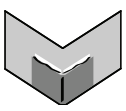
The Journal of Machine Learning Research
Volume 12
Print-Archive Edition

Pages 1–1148



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

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

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

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

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

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

Collection copyright © 2011 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
Lawrence Saul, University of California, San Diego

Managing Editor
Aron Culotta, Southeastern Louisiana University

Production Editor
Rich Maclin, University of Minnesota, Duluth

JMLR Action Editors
Francis Bach, INRIA, France **Mikhail Belkin**, Ohio State University, USA **Yoshua Bengio**, Université de Montréal, Canada **David Blei**, Princeton University, USA **Léon Bottou**, NEC Research Institute, USA Germany **Nicolò Cesa-Bianchi**, Università degli Studi di Milano, Italy **David Maxwell Chickering**, Microsoft Research, USA **William W. Cohen**, Carnegie-Mellon University, USA **Michael Collins**, Massachusetts Institute of Technology, USA **Corinna Cortes**, Google, Inc., USA **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 **Luc De Raedt**, Katholieke Universiteit Leuven, Belgium **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 **Russ Greiner**, University of Alberta, Canada **Isabelle Guyon**, ClopiNet, USA **Aapo Hyvärinen**, University of Helsinki, Finland **Tommi Jaakkola**, Massachusetts Institute of Technology, USA **Tony Jebara**, Columbia University, USA **Sathiya Keerthi**, Yahoo! Research, USA **Daphne Koller**, Stanford University, USA **Athanasios Kottas**, University of California, Santa Cruz, USA **John Lafferty**, Carnegie Mellon University, USA **Gert Lanckriet**, University of California, San Diego, USA **Neil Lawrence**, University of Manchester, UK **Daniel Lee**, University of Pennsylvania, USA **Gábor Lugosi**, Pompeu Fabra University, Spain **Sridhar Mahadevan**, University of Massachusetts, Amherst, USA **Shie Mannor**, McGill University, Canada and Technion, Israel **Chris Meek**, Microsoft Research, USA **Marina Meila**, University of Washington, USA **Mehryar Mohri**, New York University, USA **Manfred Opper**, Technical University of Berlin, Germany **Una-May O'Reilly**, Massachusetts Institute of Technology, USA **Ronald Parr**, Duke University, USA **Joelle Pineau**, McGill University, Canada **Saharon Rosset**, IBM TJ Watson Research Center, USA **John Shawe-Taylor**, Southampton University, UK **Xiaotong Shen**, University of Minnesota, USA **Yoram Singer**, Google, Inc., USA **Peter Spirtes**, Carnegie Mellon University, USA **Ingo Steinwart**, Los Alamos National Laboratory, USA **Ben Taskar**, University of Pennsylvania, USA **Lyle Ungar**, University of Pennsylvania, USA **Nicolas Vayatis**, Ecole Normale Supérieure de Cachan, France **Ulrike von Luxburg**, MPI for Biological Cybernetics, Germany **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 **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 **Cheng Soon Ong**, MPI for Biological Cybernetics, Germany **Sören Sonnenburg**, Fraunhofer FIRST, Germany

JMLR Editorial Board
Naoki Abe, IBM TJ Watson Research Center, USA **Yasemin Altun**, MPI for Biological Cybernetics, Germany **Jean-Yves Audibert**, CERTIS, France **Jonathan Baxter**, Panscient Pty Ltd, Australia **Richard K. Belew**, University of California at San Diego, USA **Samy Bengio**, Google,

Inc., USA **Kristin Bennett**, Rensselaer Polytechnic Institute, USA **Christopher M. Bishop**, Microsoft Research, UK **Lashon Booker**, The Mitre Corporation, USA **Henrik Boström**, Stockholm University/KTH, Sweden **Craig Boutilier**, University of Toronto, Canada **Koby Crammer**, University of Pennsylvania, USA **Nello Cristianini**, UC Davis, USA **Dennis DeCoste**, Facebook, USA **Thomas Dietterich**, Oregon State University, USA **Jennifer Dy**, Northeastern University, USA **Saso Dzeroski**, Jozef Stefan Institute, Slovenia **Douglas Fisher**, Vanderbilt University, USA **Peter Flach**, Bristol University, UK **Dan Geiger**, The Technion, Israel **Claudio Gentile**, Università dell'Insubria, Italy **Amir Globerson**, The Hebrew University of Jerusalem, Israel **Sally Goldman**, Washington University, St. Louis, USA **Arthur Gretton**, University College London, UK **Tom Griffiths**, University of California at Berkeley, USA **Carlos Guestrin**, Carnegie Mellon University, USA **David Heckerman**, Microsoft Research, USA **Katherine Heller**, University of Cambridge, UK **Larry Hunter**, University of Colorado, USA **Risi Kondor**, University College London, UK **Erik Learned-Miller**, University of Massachusetts, Amherst, USA **Jure Leskovec**, Stanford University, USA **Fei Fei Li**, Stanford University, USA **Yi Lin**, University of Wisconsin, USA **Wei-Yin Loh**, University of Wisconsin, USA **Yishay Mansour**, Tel-Aviv University, Israel **Jon McAuliffe**, University of Pennsylvania, USA **Andrew McCallum**, University of Massachusetts, Amherst, USA **Tom Mitchell**, Carnegie Mellon University, USA **Raymond J. Mooney**, University of Texas, Austin, USA **Klaus-Robert Muller**, Technical University of Berlin, Germany **Guillaume Obozinski**, INRIA, France **Pascal Poupart**, University of Waterloo, Canada **Ben Recht**, California Institute of Technology, USA **Cynthia Rudin**, Massachusetts Institute of Technology, USA **Robert Schapire**, Princeton University, USA **Fei Sha**, University of Southern California, USA **Shai Shalev-Shwartz**, Toyota Technology Institute, USA **Padhraic Smyth**, University of California, Irvine, USA **Nathan Srebro**, Toyota Technology Institute, USA **Alexander Statnikov**, New York University, USA **Richard Sutton**, University of Alberta, Canada **Csaba Szepesvari**, University of Alberta, Canada **Yee Whye Teh**, University College London, UK **Jean-Philippe Vert**, Mines ParisTech, France **Chris Watkins**, Royal Holloway, University of London, UK **Kilian Weinberger**, Yahoo! Research, USA **Max Welling**, University of California at Irvine, USA **Chris Williams**, University of Edinburgh, UK

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**, University of Southern California, USA **Thomas 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 **Bernhard Schölkopf**, Max-Planck-Institut für Biologische Kybernetik, Germany **Terrence Sejnowski**, Salk Institute for Biological Studies, USA **Richard Sutton**, University of Alberta, Canada **Leslie Valiant**, Harvard University, USA **Stefan Wrobel**, Fraunhofer IAIS and University of Bonn, Germany

JMLR Web Master

Youngmin Cho, University of California, San Diego

Journal of Machine Learning Research

Volume 12, 2011

1 **Exploitation of Machine Learning Techniques in Modelling Phrase Movements for Machine Translation**
Yizhao Ni, Craig Saunders, Sandor Szedmak, Mahesan Niranjan

31 **Improved Moves for Truncated Convex Models**
M. Pawan Kumar, Olga Veksler, Philip H.S. Torr

69 **CARP: Software for Fishing Out Good Clustering Algorithms**
Volodymyr Melnykov, Ranjan Maitra

75 **Multitask Sparsity via Maximum Entropy Discrimination**
Tony Jebara

111 **Bayesian Generalized Kernel Mixed Models**
Zhihua Zhang, Guang Dai, Michael I. Jordan

141 **Training SVMs Without Offset**
Ingo Steinwart, Don Hush, Clint Scovel

203 **Logistic Stick-Breaking Process**
Lu Ren, Lan Du, Lawrence Carin, David Dunson

241 **Online Learning in Case of Unbounded Losses Using Follow the Perturbed Leader Algorithm**
Vladimir V. V'yugin

