a procedure for performing an OMP-based variant of the SSC algorithm that we will refer to as SSC-OMP.

### 2.4 Motivating Example: Clustering Illumination Subspaces

As a motivating example, we consider the problem of clustering a collection of images of faces captured under different illumination conditions: images of Lambertian objects (no specular reflections) captured under different illumination conditions have been shown to be well-approximated by a 5-dimensional subspace (Ramamoorthi, 2002). In Figure 2, we show an example of the affinity matrices obtained via OMP, BP, and NN, for a collection of images of two faces under 64 different illumination conditions selected from the Yale B Database (Georghiades et al., 2001). In this example, each $N \times N$ image is considered a point in $\mathbb{R}^{N^2}$ and the images of a particular person's face captured under different illumination conditions lie on a low-dimensional subspace. By varying the number of unique illumination conditions that we collect (number of points per subspace), we can manipulate the sampling density of the subspaces in the ensemble. We sort the images (points) such that images of the same face are contained in a contiguous block.

**Algorithm 2** : **Sparse Subspace Clustering with OMP (SSC-OMP)**

**Input:** A data set $Y \in \mathbb{R}^{n \times d}$ containing $d$ points $\{y_i\}_{i=1}^{d}$ in its columns, a stopping criterion for OMP, and the number of clusters $p$.

**Output:** A set of $d$ labels $\mathcal{L} = \{\ell(1), \ell(2), \ldots, \ell(d)\}$, where $\ell(i) \in \{1, 2, \ldots, p\}$ is the label associated with the $i^{\text{th}}$ point $y_i$.

**Step 1. Compute Subspace Affinity via OMP**

1. Solve Algorithm 1 for the $i^{\text{th}}$ point $y_i$ to obtain a feature set $\Lambda$ and coefficient vector $c$.
2. For all $j \in \Lambda^{(i)}$, let $C_{ij} = c(j)$. Otherwise, set $C_{ij} = 0$.
3. Repeat steps (1)–(2) for all $i = 1, \ldots, d$.

**Step 2. Perform Spectral Clustering**

1. Symmetrize the subspace affinity matrix $C$ to obtain $W = |C| + |C^T|$.
2. Perform spectral clustering on $W$ to obtain a set of $d$ labels $\mathcal{L}$.

To generate the OMP affinity matrices in the right column, we use the greedy feature selection procedure outlined in Step 1 of Algorithm 2, where the sparsity level $k = 5$. To generate the BP affinity matrices in the middle column, we solved the BPDN problem in (3) via a homotopy algorithm where we vary the noise parameter $\kappa$ and choose the smallest value of $\kappa$ that produces up to 5 coefficients. The resulting coefficient vectors are then stacked into the rows of a matrix $C$ and the final subspace affinity $W$ is computed by symmetrizing the coefficient matrix, $W = |C| + |C^T|$. To generate the NN affinity matrices in the left column, we compute the absolute normalized inner products between all points in the data set and then threshold each row to select the $k = 5$ nearest neighbors to each point.

## 3. Geometric Analysis of Exact Feature Selection

In this section, we provide a formal definition of EFS and develop sufficient conditions that guarantee that EFS will occur for all of the points contained within a particular subspace cluster.

### 3.1 Exact Feature Selection

In order to guarantee that OMP returns a feature set (subset of points from $\mathcal{Y}$) that produces an accurate local subspace estimate, we will be interested in determining when the feature set returned by Algorithm 1 only contains points that belong to the same subspace cluster, that is, *exact feature selection* (EFS) occurs. EFS provides a natural condition for studying performance of both subspace consensus and spectral clustering methods due to the fact that when EFS occurs, this results in a local subspace estimate that coincides with one of the true subspaces contained within the data. We now supply a formal definition of EFS.

**Definition 1 (Exact Feature Selection)** *Let $\mathcal{Y}_k = \{y : (I - P_k)y = 0,\ y \in \mathcal{Y}\}$ index the set of points in $\mathcal{Y}$ that live in the span of subspace $\mathcal{S}_k$, where $P_k$ is a projector onto the span of subspace $\mathcal{S}_k$. For a point $y \in \mathcal{Y}_k$ with feature set $\Lambda$, if $y_i \subseteq \mathcal{Y}_k$, $\forall i \in \Lambda$, we say that $\Lambda$ contains exact features.*

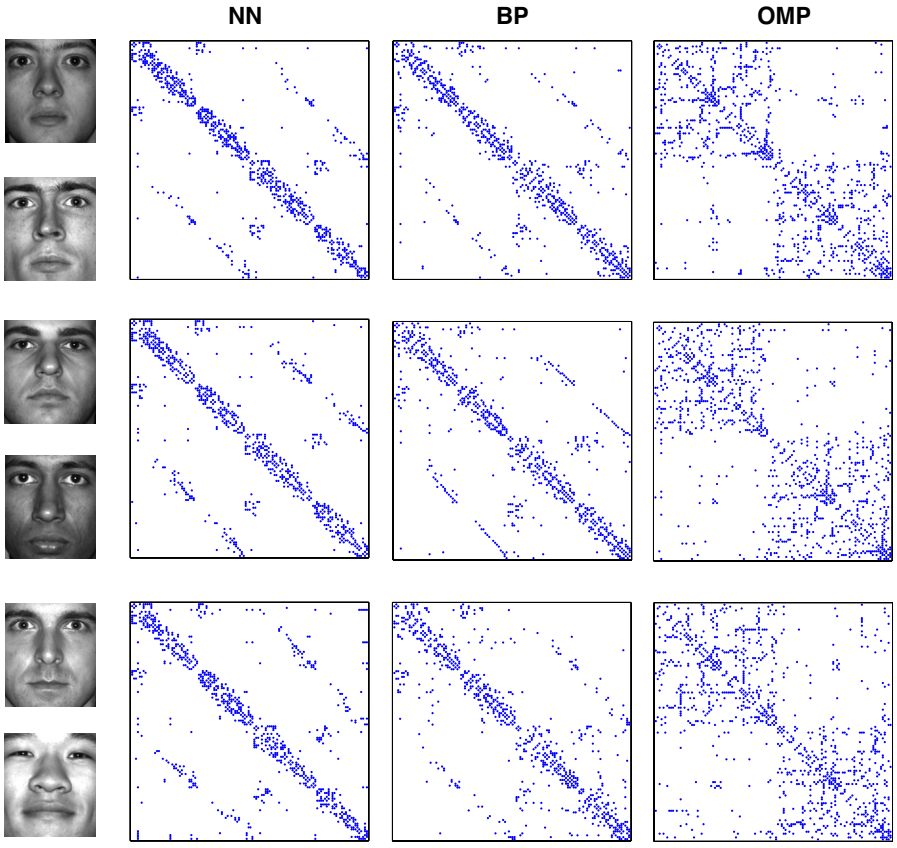



Figure 2: Comparison of subspace affinity matrices for illumination subspaces. In each row, we display the affinity matrices obtained for a different pair of illumination subspaces, for NN (left), BP (middle), and OMP (right). To the left of the affinity matrices, we display an exemplar image from each illumination subspace.

### 3.2 Sufficient Conditions for EFS

In this section, we develop geometric conditions that are sufficient for EFS with OMP. Before proceeding, however, we must introduce properties of the data set required to develop our main results.

#### 3.2.1 PRELIMINARIES

Our main geometric result in Theorem 1 below requires measures of both the distance between points in *different subspace clusters* and within the *same subspace cluster*. A natural measure of the similarity between points living in different subspaces is the *mutual coherence*. A formal definition of the mutual coherence is provided below in Def. 2.

**Definition 2 (Mutual Coherence)** *The mutual coherence between the points in the sets* $(\mathcal{Y}_i, \mathcal{Y}_j)$ *is defined as*

$$\mu_c(\mathcal{Y}_i, \mathcal{Y}_j) \;=\; \max_{u \in \mathcal{Y}_i, v \in \mathcal{Y}_j} |\langle u, v \rangle|, \text{ where } \|u\|_2 = \|v\|_2 = 1.$$

In words, the mutual coherence provides a point-wise measure of the normalized inner product (coherence) between all pairs of points that lie in two different subspace clusters.

Let $\mu_c(\mathcal{Y}_i)$ denote the maximum mutual coherence between the points in $\mathcal{Y}_i$ and all other subspace clusters in the ensemble, where

$$\mu_c(\mathcal{Y}_i) = \max_{j \neq i} \mu_c(\mathcal{Y}_i, \mathcal{Y}_j).$$

A related quantity that provides an upper bound on the mutual coherence is the cosine of the first *principal angle* between the subspaces. The first principal angle $\theta^*_{ij}$ between subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$, is the smallest angle between a pair of unit vectors $(u_1, v_1)$ drawn from $\mathcal{S}_i \times \mathcal{S}_j$. Formally, the first principal angle is defined as

$$\theta^*_{ij} = \min_{u \in \mathcal{S}_i,\ v \in \mathcal{S}_j} \arccos \langle u, v \rangle \quad \text{subject to} \quad \|u\|_2 = 1, \|v\|_2 = 1. \tag{4}$$

Whereas the mutual coherence provides a measure of the similarity between a pair of unit norm vectors that are contained in the sets $\mathcal{Y}_i$ and $\mathcal{Y}_j$, the cosine of the minimum principal angle provides a measure of the similarity between all pairs of unit norm vectors that lie in the span of $\mathcal{S}_i \times \mathcal{S}_j$. For this reason, the cosine of the first principal angle provides an upper bound on the mutual coherence. The following upper bound is in effect for each pair of subspace clusters in the ensemble:

$$\mu_c(\mathcal{Y}_i, \mathcal{Y}_j) \leq \cos(\theta^*_{ij}). \tag{5}$$

To measure how well points in the same subspace cluster cover the subspace they live on, we will study the covering radius of each normalized subspace cluster relative to the projective distance. Formally, the covering radius of the set $\mathcal{Y}_k$ is defined as

$$\text{cover}(\mathcal{Y}_k) \ = \ \max_{u \in \mathcal{S}_k} \min_{y \in \mathcal{Y}_k} \text{dist}(u, y),$$

where the projective distance between two vectors $u$ and $y$ is defined relative to the acute angle between the vectors

$$\text{dist}(u, y) = \sqrt{1 - \frac{|\langle u, y \rangle|^2}{\|u\|_2 \|y\|_2}}.$$

The covering radius of the normalized subspace cluster $\mathcal{Y}_i$ can be interpreted as the size of the largest open ball that can be placed in the set of all unit norm vectors that lie in the span of $\mathcal{S}_i$, without touching a point in $\mathcal{Y}_i$.