267 **A Bayesian Approximation Method for Online Ranking**
Ruby C. Weng, Chih-Jen Lin

301 **Cumulative Distribution Networks and the Derivative-sum-product Algorithm: Models and Inference for Cumulative Distribution Functions on Graphs**
Jim C. Huang, Brendan J. Frey

349 **Models of Cooperative Teaching and Learning**
Sandra Zilles, Steffen Lange, Robert Holte, Martin Zinkevich

385 **Operator Norm Convergence of Spectral Clustering on Level Sets**
Bruno Pelletier, Pierre Pudlo

417 **Approximate Marginals in Latent Gaussian Models**
Botond Cseke, Tom Heskes

455 **Posterior Sparsity in Unsupervised Dependency Parsing**
Jennifer Gillenwater, Kuzman Ganchev, João Graça, Fernando Pereira, Ben Taskar

491 **Learning Multi-modal Similarity**
Brian McFee, Gert Lanckriet

525 **Minimum Description Length Penalization for Group and Multi-Task Sparse Learning**
Paramveer S. Dhillon, Dean Foster, Lyle H. Ungar

565 **Variable Sparsity Kernel Learning**
Jonathan Aflalo, Aharon Ben-Tal, Chiranjib Bhattacharyya, Jagarlapudi Saketha Nath, Sankaran Raman

593 **Regression on Fixed-Rank Positive Semidefinite Matrices: A Riemannian Approach**
Gilles Meyer, Silvère Bonnabel, Rodolphe Sepulchre

627 **Parameter Screening and Optimisation for ILP using Designed Experiments**
Ashwin Srinivasan, Ganesh Ramakrishnan

663 **Efficient Structure Learning of Bayesian Networks using Constraints**
Cassio P. de Campos, Qiang Ji

691 **Inverse Reinforcement Learning in Partially Observable Environments**
Jaedeug Choi, Kee-Eung Kim

731 **Information, Divergence and Risk for Binary Experiments**
Mark D. Reid, Robert C. Williamson

819 **Learning Transformation Models for Ranking and Survival Analysis**
Vanya Van Belle, Kristiaan Pelckmans, Johan A. K. Suykens, Sabine Van Huffel

863 **Sparse Linear Identifiable Multivariate Modeling**
Ricardo Henao, Ole Winther

907 **Forest Density Estimation**
Han Liu, Min Xu, Haijie Gu, Anupam Gupta, John Lafferty, Larry Wasserman

953 **l_p-Norm Multiple Kernel Learning**
Marius Kloft, Ulf Brefeld, Sören Sonnenburg, Alexander Zien

999 **Unsupervised Similarity-Based Risk Stratification for Cardiovascular Events Using Long-Term Time-Series Data**
Zeeshan Syed, John Guttag

1025 **Two Distributed-State Models For Generating High-Dimensional Time Series**
Graham W. Taylor, Geoffrey E. Hinton, Sam T. Roweis

1069 **Differentially Private Empirical Risk Minimization**
Kamalika Chaudhuri, Claire Monteleoni, Anand D. Sarwate

1111 **Anechoic Blind Source Separation Using Wigner Marginals**
Lars Omlor, Martin A. Giese

1149 **Laplacian Support Vector Machines Trained in the Primal**
Stefano Melacci, Mikhail Belkin

1185 **The Indian Buffet Process: An Introduction and Review**
Thomas L. Griffiths, Zoubin Ghahramani

1225 **DirectLiNGAM: A Direct Method for Learning a Linear Non-Gaussian Structural Equation Model**
Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvärinen, Yoshinobu Kawahara, Takashi Washio, Patrik O. Hoyer, Kenneth Bollen

1249 **Locally Defined Principal Curves and Surfaces**
Umut Ozertem, Deniz Erdogmus

1287 **Better Algorithms for Benign Bandits**
Elad Hazan, Satyen Kale

1313 **A Family of Simple Non-Parametric Kernel Learning Algorithms**
Jinfeng Zhuang, Ivor W. Tsang, Steven C.H. Hoi

1349 **Faster Algorithms for Max-Product Message-Passing**
Julian J. McAuley, Tibério S. Caetano

1389 **Clustering Algorithms for Chains**
Antti Ukkonen

1425 **Introduction to the Special Topic on Grammar Induction, Representation of Language and Language Learning**
Dorota Głowacka, John Shawe-Taylor, Alex Clark, Colin de la Higuera, Mark Johnson

1429 **Learning a Robust Relevance Model for Search Using Kernel Methods**
Wei Wu, Jun Xu, Hang Li, Satoshi Oyama

1459 **Computationally Efficient Convolved Multiple Output Gaussian Processes**
Mauricio A. Álvarez, Neil D. Lawrence

1501 **Learning from Partial Labels**
Timothee Cour, Ben Sapp, Ben Taskar

1537 **Super-Linear Convergence of Dual Augmented Lagrangian Algorithm for Sparsity Regularized Estimation**
Ryota Tomioka, Taiji Suzuki, Masashi Sugiyama

1587 **Double Updating Online Learning**
Peilin Zhao, Steven C.H. Hoi, Rong Jin

1617 **Learning High-Dimensional Markov Forest Distributions: Analysis of Error Rates**
Vincent Y.F. Tan, Animashree Anandkumar, Alan S. Willsky

1655 **X-Armed Bandits**
Sébastien Bubeck, Rémi Munos, Gilles Stoltz, Csaba Szepesvári

1697 **Domain Decomposition Approach for Fast Gaussian Process Regression of Large Spatial Data Sets**
Chiwoo Park, Jianhua Z. Huang, Yu Ding

1729 **A Bayesian Approach for Learning and Planning in Partially Observable Markov Decision Processes**
Stéphane Ross, Joelle Pineau, Brahim Chaib-draa, Pierre Kreitmann

1771 **Learning Latent Tree Graphical Models**
Myung Jin Choi, Vincent Y. F. Tan, Animashree Anandkumar, Alan S. Willsky

1813 **Hyper-Sparse Optimal Aggregation**
Stéphane Gaïffas, Guillaume Lecué

1835 **A Refined Margin Analysis for Boosting Algorithms via Equilibrium Margin**
Liwei Wang, Masashi Sugiyama, Zhaoxiang Jing, Cheng Yang, Zhi-Hua Zhou, Jufu Feng

1865 **Stochastic Methods for l_1-regularized Loss Minimization**
Shai Shalev-Shwartz, Ambuj Tewari

1893 **Internal Regret with Partial Monitoring: Calibration-Based Optimal Algorithms**
Vianney Perchet

1923 **Dirichlet Process Mixtures of Generalized Linear Models**
Lauren A. Hannah, David M. Blei, Warren B. Powell

1955 **Kernel Regression in the Presence of Correlated Errors**
Kris De Brabanter, Jos De Brabanter, Johan A.K. Suykens, Bart De Moor

1977 **Generalized TD Learning**
Tsuyoshi Ueno, Shin-ichi Maeda, Motoaki Kawanabe, Shin Ishii

2021 **The arules R-Package Ecosystem: Analyzing Interesting Patterns from Large Transaction Data Sets**
Michael Hahsler, Sudheer Chelluboina, Kurt Hornik, Christian Buchta

2027 **A Cure for Variance Inflation in High Dimensional Kernel Principal Component Analysis**
Trine Julie Abrahamsen, Lars Kai Hansen

2045 **Exploiting Best-Match Equations for Efficient Reinforcement Learning**
Harm van Seijen, Shimon Whiteson, Hado van Hasselt, Marco Wiering

2095 **Information Rates of Nonparametric Gaussian Process Methods**
Aad van der Vaart, Harry van Zanten

2121 **Adaptive Subgradient Methods for Online Learning and Stochastic Optimization**
John Duchi, Elad Hazan, Yoram Singer

2161 **On the Relation between Realizable and Nonrealizable Cases of the Sequence Prediction Problem**
Daniil Ryabko

2181 **Discriminative Learning of Bayesian Networks via Factorized Conditional Log-Likelihood**
Alexandra M. Carvalho, Teemu Roos, Arlindo L. Oliveira, Petri Myllymäki

2211 **Multiple Kernel Learning Algorithms**
Mehmet Gönen, Ethem Alpaydın

2269 **Smoothness, Disagreement Coefficient, and the Label Complexity of Agnostic Active Learning**
Liwei Wang

2293 **MSVMpack: A Multi-Class Support Vector Machine Package**
Fabien Lauer, Yann Guermeur

2297 **Proximal Methods for Hierarchical Sparse Coding**
Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, Francis Bach

2335 **Producing Power-Law Distributions and Damping Word Frequencies with Two-Stage Language Models**
Sharon Goldwater, Thomas L. Griffiths, Mark Johnson

2383 **Waffles: A Machine Learning Toolkit**
Michael Gashler

2389 **Universality, Characteristic Kernels and RKHS Embedding of Measures**
Bharath K. Sriperumbudur, Kenji Fukumizu, Gert R.G. Lanckriet

2411 **MULAN: A Java Library for Multi-Label Learning**
Grigorios Tsoumakas, Eleftherios Spyromitros-Xioufis, Jozef Vilcek, Ioannis Vlahavas

2415 **Union Support Recovery in Multi-task Learning**
Mladen Kolar, John Lafferty, Larry Wasserman

2437 **Parallel Algorithm for Learning Optimal Bayesian Network Structure**
Yoshinori Tamada, Seiya Imoto, Satoru Miyano

2461 **Distance Dependent Chinese Restaurant Processes**
David M. Blei, Peter I. Frazier

2489 **LPmade: Link Prediction Made Easy**
Ryan N. Lichtenwalter, Nitesh V. Chawla

2493 **Natural Language Processing (Almost) from Scratch**
Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, Pavel Kuksa

2539 **Weisfeiler-Lehman Graph Kernels**
Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, Karsten M. Borgwardt

2563 **Kernel Analysis of Deep Networks**
Grégoire Montavon, Mikio L. Braun, Klaus-Robert Müller

2583 **Theoretical Analysis of Bayesian Matrix Factorization**
Shinichi Nakajima, Masashi Sugiyama

2649 **Bayesian Co-Training**
Shipeng Yu, Balaji Krishnapuram, Rómer Rosales, R. Bharat Rao

2681 **Convex and Network Flow Optimization for Structured Sparsity**
Julien Mairal, Rodolphe Jenatton, Guillaume Obozinski, Francis Bach

2721 **Large Margin Hierarchical Classification with Mutually Exclusive Class Membership**
Huixin Wang, Xiaotong Shen, Wei Pan

2749 **Non-Parametric Estimation of Topic Hierarchies from Texts with Hierarchical Dirichlet Processes**
Elias Zavitsanos, Georgios Paliouras, George A. Vouros

2777 **Structured Variable Selection with Sparsity-Inducing Norms**
Rodolphe Jenatton, Jean-Yves Audibert, Francis Bach

2825 **Scikit-learn: Machine Learning in Python**
Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay

2831 **Neyman-Pearson Classification, Convexity and Stochastic Constraints**
Philippe Rigollet, Xin Tong

2857 **Efficient Learning with Partially Observed Attributes**
Nicolò Cesa-Bianchi, Shai Shalev-Shwartz, Ohad Shamir

2879 **Convergence Rates of Efficient Global Optimization Algorithms**
Adam D. Bull

2905 **On Equivalence Relationships Between Classification and Ranking Algorithms**
Şeyda Ertekin, Cynthia Rudin

2931 **Hierarchical Knowledge Gradient for Sequential Sampling**
Martijn R.K. Mes, Warren B. Powell, Peter I. Frazier

2975 **High-dimensional Covariance Estimation Based On Gaussian Graphical Models**
Shuheng Zhou, Philipp Rütimann, Min Xu, Peter Bühlmann

3027 **Robust Approximate Bilinear Programming for Value Function Approximation**
Marek Petrik, Shlomo Zilberstein

3065 **The Stationary Subspace Analysis Toolbox**
Jan Saputra Müller, Paul von Bünau, Frank C. Meinecke, Franz J. Király, Klaus-Robert Müller

3071 **In All Likelihood, Deep Belief Is Not Enough**
Lucas Theis, Sebastian Gerwinn, Fabian Sinz, Matthias Bethge

3097 **Efficient and Effective Visual Codebook Generation Using Additive Kernels**
Jianxin Wu, Wei-Chian Tan, James M. Rehg

3119 **Unsupervised Supervised Learning II: Margin-Based Classification Without Labels**
Krishnakumar Balasubramanian, Pinar Donmez, Guy Lebanon

3147 **Adaptive Exact Inference in Graphical Models**
¨Ozgür Sümer, Umut A. Acar, Alexander T. Ihler, Ramgopal R. Mettu

3187 **Group Lasso Estimation of High-dimensional Covariance Matrices**
Jérémie Bigot, Rolando J. Biscay, Jean-Michel Loubes, Lillian Muñiz-Alvarez

3227 **Robust Gaussian Process Regression with a Student-t Likelihood**
Pasi Jylänki, Jarno Vanhatalo, Aki Vehtari

3259 **The Sample Complexity of Dictionary Learning**
Daniel Vainsencher, Shie Mannor, Alfred M. Bruckstein

3283 **An Asymptotic Behaviour of the Marginal Likelihood for General Markov Models**
Piotr Zwiernik

3311 **Semi-Supervised Learning with Measure Propagation**
Amarnag Subramanya, Jeff Bilmes

3371 **Learning with Structured Sparsity**
Junzhou Huang, Tong Zhang, Dimitris Metaxas

3413 **A Simpler Approach to Matrix Completion**
Benjamin Recht

3431 **Convergence of Distributed Asynchronous Learning Vector Quantization Algorithms**
Benoît Patra

Journal of Machine Learning Research 12 (2011) 1-30 Submitted 5/10; Revised 11/10; Published 1/11

Exploitation of Machine Learning Techniques in Modelling Phrase Movements for Machine Translation

Yizhao Ni ENXYN@BRISTOL.AC.UK
Pattern Analysis and Intelligent Systems Group
Department of Engineering Mathematics, University of Bristol
Bristol, BS8 1UB, United Kindom

Craig Saunders CRAIG.SAUNDERS@XRCE.XEROX.COM
Xerox Research Centre Europe
6 Chemin de Maupertuis
38240 Meylan, France

Sandor Szedmak SS03V@ECS.SOTON.AC.UK
Mahesan Niranjan MN@ECS.SOTON.AC.UK
ISIS Group, School of Electronics and Computer Science
University of Southampton, Southampton
SO17 1BJ, United Kingdom

Editor: John Shawe-Taylor

Abstract

We propose a distance phrase reordering model (DPR) for statistical machine translation (SMT), where the aim is to learn the grammatical rules and context dependent changes using a phrase reordering classification framework. We consider a variety of machine learning techniques, including state-of-the-art structured prediction methods. Techniques are compared and evaluated on a Chinese-English corpus, a language pair known for the high reordering characteristics which cannot be adequately captured with current models. In the reordering classification task, the method significantly outperforms the baseline against which it was tested, and further, when integrated as a component of the state-of-the-art machine translation system, MOSES, it achieves improvement in translation results.

Keywords: statistical machine translation (SMT), phrase reordering, lexicalized reordering (LR), maximum entropy (ME), support vector machine (SVM), maximum margin regression (MMR) , max-margin structure learning (MMS)

1. Introduction

Machine translation (MT) is a challenging problem in artificial intelligence. Natural languages are characterised by large variabilities of expressions, exceptions to grammatical rules and context dependent changes. Differences in these across different languages make automatic translation a very difficult task. While early work in machine translation was dominated by rule based approaches (Bennett and Slocum, 1985), the availability of large corpora, and the ease with which they can be processed on computers has, similar to developments in other areas of artificial intelligence, paved the way for statistical methods to be applied. The process of translation from a *source* language to a *target* language is considered equivalent to a problem of retrieving a target message from the

©2011 Yizhao Ni, Craig Saunders, Sandor Szedmak and Mahesan Niranjan.