Let $(u^*_i, y^*_i)$ denote a pair of points that attain the maximum covering diameter for $\mathcal{Y}_i$; $u^*_i$ is referred to as a deep hole in $\mathcal{Y}_i$ along $\mathcal{S}_i$. The covering radius can be interpreted as the sine of the angle between the deep hole $u^*_i \in \mathcal{S}_i$ and its nearest neighbor $y^*_i \in \mathcal{Y}_i$. We show the geometry underlying the covering radius in Figure 1.

In the sequel, we will be interested in the maximum (worst-case) covering attained over all $d_i$ sets formed by removing a single point from $\mathcal{Y}_i$. We supply a formal definition below in Def. 3.

**Definition 3 (Covering Radius)** *The maximum covering diameter* $\varepsilon$ *of the set* $\mathcal{Y}_i$ *along the subspace* $\mathcal{S}_i$ *is defined as*

$$\varepsilon \ = \ \max_{j=1,\ldots,d_i} 2 \ \text{cover}(\{\mathcal{Y}_i \setminus y_j\}).$$

*Hence, the covering radius equals* $\varepsilon/2$.

A related quantity is the *inradius* of the set $\mathcal{Y}_i$, or the cosine of the angle between a point in $\mathcal{Y}_i$ and any point in $\mathcal{S}_i$ that attains the covering radius. The relationship between the covering diameter $\varepsilon$ and inradius $r(\mathcal{Y}_i)$ is given by

$$r(\mathcal{Y}_i) = \sqrt{1 - \frac{\varepsilon^2}{4}}. \tag{6}$$

A geometric interpretation of the inradius is that it measures the distance from the origin to the maximal gap in the antipodal convex hull of the points in $\mathcal{Y}_i$. The geometry underlying the covering radius and the inradius is displayed in Figure 1.

### 3.2.2 MAIN RESULT FOR EFS

We are now equipped to state our main result for EFS with OMP. The proof is contained in Section 7.1.

**Theorem 1** *Let $\varepsilon$ denote the maximal covering diameter of the subspace cluster $\mathcal{Y}_i$ as defined in Def. 3. A sufficient condition for Algorithm 1 to return exact feature sets for all points in $\mathcal{Y}_i$ is that the mutual coherence*

$$\mu_c(\mathcal{Y}_i) \;<\; \sqrt{1 - \frac{\varepsilon^2}{4}} - \frac{\varepsilon}{\sqrt[4]{12}} \max_{j \neq i} \cos(\theta^*_{ij}), \tag{7}$$

*where $\theta^*_{ij}$ is the minimum principal angle defined in (4).*

In words, this condition requires that the mutual coherence between points in *different subspaces* is less than the difference of two terms that both depend on the covering radius of points along a *single subspace*. The first term on the RHS of (7) is equal to the inradius, as defined in (6). The second term on the RHS of (7) is the product of the cosine of the minimum principal angle between pairs of subspaces in the ensemble and the covering diameter $\varepsilon$ of the points in $\mathcal{Y}_i$.

When subspaces in the ensemble intersect, that is, $\cos(\theta^*_{ij}) = 1$, condition (7) in Theorem 1 can be simplified as

$$\mu_c(\mathcal{Y}_i) < \sqrt{1 - \frac{\varepsilon^2}{4}} - \frac{\varepsilon}{\sqrt[4]{12}} \approx \sqrt{1 - \frac{\varepsilon^2}{4}} - \frac{\varepsilon}{1.86}.$$

In this case, EFS can be guaranteed for intersecting subspaces as long as the points in distinct subspace clusters are bounded away from intersections between subspaces. When the covering radius shrinks to zero, Theorem 1 requires that $\mu_c < 1$, or that points from different subspaces do not lie exactly in the subspace intersection, that is, are identifiable from one another.

### 3.2.3 EFS FOR DISJOINT SUBSPACES

When the subspaces in the ensemble are *disjoint*, that is, $\cos(\theta^*_{ij}) < 1$, Theorem 1 can be simplified further by using the bound for the mutual coherence in (5). This simplification results in the following corollary.

**Corollary 1** *Let $\theta^*_{ij}$ denote the first principal angle between a pair of disjoint subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$, and let $\varepsilon$ denote the maximal covering diameter of the points in $\mathcal{Y}_i$. A sufficient condition for Algorithm 1 to return exact feature sets for all points in $\mathcal{Y}_i$ is that*

$$\max_{j \neq i} \cos(\theta^*_{ij}) \;<\; \frac{\sqrt{1 - \varepsilon^2/4}}{1 + \varepsilon/\sqrt[4]{12}}.$$

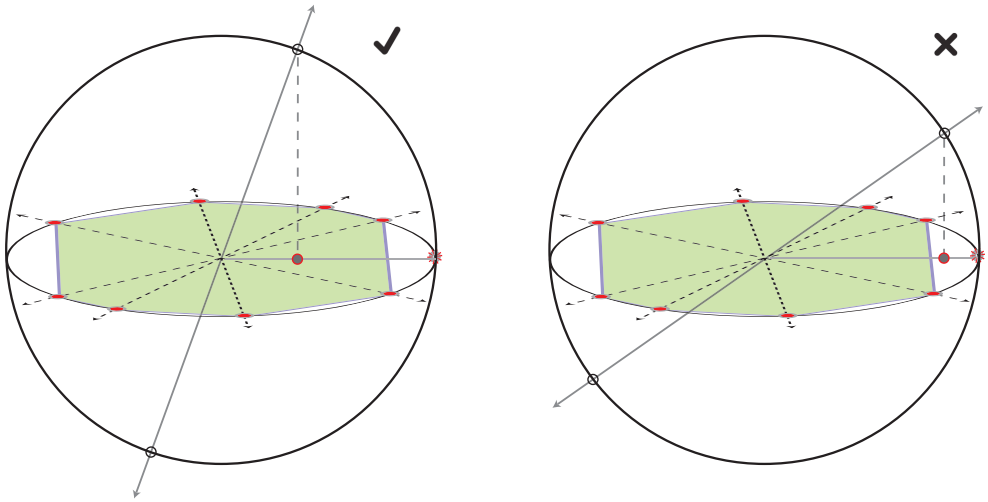


Figure 3: Geometry underlying EFS. A union of two disjoint subspaces of different dimension: the convex hull of a set of points (red circles) living on a 2D subspace is shaded (green). In (a), we show an example where EFS is guaranteed—the projection of points along the 1D subspace lie inside the shaded region. In (b), we show an example where EFS is not guaranteed—the projection of points along the 1D subspace lie outside the shaded region.

### 3.2.4 GEOMETRY UNDERLYING EFS WITH OMP

The main idea underlying the proof of Theorem 1 is that, at each iteration of Algorithm 1, we require that the residual used to select a point to be included in the feature set is closer to a point in the *correct subspace cluster* ($\mathcal{Y}_i$) than a point in an *incorrect subspace cluster* ($\mathcal{Y}_{-i}$). To be precise, we require that the normalized inner product of the residual signal $s$ and all points outside of the correct subspace cluster

$$\max_{y \in \mathcal{Y}_{-i}} \frac{|\langle s, y \rangle|}{\|s\|_2} < r(\mathcal{Y}_i), \tag{8}$$

at each iteration of Algorithm 1. To provide the result in Theorem 1, we require that (8) holds for all $s \in \mathcal{S}_i$, or all possible residual vectors.

A geometric interpretation of the EFS condition in Theorem 1 is that the orthogonal projection of all points outside of a subspace must lie within the antipodal convex hull of the set of normalized points that span the subspace. To see this, consider the projection of the points in $\mathcal{Y}_{-i}$ onto $\mathcal{S}_i$. Let $z_j^*$ denote the point on subspace $\mathcal{S}_i$ that is closest to the signal $y_j \in \mathcal{Y}_{-i}$,

$$z_j^* = \arg\min_{z \in \mathcal{S}_i} \|z - y_j\|_2.$$

We can also write this projection in terms of a orthogonal projection operator $P_i = \Phi_i \Phi_i^T$, where $\Phi_i$ is an ONB that spans $\mathcal{S}_i$ and $z_j^* = P_i y_j$.

By definition, the normalized inner product of the residual with points in incorrect subspace clusters is upper bounded as

$$\max_{y_j \in \mathcal{Y}_{-i}} \frac{|\langle s, y_j \rangle|}{\|s\|_2} \leq \max_{y_j \in \mathcal{Y}_{-i}} \frac{|\langle z_j^*, y_j \rangle|}{\|z_j^*\|_2} = \max_{y_j \in \mathcal{Y}_{-i}} \cos \angle \{z_j^*, y_j\}$$

Thus to guarantee EFS, we require that the cosine of the angle between all signals in $\mathcal{Y}_{-i}$ and their projection onto $\mathcal{S}_i$ is less than the inradius of $\mathcal{Y}_i$. Said another way, the EFS condition requires that the length of all projected points be less than the inradius of $\mathcal{Y}_i$.

In Figure 3, we provide a geometric visualization of the EFS condition for a union of disjoint subspaces (union of a 1D subspace with a 2D subspace). In (a), we show an example where EFS is guaranteed because the projection of the points outside of the 2D subspace lie well within the antipodal convex hull of the points along the normalized 2D subspace (ring). In (b), we show an example where EFS is not guaranteed because the projection of the points outside of the 2D subspace lie outside of the antipodal convex hull of the points along the normalized 2D subspace (ring).