Symbol	Notation
$\mathbf{f}$	the source sentence (string)
$\mathbf{e}$	the target sentence (string)
f_j	the j-th word in the source sentence
e_i	the i-th word in the target sentence
$\mathbf{f}^I$	the source phrase sequence
$\mathbf{e}^I$	the target phrase sequence
f_j	the source phrase where f denotes the sequence of words $[f_{j_l}, \ldots, f_{j_r}]$
	and j denotes that f_j is the j-th phrase in $\mathbf{f}^I$
e_i	the target phrase where e denotes the sequence of words $[e_{i_l}, \ldots, e_{i_r}]$
	and i denotes that e_i is the i-th phrase in $\mathbf{e}^I$
Υ	the set of phrase pairs $(f_j, e_i) \in \Upsilon$
N	the number of examples in Υ
(f_j^n, e_i^n)	the n-th example in Υ that is also abbreviated as (f^n, e^n)
$\phi(f_j, e_i)$	the feature vector of phrase pair (f_j, e_i)
d	the phrase reordering distance
o	the phrase orientation class
O	the set of phrase orientations $o \in O$
C_O	the number of phrase orientations in O
φ	embedding function to map the orientation set to an output space $\varphi : O \to \mathbb{R}$
$\mathbf{w}_o$	weight vector measuring features' contribution to an orientation o
$\{\mathbf{w}_o\}_{o \in O}$	The set of weight vectors for the phrase reordering model
dim	the dimension of

Table 1: Notation used in this paper.

"source code" (Weaver, 1949). This view enables a probabilistic formulation in which the task becomes the maximisation of the posterior probability over all the phrase sequences in the target language. Principled approaches to designing the different components of such a system, shown in Figure 1, have been developed in recent years (Koehn et al., 2005).

Phrase-based *statistical machine translation* (SMT) is a task where each source sentence $\mathbf{f}$ is segmented into a sequence of I phrases $\mathbf{f}^I$ and translated into a target sequence $\mathbf{e}^I$, often by means of a stochastic process that maximises the posterior probability $\mathbf{e}^I = \arg\max_{\hat{\mathbf{e}}^I \in E} \left\{ P(\hat{\mathbf{e}}^I | \mathbf{f}^I) \right\}$. Usually the posterior probability $P(\hat{\mathbf{e}}^I | \mathbf{f}^I)$ is modelled in a log-linear maximum entropy framework (Berger et al., 1996) which permits easy integration of additional models, and is given by

$$P(\hat{\mathbf{e}}^I | \mathbf{f}^I) = \frac{\exp\left(\sum_m \lambda_m h_m(\mathbf{f}^I, \hat{\mathbf{e}}^I) \right)}{\sum_{I', \hat{\mathbf{e}}^{I'}} \exp\left(\sum_m \lambda_m h_m(\mathbf{f}^{I'}, \hat{\mathbf{e}}^{I'}) \right)},$$

where $\{h_m\}$ represent sub-models with scaling factors $\{\lambda_m\}$. As the denominator only depends on the source phrase sequence $\mathbf{f}^I$, it is usually discarded and the solution is also represented as $\mathbf{e}^I = \arg\max_{\hat{\mathbf{e}}^I \in E} \left\{ \exp\left(\sum_m \lambda_m h_m(\mathbf{f}^I, \hat{\mathbf{e}}^I) \right) \right\}$.

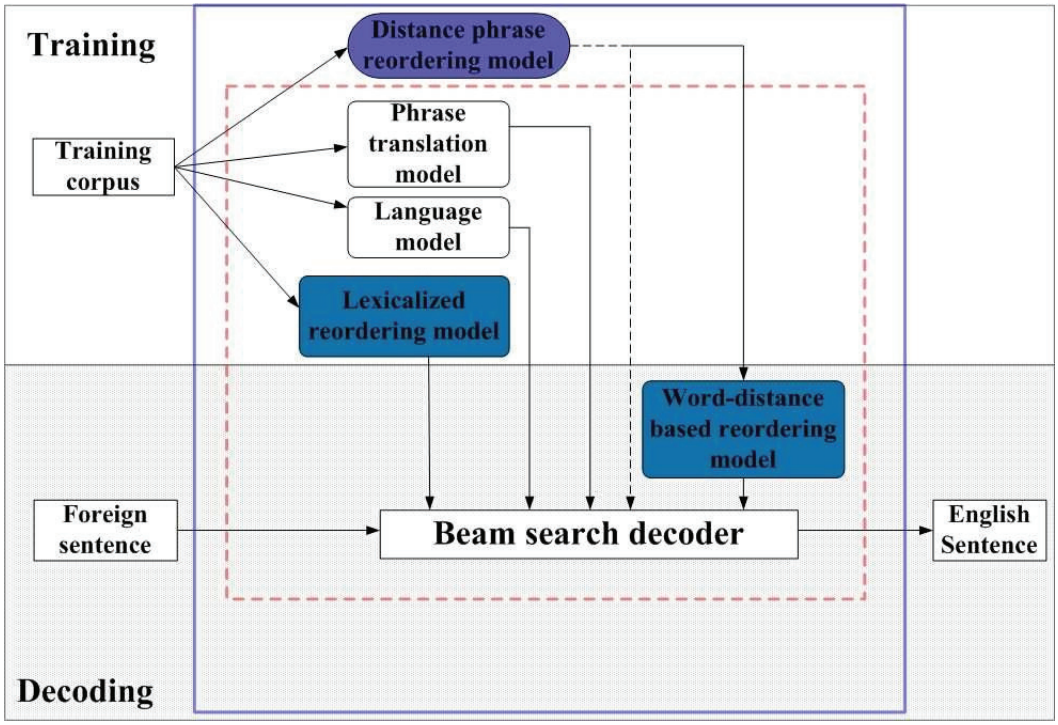


Figure 1: Training (top box) and decoding (bottom box) procedures for a state-of-the-art SMT systems (dotted line box) and our MT system (solid line box).

A combination of several sub-models $\{h_m\}$ (see Figure 1), including a phrase translation probability model, a language model and a phrase reordering model are commonly used. Each sub-model is trained individually and then weighted by a scale factor λ_m tuned to achieve good final translation quality (Och, 2003). Finally, the decoder searches a Viterbi-best string path given the joint decoding information. The reader is referred to Ni (2010) for detailed discussions on these models.

1.1 Modelling Phrase Movements

In this paper, we focus on developing a crucial component in statistical machine translation—the *phrase reordering model*. Word or phrase reordering is a common problem in bilingual translations arising from different grammatical structures. For example, the Chinese "NP_1 DEG NP_2" sequence is analogous to the English possessive structure of "NP_1's NP_2" and does not require reordering (see Figure 2 (a)). However, due to different linguistic environment it may come from, this Chinese possessive structure can express more sophisticated relationships which are inappropriate for the "NP_1's NP_2" expression, for example, the "NP_2 of NP_1" sequence which requires phrase swapping (see Figure 2 (b)). In general, if the decoder "knows" the orders of phrase translations in the target

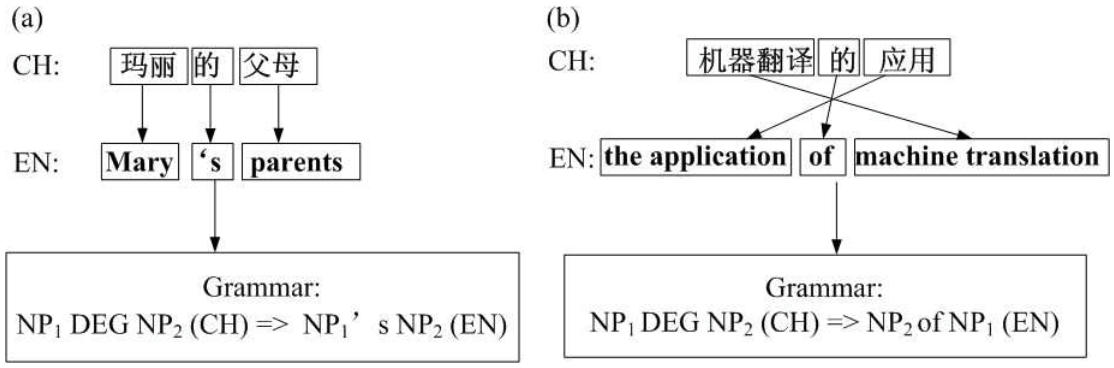


Figure 2: Example: the distance phrase reordering in Chinese-to-English bilingual translation.