### 3.3 Connections to Previous Work

In this section, we will connect our results for OMP with previous analyses of EFS with BP for disjoint (Elhamifar and Vidal, 2010, 2013) and intersecting (Soltanolkotabi and Candès, 2012) subspaces. Following this, we will contrast the geometry underlying EFS with exact recovery conditions used to guarantee support recovery for both OMP and BP (Tropp, 2004, 2006).

#### 3.3.1 SUBSPACE CLUSTERING WITH BP

Elhamifar and Vidal (2010) develop the following sufficient condition for EFS to occur for BP from a union of disjoint subspaces,

$$\max_{j \neq i} \cos(\theta_{ij}^*) \; < \; \max_{\widetilde{Y}_i \in \mathbb{W}_i} \frac{\sigma_{\min}(\widetilde{Y}_i)}{\sqrt{k_i}}, \tag{9}$$

where $\mathbb{W}_i$ is the set of all full rank sub-matrices $\widetilde{Y}_i \in \mathbb{R}^{n \times k_i}$ of the data matrix $Y_i \in \mathbb{R}^{n \times d_i}$ and $\sigma_{\min}(\widetilde{Y}_i)$ is the minimum singular value of the sub-matrix $\widetilde{Y}_i$. Since we assume that all of the data points have been normalized, $\sigma_{\min}(\widetilde{Y}_i) \leq 1$; thus, the best case result that can be obtained is that the minimum principal angle, $\cos(\theta_{ij}^*) < 1/\sqrt{k_i}$. This suggests that the minimum principal angle of the union must go to zero, that is, the union must consist of orthogonal subspaces, as the subspace dimension increases.

In contrast to the condition in (9), the conditions we provide in Theorem 1 and Corollary 1 do not depend on the subspace dimension. Rather, we require that there are enough points in each subspace to achieve a sufficiently small covering; in which case, EFS can be guaranteed for subspaces of any dimension.

Soltanolkotabi and Candès (2012) develop the following sufficient condition for EFS to occur for BP from a union of intersecting subspaces,

$$\mu_v(\mathcal{Y}_i) = \max_{y \in \mathcal{Y}_{-i}} \|V_{(i)}{}^T y\|_\infty < r(\mathcal{Y}_i), \tag{10}$$

where the matrix $V_{(i)} \in \mathbb{R}^{d_i \times n}$ contains the dual directions (the dual vectors for each point in $\mathcal{Y}_i$ embedded in $\mathbb{R}^n$) in its columns,[3] and $r(\mathcal{Y}_i)$ is the inradius as defined in (6). In words, (10) requires that the maximum coherence between any point in $\mathcal{Y}_{-i}$ and the dual directions contained in $V_{(i)}$ be less than the inradius of the points in $\mathcal{Y}_i$.

To link the result in (10) to our guarantee for OMP in Theorem 1, we observe that while (10) requires that $\mu_v(\mathcal{Y}_i)$ (coherence between a point in a subspace cluster and the dual directions of points in a different subspace cluster) be less than the inradius, Theorem 1 requires that the mutual coherence $\mu_c(\mathcal{Y}_i)$ (coherence between two points in different subspace clusters) be less than the inradius minus an additional term that depends on the covering radius. For an arbitrary set of points that live on a union of subspaces, the precise relationship between the two coherence parameters $\mu_c(\mathcal{Y}_i)$ and $\mu_v(\mathcal{Y}_i)$ is not straightforward; however, when the points in each subspace cluster are distributed uniformly and at random along each subspace, the dual directions will also be distributed uniformly along each subspace.[4] In this case, $\mu_v(\mathcal{Y}_i)$ will be roughly equivalent to the mutual coherence $\mu_c(\mathcal{Y}_i)$.

This simplification reveals the connection between the result in (10) for BP and the condition in Theorem 1 for OMP. In particular, when $\mu_v(\mathcal{Y}_i) \approx \mu_c(\mathcal{Y}_i)$, our result for OMP requires that the mutual coherence is smaller than the inradius minus an additional term that is linear in the covering diameter $\varepsilon$. For this reason, our result in Theorem 1 is more restrictive than the result provided in (10). The gap between the two bounds shrinks to zero only when the minimum principal angle $\theta^*_{ij} \to \pi/2$ (orthogonal subspaces) or when the covering diameter $\varepsilon \to 0$.

In our empirical studies, we find that when BPDN is tuned to an appropriate value of the noise parameter $\kappa$, BPDN tends to produce higher rates of EFS than OMP. This suggests that the theoretical gap between the two approaches might not be an artifact of our current analysis; rather, there might exist an intrinsic gap between the performance of each method with respect to EFS. Nonetheless, an interesting finding from our empirical study in Section 5.4, is that despite the fact that BPDN provides better rates of EFS than OMP, OMP typically provides better clustering results than BPDN. For these reasons, we maintain that OMP offers a powerful low-complexity alternative to $\ell_1$-minimization approaches for feature selection.

### 3.3.2 EXACT RECOVERY CONDITIONS FOR SPARSE RECOVERY

To provide further intuition about EFS in endogenous sparse recovery, we will compare the geometry underlying the EFS condition with the geometry of the exact recovery condition (ERC) for sparse signal recovery methods (Tropp, 2004, 2006).

To guarantee exact support recovery for a signal $y \in \mathbb{R}^n$ which has been synthesized from a linear combination of atoms from the sub-matrix $\Phi_\Lambda \in \mathbb{R}^{n \times k}$, we must ensure that our approximation of $y$ consists solely of atoms from $\Phi_\Lambda$. Let $\{\varphi_i\}_{i \notin \Lambda}$ denote the set of atoms in $\Phi$ that are not indexed by the set $\Lambda$. The *exact recovery condition* (ERC) in Theorem 2 is sufficient to guarantee that we obtain exact support recovery for both BP and OMP (Tropp, 2004).

3. See Def. 2.2 for a formal definition of the dual directions and insight into the geometry underlying their guarantees for EFS via BP (Soltanolkotabi and Candès, 2012).
4. This approximation is based upon personal correspondence with M. Soltankotabi, an author of the work in Soltanolkotabi and Candès (2012).

**Theorem 2 (Tropp, 2004)** *For any signal supported over the sub-dictionary $\Phi_\Lambda$, exact support recovery is guaranteed for both OMP and BP if*

$$\mathrm{ERC}(\Lambda) = \max_{i \notin \Lambda} \|\Phi_\Lambda{}^\dagger \varphi_i\|_1 < 1.$$

A geometric interpretation of the ERC is that it provides a measure of how far a projected atom $\varphi_i$ outside of the set $\Lambda$ lies from the antipodal convex hull of the atoms in $\Lambda$. When a projected atom lies outside of the antipodal convex hull formed by the set of points in the sub-dictionary $\Phi_\Lambda$, then the ERC condition is violated and support recovery is not guaranteed. For this reason, the ERC requires that the maximum coherence between the atoms in $\Phi$ is sufficiently low or that $\Phi$ is *incoherent*.

While the ERC condition requires a *global incoherence* property on all of the columns of $\Phi$, we can interpret EFS as requiring a *local incoherence* property. In particular, the EFS condition requires that the projection of atoms in an incorrect subspace cluster $\mathcal{Y}_{-i}$ onto $\mathcal{S}_i$ must be incoherent with any deep holes in $\mathcal{Y}_i$ along $\mathcal{S}_i$. In contrast, we require that the points within a subspace cluster exhibit local coherence in order to produce a small covering radius.

## 4. EFS for Bounded Unions of Subspaces

In this section, we study the connection between EFS and the higher-order principal angles (beyond the minimum angle) between pairs of intersecting subspaces.

### 4.1 Subspace Distances

To characterize the "distance" between pairs of subspaces in the ensemble, the *principal angles* between subspaces will prove useful. As we saw in the previous section, the first principal angle $\theta_0$ between subspaces $\mathcal{S}_1$ and $\mathcal{S}_2$ of dimension $k_1$ and $k_2$ is defined as the smallest angle between a pair of unit vectors $(u_1, v_1)$ drawn from $\mathcal{S}_1 \times \mathcal{S}_2$. The vector pair $(u_1^*, v_1^*)$ that attains this minimum is referred to as the first set of principal vectors. The second principal angle $\theta_1$ is defined much like the first, except that the second set of principal vectors that define the second principal angle are required to be orthogonal to the first set of principal vectors $(u_1^*, v_1^*)$. The remaining principal angles are defined recursively in this way. The sequence of $k = \min(k_1, k_2)$ principal angles, $\theta_0 \leq \theta_1 \leq \cdots \leq \theta_{k-1}$, is non-decreasing and all of the principal angles lie between $[0, \pi/2]$.

The definition above provides insight into what the principal angles/vectors tell us about the geometry underlying a pair of subspaces; in practice, however, the principal angles are not computed in this recursive manner. Rather, a computationally efficient way to compute the principal angles between two subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$ is to first compute the singular values of the matrix $G = \Phi_i^T \Phi_j$, where $\Phi_i \in \mathbb{R}^{n \times k_i}$ is an ONB that spans subspace $\mathcal{S}_i$. Let $G = U \Sigma V^T$ denote the SVD of $G$ and let $\sigma_{ij} \in [0,1]^k$ denote the singular values of $G$, where $k = \min(k_i, k_j)$ is the minimum dimension of the two subspaces. The $m^{\text{th}}$ smallest principal angle $\theta_{ij}(m)$ is related to the $m^{\text{th}}$ largest entry of $\sigma_{ij}$ via the following relationship, $\cos(\theta_{ij}(m)) = \sigma_{ij}(m)$. For our subsequent discussion, we will refer to the singular values of $G$ as the *cross-spectra* of the subspace pair $(\mathcal{S}_i, \mathcal{S}_j)$.