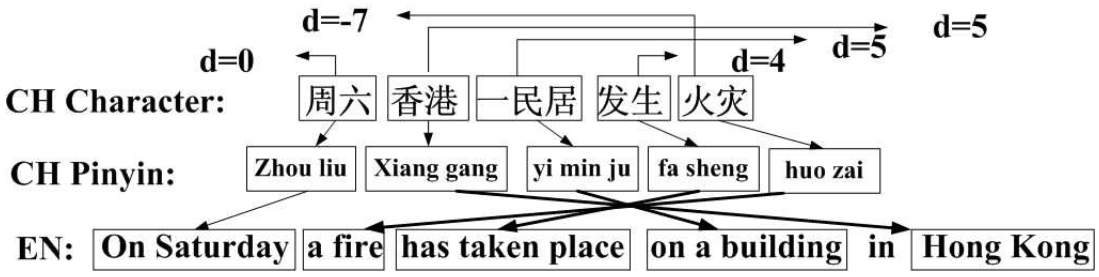


Figure 3: The phrase reordering distance d.

language, the fluency of machine translation can be greatly improved. This motivates investigations into, and development of models for, phrase reordering.

Now taking a Chinese-to-English translation (see Figure 3) for example, obviously not all words are translated one by one and some words are translated far behind after its preceding words are translated (e.g., phrase "a fire"). Therefore, an ideal phrase reordering model should be able to handle arbitrary distance phrase movements. However, handling such movements is a computationally expensive problem (Knight, 1999). Within recently developed SMT systems, a simple phrase reordering model, named *word distance-based reordering model* (WDR), is commonly used (Och et al., 1999; Koehn, 2004; Zens et al., 2005). This model defines a reordering distance for the j-th source phrase f_j as (see Figure 3 for an illustration of this.)

$$d_j := \begin{array}{l}\text{abs(last source word position of previously translated phrase } +1 \\ \quad -\text{first source word position of newly translated phrase } f_j)\end{array}, \tag{1}$$

and the total cost of phrase movements for a sentence pair $(\mathbf{f},\mathbf{e})$ is linear proportional to these reordering distances $h_d(\mathbf{f}^J,\mathbf{e}^I) = -\alpha\sum_j d_j$ with a tuning parameter α. Although computationally efficient, this model has been shown to be weak due to its content independence. A content-based extension to WDR is the *lexicalized reordering model* (LR) (Tillmann, 2004; Koehn et al., 2005), which splits the distance space into several segments, each of which represents a phrase reordering orientation o (see Figure 4). Then the phrase reordering probability for a phrase pair (f_j, e_i) is

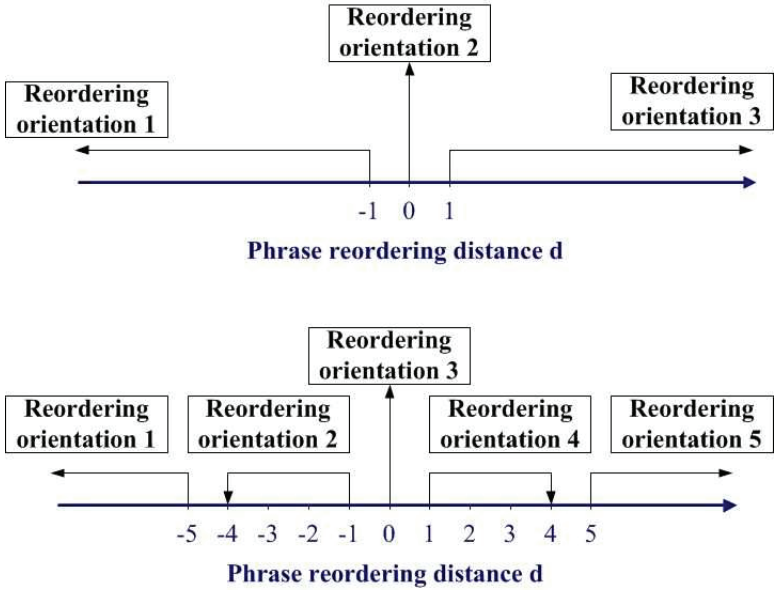


Figure 4: The phrase reordering orientations: the three-class setup (top) and the five-class setup (bottom).

predicted using *maximum likelihood estimation* (MLE)

$$p\big(o|(f_j,e_i)\big) = \frac{count(o,(f_j,e_i))}{\sum_{o'} count(o',(f_j,e_i))},$$

where $h_d(\mathbf{f}^J,\mathbf{e}^I) = \sum_{(f_j,e_i)\in(\mathbf{f}^J,\mathbf{e}^I)} p(o|(f_j,e_i))$ is used to represent the cumulative cost of phrase movements. Although the overall performance is better than WDR, it usually suffers from data sparseness, and some heuristics have to be employed to make the approach effective.

Adopting the idea of predicting phrase reordering orientations, researchers started exploiting context or grammatical content which may relate to phrase movements (Tillmann and Zhang, 2005; Xiong et al., 2006; Zens and Ney, 2006). In general, the distribution of phrase reorderings is expressed with a log-linear form

$$p(o|(f_j,e_i),\mathbf{w}_o) = \frac{h(\mathbf{w}_o^T\phi(f_j,e_i))}{Z(f_j,e_i)} \tag{2}$$

with the normalisation term $Z(f_j,e_i) = \sum_{o\in O} h(\mathbf{w}_o^T\phi(f_j,e_i))$. The feature parameters $\{\mathbf{w}_o\}_{o\in O}$ are then tuned by different discriminative models, among which the *maximum entropy* (ME) framework is a popular candidate. To characterise phrase movements, a variety of linguistic features are proposed

- Context features – word sequence (n-gram) features in (or around) the phrases. These indicator functions are the basic features used in Zens and Ney (2006) and also used in other MT experiments such as the word-sense disambiguation of Vickrey et al. (2005).

- Shallow syntactic features – part-of-speech (POS) tags or word-class features in (or around) the phrases. These indicator features are also used in the models above, and also in the context-aware phrase selection model of Giménez and Màrquez (2007).

- Statistical features – features such as the lexicalized reordering probability (Koehn et al., 2005) and the language model probability, etc. These real-value features are introduced by Tillmann and Zhang (2005) and are shown to be beneficial in capturing the local phrase reordering information.

Many other feature sets, such as lemma features and syntactic relationships in POS tags have also been investigated, posing a feature selection problem for any learning algorithm. Instead of investigating features sets, in this paper we concentrate on exploiting a limited set of linguistic features with different learning agents. We propose a *distance phrase reordering model* (DPR) that is also inspired by the orientation prediction framework (Koehn et al., 2005). Unlike Xiong et al. (2006) and Zens and Ney (2006) we regard phrase movements as a classification problem and use three multi-class learning agents—*support vector machine* (SVM), *maximum margin regression* (MMR) and *max-margin structure learning* (MMS) to perform the classification. Our goal is to find a learning agent that provides good tradeoff between classification accuracy with a limited feature set and computational efficiency. Furthermore, we also integrate the DPR model in a traditional SMT system, and the resulting MT system (solid line box in Figure 1) is compared with a state-of-the-art SMT system (dotted line box in Figure 1) on a Chinese-to-English MT task so as to demonstrate the effectiveness of the proposed DPR model.