A pair of subspaces is said to be *disjoint* if the minimum principal angle is greater than zero. Non-disjoint or intersecting subspaces are defined as subspaces with minimum principal angle equal to zero. The dimension of the intersection between two subspaces is equivalent to the number of principal angles equal to zero or equivalently, the number of entries of the cross-spectra that

are equal to one. We define the *overlap* between two subspaces as the $\text{rank}(G)$ or equivalently, $q = \|\sigma_{ij}\|_0$, where $q \geq \dim(\mathcal{S}_i \cap \mathcal{S}_j)$.

### 4.2 Sufficient Conditions for EFS from Bounded Unions

The sufficient conditions for EFS in Theorem 1 and Corollary 1 reveal an interesting relationship between the covering radius, mutual coherence, and the minimum principal angle between pairs of subspaces in the ensemble. However, we have yet to reveal any dependence between EFS and higher-order principal angles. To make this connection more apparent, we will make additional assumptions about the distribution of points in the ensemble, namely that the data set produces a *bounded union of subspaces* relative to the principal vectors supporting pairs of subspaces in the ensemble.

Let $Y = [Y_i\, Y_j]$ denote a collection of unit-norm data points, where $Y_i$ and $Y_j$ contain the points in subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$, respectively. Let $G = \Phi_i^T \Phi_j = U \Sigma V^T$ denote the SVD of $G$, where $\text{rank}(G) = q$. Let $\widetilde{U} = \Phi_i U_q$ denote the set of left principal vectors of $G$ that are associated with the $q$ nonzero singular values in $\Sigma$. Similarly, let $\widetilde{V} = \Phi_j V_q$ denote the set of right principal vectors of $G$ that are associated with the nonzero singular values in $\Sigma$. When the points in each subspace are incoherent with the principal vectors in the columns of $\widetilde{U}$ and $\widetilde{V}$, we say that the ensemble $Y$ is an *bounded union of subspaces*. Formally, we require the following incoherence property holds:

$$\left( \|Y_i^T \widetilde{U}\|_\infty, \|Y_j^T \widetilde{V}\|_\infty \right) \leq \gamma, \tag{11}$$

where $\|\cdot\|_\infty$ is the entry-wise maximum and $\gamma \in (0,1]$. This property requires that the inner products between the points in a subspace and the set of principal vectors that span non-orthogonal directions between a pair of subspaces is bounded by a fixed constant.

When the points in each subspace are distributed such that (11) holds, we can rewrite the mutual coherence between any two points from different subspaces to reveal its dependence on higher-order principal angles. In particular, we show (in Section 7.2) that the coherence between the residual $s$ used in Algorithm 1 to select the next point to be included in the representation of a point $y \in \mathcal{Y}_i$, and a point in $\mathcal{Y}_j$ is upper bounded by

$$\max_{y \in \mathcal{Y}_j} \frac{|\langle s, y \rangle|}{\|s\|_2} \leq \gamma \|\sigma_{ij}\|_1, \tag{12}$$

where $\gamma$ is the bounding constant of the data $Y$ and $\|\sigma_{ij}\|_1$ is the $\ell_1$-norm of the cross-spectra or equivalently, the trace norm of $G$. Using the bound in (12), we arrive at the following sufficient condition for EFS from bounded unions of subspaces. We provide the proof in Section 7.2.

**Theorem 3** *Let $Y$ live on a bounded union of subspaces, where $q = \text{rank}(G)$ and $\gamma < \sqrt{1/q}$. Let $\sigma_{ij}$ denote the cross-spectra of the subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$ and let $\varepsilon$ denote the covering diameter of $\mathcal{Y}_i$. A sufficient condition for Algorithm 1 to return exact feature sets for all points in $\mathcal{Y}_i$ is that the covering diameter*

$$\varepsilon \;<\; \min_{j \neq i} \sqrt{1 - \gamma^2 \|\sigma_{ij}\|_1^2}.$$

This condition requires that both the covering diameter of each subspace and the bounding constant of the union be sufficiently small in order to guarantee EFS. One way to guarantee that the

ensemble has a small bounding constant is to constrain the total amount of energy that points in $\mathcal{Y}_j$ have in the $q$-dimensional subspace spanned by the principal vectors in $\widetilde{V}$.

Our analysis for bounded unions assumes that the nonzero entries of the cross-spectra are equal, and thus each pair of supporting principal vectors in $\widetilde{V}$ are equally important in determining whether points in $\mathcal{Y}_i$ will admit EFS. However, this assumption is not true in general. When the union is supported by principal vectors with non-uniform principal angles, our analysis suggests that a weaker form of incoherence is required. Instead of requiring incoherence with all principal vectors, the data must be sufficiently incoherent with the principal vectors that correspond to small principal angles (or large values of the cross-spectra). This means that as long as points are not concentrated along the principal directions with small principal angles (i.e., intersections), then EFS can be guaranteed, even when subspaces exhibit non-trivial intersections. To test this prediction, we will study EFS for a *bounded energy model* in Section 5.2. We show that when the data set is sparsely sampled (larger covering radius), reducing the amount of energy that points contain in subspace intersections, does in fact increase the probability that points admit EFS.

Finally, our analysis of bounded unions suggests that the decay of the cross-spectra is likely to play an important role in determining whether points will admit EFS or not. To test this hypothesis, we will study the role that the structure of the cross-spectra plays in EFS in Section 5.3.

## 5. Experimental Results

In our theoretical analysis of EFS in Sections 3 and 4, we revealed an intimate connection between the covering radius of subspaces and the principal angles between pairs of subspaces in the ensemble. In this section, we will conduct an empirical study to explore these connections further. In particular, we will study the probability of EFS as we vary the covering radius as well as the dimension of the intersection and/or overlap between subspaces.

### 5.1 Generative Model for Synthetic Data

In order to study EFS for unions of subspaces with varied cross-spectra, we will generate synthetic data from unions of overlapping *block sparse signals*.

#### 5.1.1 CONSTRUCTING SUB-DICTIONARIES

We construct a pair of sub-dictionaries as follows: Take two subsets $\Omega_1$ and $\Omega_2$ of $k$ signals (atoms) from a dictionary $D$ containing $M$ atoms $\{d_m\}_{m=1}^M$ in its columns, where $d_m \in \mathbb{R}^n$ and $|\Omega_1| = |\Omega_2| = k$. Let $\Psi \in \mathbb{R}^{n\times k}$ denote the subset of atoms indexed by $\Omega_1$, and let $\Phi \in \mathbb{R}^{n\times k}$ denote the subset of atoms indexed by $\Omega_2$. Our goal is to select $\Psi$ and $\Phi$ such that $G = \Psi^T\Phi$ is diagonal, that is, $\langle \psi_i, \phi_j \rangle = 0$, if $i \neq j$, where $\psi_i$ is the $i^{\text{th}}$ element in $\Psi$ and $\phi_j$ is the $j^{\text{th}}$ element of $\Phi$. In this case, the cross-spectra is defined as $\sigma = \text{diag}(G)$, where $\sigma \in [0,1]^k$. For each union, we fix the "overlap" $q$ or the rank of $G = \Psi^T\Phi$ to a constant between zero (orthogonal subspaces) and $k$ (maximal overlap).

To generate a pair of $k$-dimensional subspaces with a $q$-dimensional overlap, we can pair the elements from $\Psi$ and $\Phi$ such that the $i^{\text{th}}$ entry of the cross-spectra equals

$$\sigma(i) = \begin{cases} |\langle \psi_i, \phi_i \rangle| & \text{if } 1 \leq i \leq q, \\ 0 & \text{if } i = q+1 \leq i \leq k. \end{cases}$$

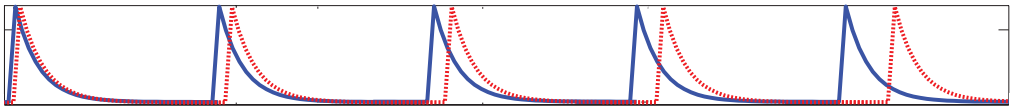


Figure 4: Generating unions of subspaces from shift-invariant dictionaries. An example of a collection of two sub-dictionaries of five atoms, where each of the atoms have a non-zero inner product with one other atom. This choice of sub-dictionaries produces a union of disjoint subspaces, where the overlap ratio $\delta = q/k = 1$.

We can leverage the banded structure of shift-invariant dictionaries, for example, dictionary matrices with localized Toeplitz structure, to generate subspaces with structured cross-spectra as follows.[5] First, we fix a set of $k$ incoherent (orthogonal) atoms from our shift-invariant dictionary, which we place in the columns of $\Psi$. Now, holding $\Psi$ fixed, we set the $i^{\text{th}}$ atom $\phi_i$ of the second sub-dictionary $\Phi$ to be a shifted version of the $i^{\text{th}}$ atom $\psi_i$ of the dictionary $\Psi$. To be precise, if we set $\psi_i = d_m$, where $d_m$ is the $m^{\text{th}}$ atom in our shift-invariant dictionary, then we will set $\phi_i = d_{m+\Delta}$ for a particular shift $\Delta$. By varying the shift $\Delta$, we can control the coherence between $\psi_i$ and $\varphi_i$. In Figure 4, we show an example of one such construction for $k = q = 5$. Since $\sigma \in (0,1]^k$, the worst-case pair of subspaces with overlap equal to $q$ is obtained when we pair $q$ identical atoms with $k-q$ orthogonal atoms. In this case, the cross-spectra attains its maximum over its entire support and equals zero otherwise. For such unions, the overlap $q$ equals the dimension of the intersection between the subspaces. We will refer to this class of block-sparse signals as *orthoblock sparse signals*.

### 5.1.2 COEFFICIENT SYNTHESIS

To synthesize a point that lives in the span of the sub-dictionary $\Psi \in \mathbb{R}^{n\times k}$, we combine the elements $\{\psi_1, \ldots, \psi_k\}$ and subspace coefficients $\{\alpha(1), \ldots, \alpha(k)\}$ linearly to form

$$y_i = \sum_{j=1}^{k} \psi_j \alpha(j),$$

where $\alpha(j)$ is the subspace coefficient associated with the $j^{\text{th}}$ column in $\Psi$. Without loss of generality, we will assume that the elements in $\Psi$ are sorted such that the values of the cross-spectra are monotonically decreasing. Let $y_i^c = \sum_{j=1}^{q} \psi_j \alpha_i(j)$ be the "common component" of $y_i$ that lies in the space spanned by the principal directions between the pair of subspaces that correspond to non-orthogonal principal angles between $(\Phi, \Psi)$ and let $y_i^d = \sum_{j=q+1}^{k} \psi_j \alpha(j)$ denote the "disjoint component" of $y_i$ that lies in the space orthogonal to the space spanned by the first $q$ principal directions.

For our experiments, we consider points drawn from one of the two following coefficient distributions, which we will refer to as *(M1)* and *(M2)* respectively.

5. While shift-invariant dictionaries appear in a wide range of applications of sparse recovery (Mailhè et al., 2008; Dyer et al., 2010), we introduce the idea of using shift-invariant dictionaries to create structured unions of subspaces for the first time here.

- *(M1) Uniformly Distributed on the Sphere:* Generate subspace coefficients according to a standard normal distribution and map the point to the unit sphere

$$y_i = \frac{\sum_j \psi_j \alpha(j)}{\|\sum_j \psi_j \alpha(j)\|_2}, \quad \text{where } \alpha(j) \sim \mathcal{N}(0,1).$$

- *(M2) Bounded Energy Model:* Generate subspace coefficients according to *(M1)* and rescale each coefficient in order to bound the energy in the common component

$$y_i = \frac{\tau\, y_i^c}{\|y_i^c\|_2} + \frac{(1-\tau) y_i^d}{\|y_i^d\|_2}.$$

By simply restricting the total energy that each point has in its common component, the bounded energy model *(M2)* can be used to produce ensembles with small bounding constant to test the predictions in Theorem 3.

### 5.2 Phase Transitions for OMP

The goal of our first experiment is to study the probability of EFS—the probability that a point in the ensemble admits exact features—as we vary both the number and distribution of points in each subspace as well as the dimension of the intersection between subspaces. For this set of experiments, we generate a union of orthoblock sparse signals, where the overlap equals the dimension of the intersection.

Along the top row of Figure 5, we display the probability of EFS for orthoblock sparse signals generated according to the coefficient model *(M1)*: the probability of EFS is computed as we vary the *overlap ratio* $\delta = q/k \in [0,1]$ in conjunction with the *oversampling ratio* $\rho = k/d \in [0,1]$, where $q = \text{rank}(\Phi_1^T \Phi_2)$ equals the dimension of the intersection between the subspaces, and $d$ is the number of points per subspace. Along the bottom row of Figure 5, we display the probability of EFS for orthoblock sparse signals generated according to the coefficient model *(M2)*: the probability of EFS is computed as we vary the overlap ratio $\delta$ and the amount of energy $\tau \in [0,1)$ each point has within its common component. For these experiments, the subspace dimension is set to $k = 20$ (left) and $k = 50$ (right). To see the phase boundary that arises when we approach critical sampling (i.e., $\rho \approx 1$), we display our results in terms of the logarithm of the oversampling ratio. For these experiments, the results are averaged over 500 trials.

As our theory predicts, the oversampling ratio has a strong impact on the degree of overlap between subspaces that can be tolerated before EFS no longer occurs. In particular, as the number of points in each subspace increases (covering radius decreases), the probability of EFS obeys a second-order phase transition, that is, there is a graceful degradation in the probability of EFS as the dimension of the intersection increases. When the pair of subspaces are densely sampled, the phase boundary is shifted all the way to $\delta = 0.7$, where70% of the dimensions of each subspace intersect. This is due to the fact that as each subspace is sampled more densely, the covering radius becomes sufficiently small to ensure that even when the overlap between planes is high, EFS still occurs with high probability. In contrast, when the subspaces are critically sampled, that is, the number of points per subspace $d \approx k$, only a small amount of overlap can be tolerated, where $\delta < 0.1$. In addition to shifting the phase boundary, as the oversampling ratio increases, the width of the transition region (where the probability of EFS goes from zero to one) also increases.

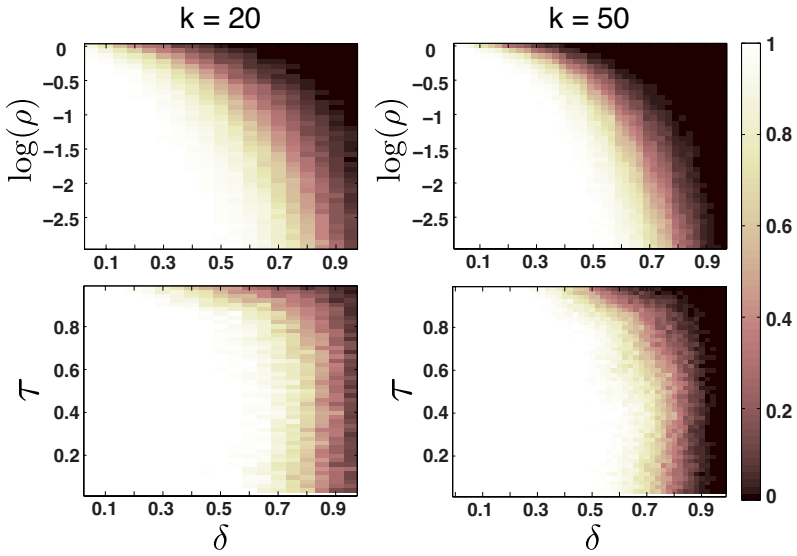



Figure 5: Probability of EFS for different coefficient distributions. The probability of EFS for a union of two subspaces of dimension $k = 20$ (left column) and $k = 50$ (right column). The probability of EFS is displayed as a function of the overlap ratio $\delta \in [0,1)$ and the logarithm of the oversampling ratio $\log(\rho)$ (top row) and the mutual energy $\tau = \|y_c\|_2$ (bottom row) .

Along the bottom row of Figure 5, we study the impact of the bounding constant on EFS, as discussed in Section 4.2. In this experiment, we fix the oversampling ratio to $\rho = 0.1$ and vary the common energy $\tau$ in conjunction with the overlap ratio $\delta$. By reducing the bounding constant of the union, the phase boundary for the uniformly distributed data from model *(M1)* is shifted from $\delta = 0.45$ to $\delta = 0.7$ for both $k = 20$ and $k = 50$. This result confirms our predictions in the discussion of Theorem 3 that by reducing the amount of energy that points have in their subspace intersections EFS will occur for higher degrees of overlap. Another interesting finding of this experiment is that, once $\tau$ reaches a threshold, the phase boundary remains constant and further reducing the bounding constant has no impact on the phase transitions for EFS.

### 5.3 Comparison of OMP and NN

In this section, we compare the probability of EFS for feature selection with OMP and nearest neighbors (NN). First, we compare the performance of both feature selection methods for unions with different cross-spectra. Second, we compare the phase transitions for unions of orthoblock sparse signals as we vary the overlap and oversampling ratio.

For our experiments, we generate pairs of subspaces with structured cross-spectra as described in Section 5.1.1. The cross-spectra arising from three different unions of block-sparse signals are displayed along the top row of Figure 6. On the left, we show the cross-spectra for a union of orthoblock sparse signals with overlap ratio $\delta = 0.75$, where $q = 15$ and $k = 20$. The cross-spectra obtained by pairing shifted Lorentzian and exponential atoms are displayed in the middle and right

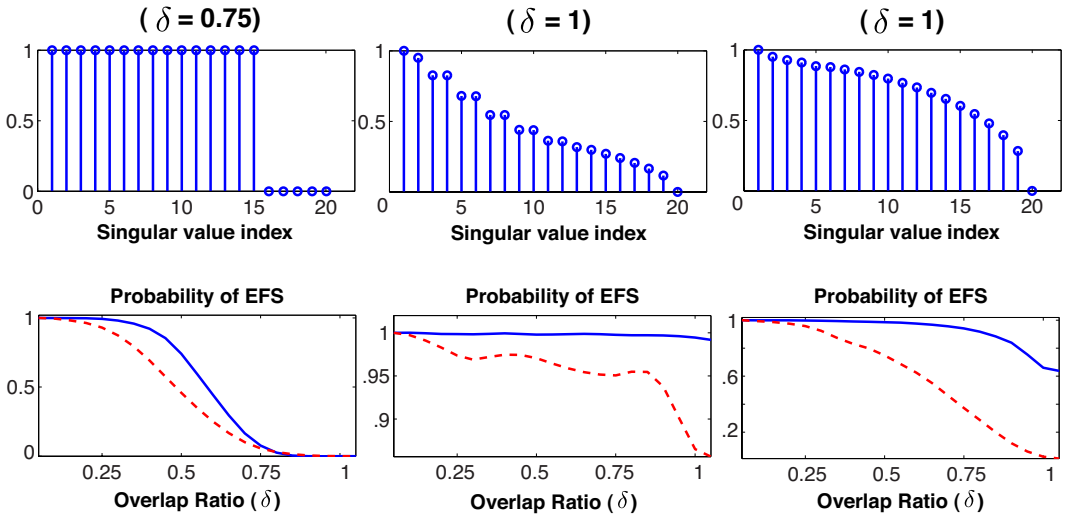



Figure 6: Probability of EFS for unions with structured cross-spectra. Along the top row, we show the cross-spectra for different unions of block-sparse signals. Along the bottom row, we show the probability of EFS as we vary the overlap ratio $\delta \in [0,1]$ for OMP (solid) and NN (dash).

columns, respectively. Along the bottom row of Figure 6, we show the probability of EFS for OMP and NN for each of these three subspace unions as we vary the overlap $q$. To do this, we generate subspaces by setting their cross-spectra equal to the first $q$ entries equal to the cross-spectra in Figure 6 and setting the remaining $k-q$ entries of the cross-spectra equal to zero. Each subspace cluster is generated by sampling $d = 100$ points from each subspace according to the coefficient model *(M1)*.