1.2 Contribution and Structure

This paper makes two significant contributions. The first is a comparison, in terms of classification accuracy and computational efficiency, between different machine learning techniques for distance phrase movements in machine translation. This is mainly in the paradigm of structured learning, including maximum margin structure learning (MMS) and maximum margin regression (MMR), which is seen as a powerful framework that takes advantage of output structures in supervised learning problems, in modern machine learning literature. Our second contribution is the demonstration that this paradigm is effective in the task of phrase movements, which is acknowledged as a challenging task in machine translation. This turns out to be true, both in stand-alone translation tasks and when the method is integrated into a complete end-to-end statistical machine translation system. This is sufficiently encouraging that we have made our work available as a public domain software package[1] (Ni et al., 2010a) in a form that it can be integrated into the widely used MOSES system.[2]

The remainder of the paper is organised as follows: a general framework of the DPR model is given in Section 2, which specifies the modelling of phrase movements and describes the motivations of using the three learning agents. Then in Section 3 we demonstrate the linguistic features used and the training procedure for the DPR model. Section 4 evaluates the performance of the DPR model with both phrase reordering classification and machine translation experiments. Finally, we draw conclusions and mention areas for future work in Section 5.

1. The software is available at `http://patterns.enm.bris.ac.uk/distance-phrase-reordering-for-moses`.
2. MOSES is available at `http://www.statmt.org/moses/`.

2. Distance Phrase Reordering (DPR)

We adopt a discriminative model to capture the frequently occurring distance reorderings (e.g., Figure 2). An ideal model would consider every word position as a class and predict the start position of the next phrase, although in practice this is rather difficult to achieve. Hence, we consider a limited set of classes.

2.1 Orientation Class Definition

Following Koehn's lexicalized reordering model, we use the phrase reordering distance d in (1) to measure phrase movements. The distance space $d \in \mathbb{Z}$ is then split into C_O segments (i.e., C_O classes) and the possible start positions of phrases are grouped to make up a phrase orientation set O. Note that the more orientation classes a model has, the closer it is to the ideal model, but the smaller amount of training samples it would receive for each class. Therefore we consider two setups: a three-class approach $O = \{d < 0, d = 0, d > 0\}$ and one with five classes $O = \{d \leq -5, -5 < d < 0, d = 0, 0 < d < 5, d \geq 5\}$[3] (see Figure 4).

2.2 Reordering Probability Model and Learning Agents

Given a (source, target) phrase pair $(f_j^n, e_i^n) \in \Upsilon$ with $f_j = [f_{j_l}, \ldots, f_{j_r}]$ and $e_i = [e_{i_l}, \ldots, e_{i_r}]$, the *distance phrase reordering probability* has the form

$$p(o|(f_j^n, e_i^n), \{\mathbf{w}_o\}) := \frac{h\big(\mathbf{w}_o^T \phi(f_j^n, e_i^n)\big)}{\sum\limits_{o' \in O} h\big(\mathbf{w}_{o'}^T \phi(f_j^n, e_i^n)\big)}, \tag{3}$$

where $\mathbf{w}_o = [w_{o,0}, \ldots, w_{o,dim(\phi)}]^T$ is the weight vector measuring features' contribution to an orientation $o \in O$, ϕ is the feature vector and h is a pre-defined monotonic function.

Equation (3) is analogous to the well-known maximum entropy framework of Equation (2). In contrast to learning $\{\mathbf{w}_o\}_{o \in O}$ by maximising the entropy over all phrase pairs' orientations

$$\max_{\{\mathbf{w}_o \in O\}} \Big\{ - \sum_{(f_j^n, e_i^n) \in \Upsilon} \sum_{o \in O} p(o|f_j^n, e_i^n, \{\mathbf{w}_o\}) \log p(o|f_j^n, e_i^n, \{\mathbf{w}_o\}) \Big\},$$

we propose using maximum-margin based approaches to learn $\{\mathbf{w}_o\}_{o \in O}$. Under this framework, three discriminative models are introduced, for different purposes of capturing phrase movements. We now describe each of these in the following subsections.

2.2.1 SUPPORT VECTOR MACHINE (SVM) LEARNING

Support vector machines (SVMs) is a learning method which has become very popular in many application areas over recent years (see, e.g., Cristianini and Shawe-Taylor, 2000 for details). The basic SVM is a binary classifier, and we learn each $\mathbf{w}_o$ with a separated SVM that solves the following convex optimisation problem

3. The five-word parameter setting is designed specifically for the MT experiments, which enables each class to have similar sizes of samples.

$$\begin{array}{ll}\min\limits_{\mathbf{w}_o,\xi} & \frac{1}{2}\mathbf{w}_o^T\mathbf{w}_o + C\sum\limits_{(f^n,e^n)\in\Upsilon}\xi(f^n,e^n) \\ s.t. & \varphi(o_n,o)\left(\mathbf{w}_o^T\phi(f^n,e^n)\right) \geq 1-\xi(f^n,e^n) \ , \\ & \xi(f^n,e^n)\geq 0,\ \forall(f^n,e^n)\in\Upsilon\end{array}$$

where $\varphi(o_n,o)$ is an embedding function for the phrase orientation o_n, which is assigned 1 if $o_n = o$ and -1 otherwise.

This approach has been successfully used for many tasks. However, for N training examples (phrase pairs) the computation complexity of the SVM model is somewhere between $O(C_O N + N^2 dim(\phi))$ and $O(C_O N^2 + N^2 dim(\phi))$ (Bishop, 2006). The dependence on C_O may cause computational problems, especially when the number of phrase orientations increase.

2.2.2 MAXIMUM MARGIN REGRESSION (MMR) LEARNING

A good agent for learning $\{\mathbf{w}_o\}_{o\in O}$ should adapt to the number of phrase orientations C_O, enabling Equation (3) to extend to more classes in the future. In this sense, we introduce the maximum margin regression (MMR) technique, that acquires $\{\mathbf{w}_o\}_{o\in O}$ by solving the following optimisation problem (Szedmak et al., 2006)

$$\begin{array}{ll}\min\limits_{\{\mathbf{w}_o\}_{o\in O}} & \frac{1}{2}\sum\limits_{o\in O}\mathbf{w}_o^T\mathbf{w}_o + C\sum\limits_{(f^n,e^n)\in\Upsilon}\xi(f^n,e^n) \\ s.t. & \sum\limits_{o\in O}\varphi(o_n,o)\mathbf{w}_o^T\phi(f^n,e^n) \geq 1-\xi(f^n,e^n) \ , \\ & \xi(f^n,e^n)\geq 0,\ \forall(f^n,e^n)\in\Upsilon\end{array}$$

where $\varphi(o_n,o)$ is an indicator function, which is assigned 1 if the phrase reordering orientation satisfies $o_n = o$ and 0 otherwise.

The computational complexity of MMR is the complexity of a binary SVM (Szedmak et al., 2006), which is independent to the output structure (i.e., number of classes). This allows the orientation class approach presented here to be extended, say to tree structured models, whilst not increasing the computational complexity. Furthermore, it allows the use of non-linear functions, going beyond the approach presented in Zens and Ney (2006), and is expected to provide more flexibility in the expression of phrase features. The reader is referred to Appendix B for further description of MMR.