This study provides a number of interesting insights into the role that higher-order principal angles between subspaces play in feature selection for both sparse recovery methods and NN. First, we observe that the gap between the probability of EFS for OMP and NN is markedly different for each of the three unions. In the first union of orthoblock sparse signals, the probability of EFS for OMP lies strictly above that obtained for the NN method, but the gap between the performance of both methods is relatively small. In the second union, both methods maintain a high probability of EFS, with OMP admitting nearly perfect feature sets even when the overlap ratio is maximal. In the third union, we observe that the gap between EFS for OMP and NN is most pronounced. In this case, the probability of EFS for NN sets decreases to 0.1, while OMP admits a very high probability of EFS, even when the overlap ratio is maximal. in summary, we observe that when data is distributed uniformly with respect to all of the principal directions between a pair of subspaces and the cross-spectra is sub-linear, then EFS may be guaranteed with high probability for all points in the set provided the sampling density is sufficiently high. This is in agreement with the discussion of EFS bounded unions in Section 4.2. Moreover, these results further support our claims that in order to truly understand and predict the behavior of endogenous sparse recovery from unions of subspaces, we require a description that relies on the entire cross-spectra.

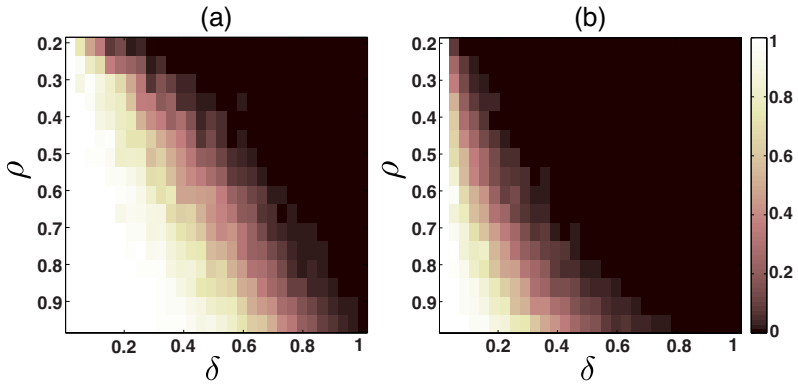



Figure 7: Phase transitions for OMP and NN. The probability of EFS for orthoblock sparse signals for OMP (a) and NN (b) feature sets as a function of the oversampling ratio $\rho = k/d$ and the overlap ratio $\delta = q/k$, where $k = 20$.

In Figure 7, we display the probability of EFS for OMP (left) and sets of NN (right) as we vary the overlap and the oversampling ratio. For this experiment, we consider unions of orthoblock sparse signals living on subspaces of dimension $k = 50$ and vary $\rho \in [0.2, 0.96]$ and $\delta \in [1/k, 1]$. An interesting result of this study is that there are regimes where the probability of EFS equals zero for NN but occurs for OMP with a non-trivial probability. In particular, we observe that when the sampling of each subspace is sparse (the oversampling ratio is low), the gap between OMP and NN increases and OMP significantly outperforms NN in terms of their probability of EFS. Our study of EFS for structured cross-spectra suggests that the gap between NN and OMP should be even more pronounced for cross-spectra with superlinear decay.

### 5.4 Clustering Illumination Subspaces

In this section, we compare the performance of sparse recovery methods, that is, BP and OMP, with NN for clustering unions of *illumination subspaces* arising from a collection of images of faces under different lighting conditions. By fixing the camera center and position of the persons face and capturing multiple images under different lighting conditions, the resulting images can be well-approximated by a 5-dimensional subspace (Ramamoorthi, 2002).

In Figure 2, we show three examples of the subspace affinity matrices obtained with NN, BP, and OMP for two different faces under 64 different illumination conditions from the Yale Database B (Georghiades et al., 2001), where each image has been subsampled to $48 \times 42$ pixels, with $n = 2016$. In all of the examples, the data is sorted such that the images for each face are placed in a contiguous block.

To generate the NN affinity matrices in the left column of Figure 2, we compute the absolute normalized inner products between all points in the data set and then threshold each row to select the $k = 5$ nearest neighbors to each point. To generate the OMP affinity matrices in the right column, we employ Step 1 of Algorithm 2 with the maximum sparsity set to $k = 5$. To generate the BP affinity matrices in the middle column, we solved the BP denoising (BPDN) problem in (3) via a homotopy algorithm where we vary the noise parameter $\kappa$ and choose the smallest value of $\kappa$ that produces

| | | Full-data | | | Half-data | | | Quarter-data | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | OMP | $\ell_1$ | NN | OMP | $\ell_1$ | NN | OMP | $\ell_1$ | NN |
| **EFS** | Mean | 55.48 | **73.48** | 60.71 | 39.74 | **52.50** | 39.50 | 21.27 | **28.78** | 14.89 |
| (%) | Median | 55.91 | **75.00** | 62.5 | 39.06 | **53.13** | 39.68 | 18.36 | **28.13** | 12.50 |
| **Clustering error** | Mean | **1.43** | 3.69 | 22.03 | **4.91** | 13.65 | 37.42 | **10.18** | 20.14 | 39.63 |
| (%) | Median | **0.78** | **0.78** | 15.63 | **3.13** | 6.25 | 42.19 | **6.25** | 18.75 | 40.63 |

Table 1: Classification and EFS rates for illumination subspaces. Shown are the aggregate results obtained over $\binom{38}{2}$ pairs of subspaces.

$k \leq 5$ coefficients.[6] The resulting coefficient vectors are then stacked into the rows of a matrix $C$ and the final subspace affinity $W$ is computed by symmetrizing the coefficient matrix, $W = |C| + |C^T|$.

After computing the subspace affinity matrix for each of these three feature selection methods, we employ a spectral clustering approach which partitions the data based upon the eigenvector corresponding to the smallest nonzero eigenvalue of the graph Laplacian of the affinity matrix (Shi and Malik, 2000; Ng et al., 2002). For all three feature selection methods, we obtain the best clustering performance when we cluster the data based upon the graph Laplacian instead of the normalized graph Laplacian (Shi and Malik, 2000). In Table 1, we display the percentage of points that resulted in EFS and the classification error for all pairs of $\binom{38}{2}$ subspaces in the Yale B database. Along the top row, we display the mean and median percentage of points that resulted in EFS for the full data set (all 64 illumination conditions), half of the data set (32 illumination conditions selected at random in each trial), and a quarter of the data set (16 illumination conditions selected at random in each trial). Along the bottom row, we display the clustering error (percentage of points that were incorrectly classified) for SSC-OMP, SSC, and NN-based clustering (spectral clustering of the NN affinity matrix).

While both sparse recovery methods (BPDN and OMP) admit EFS rates that are comparable to NN on the full data set, we find that sparse recovery methods provide higher rates of EFS than NN when the sampling of each subspace is sparse, that is, the half and quarter data sets. These results are also in agreement with our experiments on synthetic data. A surprising result is that SSC-OMP provides better clustering performance than SSC on this particular data set, even though BP provides higher rates of EFS.

## 6. Discussion

In this section, we provide insight into the implications of our results for different applications of sparse recovery and compressive sensing. Following this, we end with some open questions and directions for future research.

### 6.1 "Data Driven" Sparse Approximation

The standard paradigm in signal processing and approximation theory is to compute a representation of a signal in a fixed and pre-specified basis or overcomplete dictionary. In most cases, the dictio-

6. We also studied another variant of BPDN where we solve OMP for $k = 5$, compute the error of the resulting approximation, and then use this error as the noise parameter $\kappa$. However, this variant provided worse results than those reported in Table 1.

naries used to form these representations are designed according to some mathematical desiderata. A more recent approach has been to learn a dictionary from a collection of data, such that the data admit a sparse representation with respect to the learned dictionary (Olshausen and Field, 1997; Aharon et al., 2006).

The applicability and utility of endogenous sparse recovery in subspace learning draws into question whether we can use endogenous sparse recovery for other tasks, including approximation and compression. The question that naturally arises is, "do we design a dictionary, learn a dictionary, or use the data as a dictionary?" Understanding the advantages and tradeoffs between each of these approaches is an interesting and open question.

### 6.2 Learning Block-Sparse Signal Models

Block-sparse signals and other structured sparse signals have received a great deal of attention over the past few years, especially in the context of compressive sensing from structured unions of subspaces (Lu and Do, 2008; Blumensath and Davies, 2009) and in model-based compressive sensing (Baraniuk et al., 2010). In all of these settings, the fact that a class or collection of signals admit structured support patterns is leveraged in order to obtain improved recovery of sparse signals in noise and in the presence of undersampling.

To exploit such structure in sparse signals—especially in situations where the structure of signals or blocks of active atoms may be changing across different instances in time, space, etc.—the underlying subspaces that the signals occupy must be learned directly from the data. The methods that we have described for learning union of subspaces from ensembles of data can be used in the context of learning block sparse and other structured sparse signal models. The application of subspace clustering methods for this purpose is an interesting direction for future research.

### 6.3 Beyond Coherence

While the maximum and cumulative coherence provide measures of the uniqueness of sub-dictionaries that are necessary to guarantee exact signal recovery for sparse recovery methods (Tropp, 2004) , our current study suggests that examining the principal angles formed from pairs of sub-dictionaries could provide an even richer description of the geometric properties of a dictionary. Thus, a study of the principal angles formed by different subsets of atoms from a dictionary might provide new insights into the performance of sparse recovery methods with coherent dictionaries and for compressive sensing from structured matrices. In addition, our empirical results in Section 5.3 suggest that there might exist an intrinsic difference between sparse recovery from dictionaries that exhibit sublinear versus superlinear decay in their principal angles or cross-spectra. It would be interesting to explore whether these two "classes" of dictionaries exhibit different phase transitions for sparse recovery.