2.2.3 MAX-MARGIN STRUCTURE (MMS) LEARNING

The two techniques above only consider a fixed margin to separate one orientation class from the others. However, as the phrase reordering orientations tend to be interdependent, introducing flexible margins to separate different orientations sounds more reasonable. Take the five-class setup for example, if an example in class $d \leq -5$ is classified in class $-5 < d < 5$, intuitively the loss should be smaller than when it is classified in class $d > 5$. Therefore, learning $\{\mathbf{w}_o\}_{o\in O}$ is more than a multi-class classification problem: the output (orientation) domain has an inherent structure and the model should respect it. By this motivation, we introduce the max-margin learning framework proposed in Taskar et al. (2003) which is equivalent to minimising the sum of all classification errors

$$\min_{\{\mathbf{w}_o\}_{o\in O}} \frac{1}{N}\sum_{n=1}^{N}\rho(o_n,f^n,e^n,\{\mathbf{w}_o\}_{o\in O}) + \frac{\lambda}{2}\sum_{o\in O}\|\mathbf{w}_o\|^2, \tag{4}$$

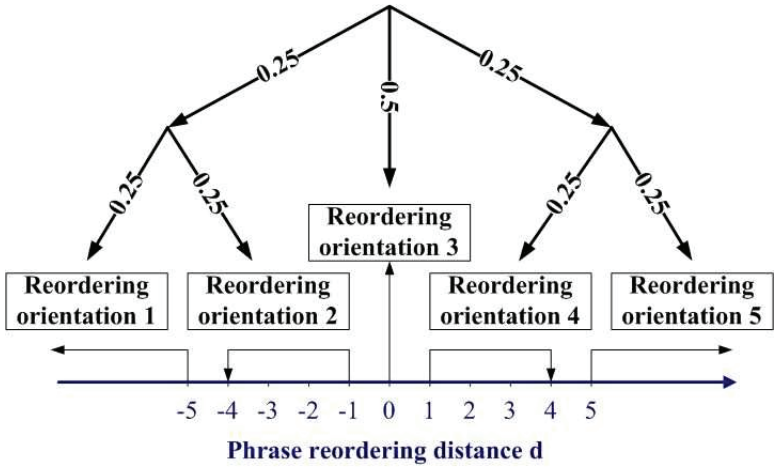


Figure 5: The tree structure constructed by the distance matrix $\triangle(o_n, o')$.

where $\lambda \geq 0$ is a regularisation parameter,

$$\rho(o_n, f^n, e^n, \{\mathbf{w}_o\}_{o\in O}) = \max\left\{0, \max_{o'\neq o_n}[\triangle(o_n, o') + \mathbf{w}_{o'}^T \phi(f^n, e^n)] - \mathbf{w}_{o_n}^T \phi(f^n, e^n)\right\} \quad (5)$$

is a structured margin loss and function $\triangle(o_n, o')$ is applied to measure the "distance" between a pseudo-orientation o' and the correct one o_n. In the experiments, the distance matrix is pre-defined as

$$\triangle(o_n, o') = \begin{cases} 0 & \text{if } o' = o_n \\ 0.5 & \text{if } o' \text{ and } o_n \text{ are close in } O \\ 1 & \text{else} \end{cases}.$$

As shown in Figure 5, this is equivalent to constructing a heuristic tree structure in the orientation domain.

Theoretically, the structured loss (5) requires that the orientation o' which is "far away" from the true orientation o_n must be classified with a large margin $\triangle(o_n, o')$, while nearby candidates are allowed to be classified with a smaller margin. This is an extension of that provided by Collins (2002) where no distance between classes is considered (i.e., $\triangle(o_n, o') = 1$, $\forall o'$), and it has been applied successfully to phrase translation tasks (Ni et al., 2010b).

Considering the training time, we ignored the regularisation term (i.e., $\lambda = 0$) and used a perceptron-based structured learning (PSL) algorithm to tune the parameters $\{\mathbf{w}_o\}_{o\in O}$, the pseudo-code is demonstrated in Table 2.

Table 2 indicates that the computational complexity of PSL is $O(Ndim(\phi)C_O)$, which still depends on the number of classes. However, compared with the previous SVM and even the MMR models, PSL is substantially faster as in practice the number of classes C_O is much smaller than the number of examples N. This time efficiency is also verified by the experiment results shown in Figure 15.

Notice that in PSL $\mathbf{w}_{o,k+1}$ is tested on the example $(o_n, \phi(f^n, e^n))$ which is not available for training $\mathbf{w}_{o,k}$, so if we can guarantee a low cumulative loss we are already guarding against over-fitting. If one wished to add regularisation to further guard against over-fitting, one could apply methods

Input of the learner: The samples $\Upsilon = \{o_n, \phi(f^n, e^n)\}_{n=1}^N$, learning rate η
Initialization: $k = 0$; $\mathbf{w}_{o,k} = \mathbf{0} \quad \forall o \in O$;
Repeat
  randomly sample $(f^n, e^n) \in \Upsilon$
  $V = \max_{o'} \{\triangle(o_n, o') + \mathbf{w}_{o',k}^T \phi(f^n, e^n)\}$
  $o^* = \arg\max_{o'} \{\triangle(o_n, o') + \mathbf{w}_{o',k}^T \phi(f^n, e^n)\}$
  if $\mathbf{w}_{o_n,k}^T \phi(f^n, e^n) < V$ **then**
    $\mathbf{w}_{o,k+1}|_{o=o_n} = \mathbf{w}_{o,k}|_{o=o_n} + \eta\phi(f^n, e^n)$
    $\mathbf{w}_{o,k+1}|_{o=o^*} = \mathbf{w}_{o,k}|_{o=o^*} - \eta\phi(f^n, e^n)$
    $k = k + 1$
until converge
Output of the learner: $\mathbf{w}_{o,k+1} \in \mathbb{R}^{dim(\phi)} \quad \forall o \in O$

Table 2: Pseudo-code of perceptron-based structured learning (PSL).

such as ALMA (Gentile, 2001) or NORMA (Kivinen et al., 2004). However, the requirement of normalising $\mathbf{w}_o$ at each step makes the implementation intractable for a large structured learning problem. As an alternative, the risk function (4) can be reformulated as a joint convex optimisation problem[4]

$$\min_{\{\|\mathbf{w}_o\| \le R\}} \max_{\{\mathbf{z}_o \in Z\}} L(\{\mathbf{w}_o\}_{o \in O}, \{\mathbf{z}_o\}_{o \in O}) \tag{6}$$

with

$$L(\{\mathbf{w}_o\}_{o \in O}, \{\mathbf{z}_o\}_{o \in O}) = \sum_{n=1}^{N} \max\left\{0, \sum_{o \in O} z_o^n \left(\triangle(o_n, o) + \mathbf{w}_o^T \phi(f^n, e^n)\right)\right\}$$
$$s.t. \quad \begin{cases} z_o^n = -1 & o = o_n \\ z_o^n \ge 0 & o \ne o_n \quad n = 1, \ldots, N \\ \sum_{o \in O} z_o^n = 0 \end{cases}.$$

This min-max problem can be solved by the *extra-gradient* algorithm, which is guaranteed to converge linearly to a solution of $\{\mathbf{w}_o^*\}_{o \in O}$ and $\{\mathbf{z}_o^*\}_{o \in O}$ under mild conditions (Taskar et al., 2006).

3. Feature Extraction and Application

In this section, we describe two key steps for the method: feature extraction and model training.

3.1 Feature Extraction

Following (Vickrey et al., 2005; Zens and Ney, 2006), we consider different kinds of information extracted from the phrase environment (see Table 3). To capture unknown grammars and syntactic structures, some of the features would depend on the word-class[5] information. Mathematically, given a sequence s from the feature environment (e.g., $s = [f_{j_l - d_l}, \ldots, f_{j_l}]$ in Figure 6), the features

4. The reader is referred to Appendix A for detailed inference.
5. The word-class tags are provided by the state-of-the-art SMT system (MOSES).

	features for source phrase f_j	features for target phrase e_i
Context features	Word n-grams within a window (length d) around the source phrase edge $[j_l]$ and $[j_r]$	Word n-grams (subphrases) of the target phrase $[e_{i_l}, \ldots, e_{i_r}]$
Syntactic features	Word-class n-grams within a window (length d) around the source phrase edge $[j_l]$ and $[j_r]$	Word-class n-grams (subphrases) of the target phrase $[e_{i_l}, \ldots, e_{i_r}]$

Table 3: Features extracted from the phrase environment. n-gram indicates a word sequence of length n.