### 6.4 Discriminative Dictionary Learning

While dictionary learning was originally proposed for learning dictionaries that admit sparse representations of a collection of signals (Olshausen and Field, 1997; Aharon et al., 2006), dictionary learning has recently been employed for classification. To use learned dictionaries for classification, a dictionary is learned for each class of training signals and then a sparse representation of a test signal is formed with respect to each of the learned dictionaries. The idea is that the test signal will

admit a more compact representation with respect to the dictionary that was learned from the class of signals that the test signal belongs to.

Instead of learning these dictionaries independently of one another, *discriminative dictionary learning* (Mairal et al., 2008; Ramirez et al., 2010), aims to learn a collection of dictionaries $\{\Phi_1, \Phi_2, \ldots, \Phi_p\}$ that are incoherent from one another. This is accomplished by minimizing either the spectral (Mairal et al., 2008) or Frobenius norm (Ramirez et al., 2010) of the matrix product $\Phi_i^T \Phi_j$ between pairs of dictionaries. This same approach may also be used to learn sensing matrices for CS that are incoherent with a learned dictionary (Mailhè et al., 2012).

There are a number of interesting connections between discriminative dictionary learning and our current study of EFS from collections of unions of subspaces. In particular, our study provides new insights into the role that the principal angles between two dictionaries tell us about our ability to separate classes of data based upon their sparse representations. Our study of EFS from unions with structured cross-spectra suggests that the decay of the cross-spectra between different data classes provides a powerful predictor of the performance of sparse recovery methods from data living on a union of low-dimensional subspaces. This suggests that in the context of discriminative dictionary learning, it might be more advantageous to reduce the $\ell_1$-norm of the cross-spectra rather than simply minimizing the maximum coherence and/or Frobenius norm between points in different subspaces. To do this, each class of data must first be embedded within a subspace, a ONB is formed for each subspace, and then the $\ell_1$- norm of the cross-spectra must be minimized. An interesting question is how one might impose such a constraint in discriminative dictionary learning methods.

### 6.5 Open Questions and Future Work

While EFS provides a natural measure of how well a feature selection algorithm will perform for the task of subspace clustering, our empirical results suggest that EFS does not necessarily predict the performance of spectral clustering methods when applied to the resulting subspace affinity matrices. In particular, we find that while OMP obtains lower rates of EFS than BPDN on real-world data, OMP yields better clustering results on the same data set. Understanding where this difference in performance might arise from is an interesting direction for future research.

Another interesting finding of our empirical study is that the gap between the rates of EFS for sparse recovery methods and NN depends on the sampling density of each subspace. In particular, we found that for dense samplings of each subspace, the performance of NN is comparable to sparse recovery methods; however, when each subspace is more sparsely sampled, sparse recovery methods provide significant gains over NN methods. This result suggests that endogenous sparse recovery provides a powerful strategy for clustering when the sampling of subspace clusters is sparse. Analyzing the gap between sparse recovery methods and NN methods for feature selection is an interesting direction for future research.

## 7. Proofs

In this section, we provide proofs for main theorems in the paper.

### 7.1 Proof of Theorem 1

Our goal is to prove that, if (7) holds, then it is sufficient to guarantee that EFS occurs for every point in $\mathcal{Y}_k$ when OMP is used for feature selection. We will prove this by induction.

Consider the greedy selection step in OMP (see Algorithm 1) for a point $y_i$ which belongs to the subspace cluster $\mathcal{Y}_k$. Recall that at the $m^{th}$ step of OMP, the point that is maximally correlated with the signal residual will be selected to be included in the feature set $\Lambda$. The normalized residual at the $m^{\text{th}}$ step is computed as

$$s^m = \frac{(I - P_\Lambda) y_i}{\|(I - P_\Lambda) y_i\|_2},$$

where $P_\Lambda = Y_\Lambda Y_\Lambda^\dagger \in \mathbb{R}^{n \times n}$ is a projector onto the subspace spanned by the points in the current feature set $\Lambda$, where $|\Lambda| = m - 1$.

To guarantee that we select a point from $\mathcal{S}_k$, we require that the following greedy selection criterion holds:

$$\max_{v \in \mathcal{Y}_k} |\langle s^m, v \rangle| > \max_{v \notin \mathcal{Y}_k} |\langle s^m, v \rangle|.$$

We will prove that this selection criterion holds at each step of OMP by developing an upper bound on the RHS (the maximum inner product between the residual and a point outside of $\mathcal{Y}_k$) and a lower bound on the LHS (the minimum inner product between the residual and a point in $\mathcal{Y}_k$).

First, we will develop the upper bound on the RHS. In the first iteration, the residual is set to the signal of interest ($y_i$). In this case, we can bound the RHS by the mutual coherence $\mu_c = \max_{i \neq j} \mu_c(\mathcal{Y}_i, \mathcal{Y}_j)$ across all other sets

$$\max_{y_j \notin \mathcal{Y}_k} |\langle y_i, y_j \rangle| \leq \mu_c.$$

Now assume that at the $m^{\text{th}}$ iteration we have selected points from the correct subspace cluster. This implies that our signal residual still lies within the span of $\mathcal{Y}_k$, and thus we can write the residual $s^m = z + e$, where $z$ is the closest point to $s^m$ in $\mathcal{Y}_k$ and $e$ is the remaining portion of the residual which also lies in $\mathcal{S}_k$. Thus, we can bound the RHS as follows

$$\begin{aligned}
\max_{y_j \notin \mathcal{Y}_k} |\langle s^m, y_j \rangle| &= \max_{y_j \notin \mathcal{Y}_k} |\langle z + e, y_j \rangle| \\
&\leq \max_{y_j \notin \mathcal{Y}_k} |\langle z, y_j \rangle| + |\langle e, y_j \rangle| \\
&\leq \mu_c + \max_{y_j \notin \mathcal{Y}_k} |\langle e, y_j \rangle| \\
&\leq \mu_c + \cos(\theta_0) \|e\|_2 \|y_i\|_2,
\end{aligned}$$

where $\theta_0$ is the minimum principal angle between $\mathcal{S}_k$ and all other subspaces in the ensemble.

Using the fact that $\text{cover}(\mathcal{Y}_k) = \varepsilon/2$, we can bound the $\ell_2$-norm of the vector $e$ as

$$\begin{aligned}
\|e\|_2 &= \|s - z\|_2 \\
&= \sqrt{\|s\|_2^2 + \|z\|_2^2 - 2|\langle s, z \rangle|} \\
&\leq \sqrt{2 - 2\sqrt{1 - (\varepsilon/2)^2}} \\
&= \sqrt{2 - \sqrt{4 - \varepsilon^2}}.
\end{aligned}$$

Plugging this quantity into our expression for the RHS, we arrive at the following upper bound

$$\max_{y_j \notin \mathcal{Y}_k} |\langle s^m, y_j \rangle| \leq \mu_c + \cos(\theta_0) \sqrt{2 - \sqrt{4 - \varepsilon^2}} < \mu_c + \cos(\theta_0) \frac{\varepsilon}{\sqrt[4]{12}},$$

where the final simplification comes from invoking the following Lemma.

**Lemma 1** For $0 \leq x \leq 1$,

$$\sqrt{2-\sqrt{4-x^2}} \leq \frac{x}{\sqrt[4]{12}}.$$

**Proof of Lemma 1:** We wish to develop an upper bound on the function

$$f(x) = 2-\sqrt{4-x^2}, \quad \text{for } 0 \leq x \leq 1.$$

Thus our goal is to identify a function $g(x)$, where $f'(x) \leq g'(x)$ for $0 \leq x \leq 1$, and $g(0) = f(0)$. The derivative of $f(x)$ can be upper bounded easily as follows

$$f'(x) = \frac{x}{\sqrt{4-x^2}} \leq \frac{x}{\sqrt{3}}, \quad \text{for } 0 \leq x \leq 1.$$

Thus, $g'(x) = x/\sqrt{3}$, and $g(x) = x^2/\sqrt{12}$; this ensures that $f'(x) \leq g'(x)$ for $0 \leq x \leq 1$, and $g(0) = f(0)$. By the Fundamental Theorem of Integral Calculus, $g(x)$ provides an upper bound for $f(x)$ over the domain of interest where, $0 \leq x \leq 1$. To obtain the final result, take the square root of both sides, $\sqrt{2-\sqrt{4-x^2}} \leq \sqrt{x^2/\sqrt{12}} = x/\sqrt[4]{12}$. □

Second, we will develop the lower bound on the LHS of the greedy selection criterion. To ensure that we select a point from $\mathcal{Y}_k$ at the first iteration, we require that $y_i$'s nearest neighbor belongs to the same subspace cluster. Let $y^i_{nn}$ denote the nearest neighbor to $y_i$

$$y^i_{nn} = \arg\max_{j \neq i} |\langle y_i, y_j \rangle|.$$

If $y^i_{nn}$ and $y_i$ both lie in $\mathcal{Y}_k$, then the first point selected via OMP will result in EFS.

Let us assume that the points in $\mathcal{Y}_k$ admit an $\varepsilon$-covering of the subspace cluster $\mathcal{S}_k$, or that $\text{cover}(\mathcal{Y}_k) = \varepsilon/2$. In this case, we have the following bound in effect

$$\max_{y_j \in \mathcal{Y}_k} |\langle s^m, y_j \rangle| \geq \sqrt{1-\frac{\varepsilon^2}{4}}.$$

Putting our upper and lower bound together and rearranging terms, we arrive at our final condition on the mutual coherence

$$\mu_c < \sqrt{1-\frac{\varepsilon^2}{4}} - \cos(\theta_0)\frac{\varepsilon}{\sqrt[4]{12}}.$$

Since we have shown that this condition is sufficient to guarantee EFS at each step of Algorithm 1 provided the residual stays in the correct subspace, Theorem 1 follows by induction. □

### 7.2 Proof of Theorem 3

To prove Theorem 3, we will assume that the union of subspaces is bounded in accordance with (11). This assumption enables us to develop a tighter upper bound on the mutual coherence between any residual signal $s \in \mathcal{S}_i$ and the points in $\mathcal{Y}_j$. Since $s \in \mathcal{S}_i$, the residual can be expressed as $s = \Phi_i \alpha$, where $\Phi_i \in \mathbb{R}^{n \times k_i}$ is an ONB that spans $\mathcal{S}_i$ and $\alpha = \Phi_i^T s$. Similarly, we can write each point in $\mathcal{Y}_j$ as

$y = \Phi_j \beta$, where $\Phi_j \in \mathbb{R}^{n \times k_j}$ is an ONB that spans $\mathcal{S}_j$, $\beta = \Phi_j^T y$. Let $\mathcal{B}_j = \{\Phi_j^T y_i\}_{i=1}^{d_j}$ denote the set of all subspace coefficients for all $y_i \in \mathcal{Y}_j$.

The coherence between the residual and a point in a different subspace can be expanded as follows:

$$\begin{aligned}
\max_{y \in \mathcal{Y}_j} \frac{|\langle s, y \rangle|}{\|s\|_2} &= \max_{\beta \in \mathcal{B}_j} \frac{|\langle \Phi_i \alpha, \Phi_j \beta \rangle|}{\|\alpha\|_2} \\
&= \max_{\beta \in \mathcal{B}_j} \frac{|\langle \alpha, \Phi_i^T \Phi_j \beta \rangle|}{\|\alpha\|_2} \\
&= \max_{\beta \in \mathcal{B}_j} \frac{|\langle \alpha, U \Sigma V^T \beta \rangle|}{\|\alpha\|_2} \\
&= \max_{\beta \in \mathcal{B}_j} \frac{|\langle U^T \alpha, \Sigma V^T \beta \rangle|}{\|\alpha\|_2} \\
&\leq \max_{\beta \in \mathcal{B}_j} \frac{\|U^T \alpha\|_\infty}{\|\alpha\|_2} \|\Sigma V^T \beta\|_1, \qquad (13)
\end{aligned}$$

where the last step comes from an application of Holder's inequality, that is, $|\langle w, z \rangle| < \|w\|_\infty \|z\|_1$.

Now, we tackle the final term in (13), which we can write as

$$\max_{\beta \in \mathcal{B}_j} \|\Sigma V^T \beta\|_1 = \max_{y \in \mathcal{Y}_j} \|\Sigma V^T \Phi_j^T y\|_1 = \max_{y \in \mathcal{Y}_j} \|\Sigma (\Phi_j V)^T y\|_1,$$

where the matrix $\Phi_j V$ contains the principal vectors in subspace $\mathcal{S}_j$. Thus, this term is simply a sum of weighted inner products between the principal vectors $\Phi_j V$ and all of the points in $\mathcal{S}_j$, where $\Sigma$ contains the cross-spectra in its diagonal entries.

Since we have assumed that the union is bounded, this implies that the inner product between the first $q$ principal vectors and the points in $\mathcal{Y}_j$ are bounded by $\gamma$, where $q = \|\sigma_{ij}\|_0 = \operatorname{rank}(G)$. Let $\Phi_j V_q \in \mathbb{R}^{n \times q}$ be the first $q$ singular vectors of $G$ corresponding to the nonzero singular values in $\Sigma$ and let $\Sigma_q \in \mathbb{R}^{q \times q}$ be a diagonal matrix with the first $q$ nonzero singular values of $G$ along its diagonal. It follows that $\|\Sigma (\Phi_j V)^T y\|_\infty = \|\Sigma_q (\Phi_j V_q)^T y\|_\infty \leq \gamma$. Now, suppose that the bounding constant $\gamma < \sqrt{1/q}$. In this case,

$$\max_{y \in \mathcal{Y}_j} \|\Sigma (\Phi_j V)^T y\|_1 \leq \gamma \|\sigma_{ij}\|_1.$$

Note that for bounded unions of subspaces, the term on the right can be made small by requiring that the bounding constant $\gamma \ll 1$. Plugging this bound into (13), we obtain the following expression

$$\max_{y \in \mathcal{Y}_j} \frac{|\langle s, y \rangle|}{\|r\|_2} \leq \gamma \|\sigma_{ij}\|_1 \frac{\|U^T \alpha\|_\infty}{\|\alpha\|_2} = \gamma \|\sigma_{ij}\|_1 \|U\|_{2,2} = \gamma \|\sigma_{ij}\|_1,$$

where this last simplification comes from the fact that $U$ is unitary and has spectral norm equal to one. Note that this bound on the mutual coherence is informative only when $\gamma \|\sigma_{ij}\|_1 < \sigma_{\max} \leq 1$. This completes the proof. □

## Acknowledgments

Thanks to Dr. Chinmay Hegde, Dr. Ankit Patel, Mahdi Soltanolkotabi, and Dr. Christoph Studer for helpful discussions and comments on this paper. Thanks also to Dr. Arian Maleki and Dr. Joel Tropp for helpful discussions. We would like to thank the anonymous reviewers, whose comments and suggestions were invaluable. ED was supported by a NSF GRFP 0940902 and a Texas Instruments Distinguished Graduate Fellowship. ACS and RGB were partially supported by following grants: NSF CCF-1117939, CCF-0431150, CCF-0728867, CCF-0926127; DARPA N66001-11-1-4090, N66001-11-C-4092; ONR N00014-08-1-1112, N00014-10-1-0989; AFOSR FA9550-09-1-0432; ARO MURIs W911NF-07-1-0185 and W911NF-09-1-0383.

## References

M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54(11):4311–4322, 2006.

E. Arias-Castro, G. Chen, and G. Lerman. Spectral clustering based on local linear approximations. *Electron. J. Stat.*, 5(0):217–240, 2011.

R. G. Baraniuk, V. Cevher, M. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Trans. Inform. Theory*, 56(4):1982–2001, 2010.

R. Basri and D. Jacobs. Lambertian reflectance and linear subspaces. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(2):218–233, February 2003.

T. Blumensath and M. Davies. Sampling theorems for signals from the union of finite-dimensional linear subspaces. *IEEE Trans. Inform. Theory*, 55(4):1872–1882, 2009.

G. Chen and G. Lerman. Spectral curvature clustering. *Int. J. Computer Vision*, 81:317–330, 2009.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comp.*, 20(1):33–61, 1998.

G. Davis, S. Mallat, and Z. Zhang. Adaptive time-frequency decompositions. *SPIE J. Opt. Engin.*, 33(7):2183–2191, 1994.

E. L. Dyer. Endogenous sparse recovery. Master's thesis, Electrical & Computer Eng. Dept., Rice University, October 2011.

E. L. Dyer, M. Duarte, D. J. Johnson, and R. G. Baraniuk. Recovering spikes from noisy neuronal calcium signals via structured sparse approximation. *Proc. Int. Conf. on Latent Variable Analysis and Sig. Separation*, pages 604–611, 2010.

E. Elhamifar and R. Vidal. Sparse subspace clustering. In *Proc. IEEE Conf. Comp. Vis. Patt. Recog. (CVPR)*, June 2009.

E. Elhamifar and R. Vidal. Clustering disjoint subspaces via sparse representation. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP)*, pages 1926–1929, March 2010.

E. Elhamifar and R. Vidal. Sparse subspace clustering: algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Machine Intell.*, 2013.

A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Machine Intell.*, 23(6):643–660, 2001.

B. V. Gowreesunker, A. Tewfik, V. Tadipatri, J. Ashe, G. Pellize, and R. Gupta. A subspace approach to learning recurrent features from brain activity. *IEEE Trans. Neur. Sys. Reh*, 19(3):240–248, 2011.

K. Kanatani. Motion segmentation by subspace separation and model selection. In *Proc. IEEE Int. Conf. Comp. Vis. (ICCV)*, 2001.

Y. Lu and M. Do. Sampling signals from a union of subspaces. *IEEE Sig. Proc. Mag.*, 25(2):41–47, March 2008.

B. Mailhè, S. Lesage, R. Gribonval, F. Bimbot, and P. Vandergheynst. Shift-invariant dictionary learning for sparse representations: extending K-SVD. In *Proc. Europ. Sig. Processing Conf. (EUSIPCO)*, 2008.

B. Mailhè, D. Barchiesi, and M. D. Plumbley. INK-SVD: Learning incoherent dictionaries for sparse representations. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP)*, pages 3573 –3576, March 2012.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *Proc. IEEE Conf. Comp. Vis. Patt. Recog. (CVPR)*, June 2008.

A.Y. Ng, M.I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Proc. Adv. in Neural Processing Systems (NIPS)*, 2:849–856, 2002.

B. Olshausen and D. Field. Sparse coding with an overcomplete basis set: a strategy employed by V1. *Vision Res.*, 37:3311–3325, 1997.

R. Ramamoorthi. Analytic PCA construction for theoretical analysis of lighting variability in images of a lambertian object. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(10):1322–1333, 2002.

I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *Proc. IEEE Conf. Comp. Vis. Patt. Recog. (CVPR)*, pages 3501–3508, June 2010.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(8):888–905, August 2000.

M. Soltanolkotabi and E. J. Candès. A geometric analysis of subspace clustering with outliers. *Annals of Statistics*, 40(4):2195–2238, 2012.

M. Soltanolkotabi, E. Elhamifar, and E. J. Candès. Robust subspace clustering. *CoRR*, abs/1301.2603, 2013.

J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50(10):2231–2242, 2004.

J. A. Tropp. Just relax: convex programming methods for identifying sparse signals in noise. *IEEE Trans. Inform. Theory*, 52(3):1030 –1051, March 2006.

R. Vidal. Subspace clustering. *IEEE Sig. Proc. Mag.*, 28(2):52–68, 2011.

R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *IEEE Trans. Pattern Anal. Machine Intell.*, 27(12):1945–1959, 2005.

Y. Wang and H. Xu. Noisy sparse subspace clustering. *Proc. Int. Conf. Machine Learning*, 2013.

J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Proc. European Conf. Comp. Vision (ECCV)*, 2006.

T. Zhang, A. Szlam, Y. Wang, and G. Lerman. Hybrid linear modeling via local best-fit flats. *Int. J. Computer Vision*, 100(3):217–240, 2012.