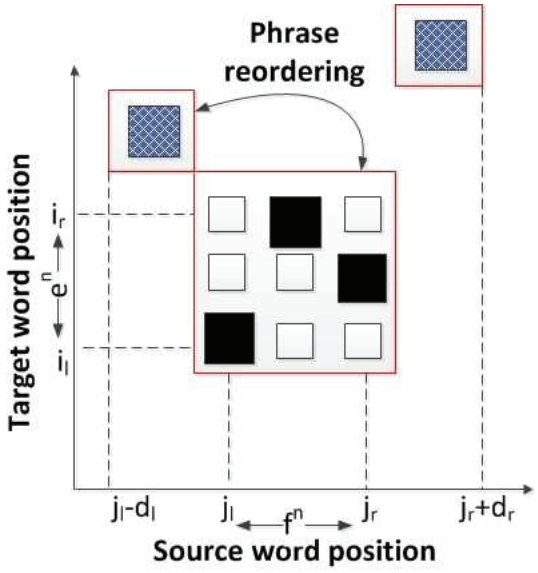


Figure 6: Illustration of the phrase pair (f_j^n, e_i^n) (the word alignments are in black rectangle). The linguistic features are extracted from the target phrase and a window environment (blue shadow boxes) around the source phrase.

extracted are of the form

$$\phi_u(s_p^{|u|}) = \delta(s_p^{|u|}, u),$$

with the indicator function $\delta(\cdot,\cdot)$, $p = \{j_l - d_l, \ldots, j_l, j_r, \ldots, j_r + d_r\}$ and string $s_p^{|u|} = [f_p, \ldots, f_{p+|u|}]$. In this way, the phrase features are distinguished by both the content u and its start position p.

This *position-dependent linguistic* feature expression creates a very high dimensional feature space where each example (f_j^n, e_i^n) is assigned a sparse feature vector. Figure 7 shows the context feature space created for all five phrase pairs in Figure 3 and the non-zero features for the phrase pair ("Xiang gang", "Hong Kong"). The whole feature space contains 180 features and only 9

features are non-zero for this phrase pair. The advantage of this feature expression is the collection of comprehensive linguistic information which may relate to phrase movements. However, the side effect it brings in is a large set of free parameters which may cause over-fitting on the training data.

3.2 Training and Application

The training samples $\{o_n, (f^n, e^n)\}_{n=1}^{N}$ (phrase pairs up to length 8) for the DPR model are derived from a general phrase pair extraction procedure described in Koehn et al. (2005). At translation time, we follow the idea of Giménez and Màrquez (2007), where the samples having the same source phrase f are considered to be from the same cluster (cf., Figure 8 (a)). A sub-model using the above learning agents is then trained for each cluster. In our largest experiment, this framework results in training approximately $70,000$ sub-DPR models (Figure 8 (b)). A statistics of the number of free parameters (features) against the number of training examples for each cluster is depicted in Figure 8 (c), implying a potential over-fitting risk. To avoid the over-fitting, a prior of $\{\mathbf{w}_o\}_{o \in O}$ is applied to the maximum entropy (ME) model as used in Zens and Ney (2006) and for the MMS model, the *early stopping* strategy[6] is used which involves the careful design of the maximum number of iterations.

During the decoding, the DPR model finds the corresponding sub-model for a source phrase f_j and generates the phrase reordering probability for each orientation class with Equation (3). In particular, for the classification experiments, the most-confident orientation is selected as the predicted class.

4. Experiments

Experiments used the *parallel texts of Hong Kong laws*.[7] This bilingual *Chinese-English* corpus consists of mainly legal and documentary texts from Hong Kong which is aligned at the sentence level. The sizes of the corpus are shown in Figure 9. As the vocabulary sizes of the corpus are very small, the content information is relatively easy to learn. However, due to many differences in word order (grammar) occurring for Chinese-English, this corpus contains many long distance phrase movements (see Figure 9). In this case, the phrase reordering model is expected to have more influence on the translation results, which makes this a suitable data set to analyse and demonstrate the effectiveness of our proposed DPR model.

For the experiments, sentences of lengths between 1 and 100 words were extracted and the ratio of source/target lengths was no more than $2:1$. The training set was taken among $\{20K, 50K, 100K, 150K, 185K\}$ sentences while the test set was fixed at $1K$ sentences.

4.1 Classification Experiments

We used *GIZA++* to produce word alignments, enabling us to compare a DPR model against a baseline LR model (Koehn et al., 2005) that uses MLE orientation prediction and a discriminative model that uses an ME framework (Zens and Ney, 2006). In addition, we also compared the clas-

6. The strategy selects the maximum number of iterations and the learning rate η by cross-validating on a validation set. In our experiments, this was done on the $185K$-sentence Chinese-to-English MT task and the (max-iteration, learning rate) with the best performance was chosen for all other MT experiments.

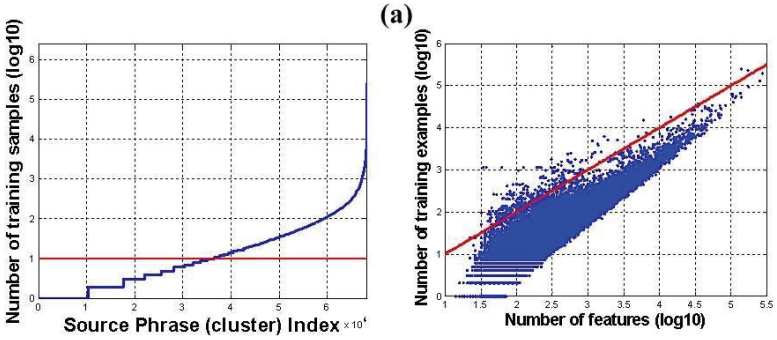
7. The original corpus is available at `http://projects.ldc.upenn.edu/Chinese/hklaws.htm`, which however contains some sentence alignment errors. The corpus has been further cleaned up and aligned at the sentence level by the authors. This refined corpus is now available upon request.

Context (source) / Position	-3	-2	-1	1	2	3
Zhou	0	**1**	0	0	0	0
liu	0	0	**1**	0	0	0
xiang	0	0	0	0	0	0
gang	0	0	0	0	0	0
yi	0	0	0	**1**	0	0
min	0	0	0	0	**1**	0
ju	0	0	0	0	0	**1**
fa	0	0	0	0	0	0
sheng	0	0	0	0	0	0
huo	0	0	0	0	0	0
zai	0	0	0	0	0	0
Zhou liu	0	**1**	0	0	0	0
liu Xiang	0	0	0	0	0	0
Xiang gang	0	0	0	0	0	0
gang yi	0	0	0	0	0	0
yi min	0	0	0	**1**	0	0
min ju	0	0	0	0	**1**	0
ju fa	0	0	0	0	0	0
fa sheng	0	0	0	0	0	0
sheng huo	0	0	0	0	0	0
huo zai	0	0	0	0	0	0
Zhou liu Xiang	0	0	0	0	0	0
liu Xiang gang	0	0	0	0	0	0
Xiang gang yi	0	0	0	0	0	0
gang yi min	0	0	0	0	0	0
yi min ju	0	0	0	**1**	0	0
min ju fa	0	0	0	0	0	0
ju fa sheng	0	0	0	0	0	0
fa sheng huo	0	0	0	0	0	0
sheng huo zai	0	0	0	0	0	0

Figure 7: An example of the linguistic feature space created for all phrases in Figure 3 and the non-zero features for the phrase pair ("Xiang gang", "Hong Kong"). Due to space limitation, this example only demonstrates the context features for the source phrases (i.e., the top left block in Table 3).

Cluster label	安全 (Chinese character) an quan (Chinese pinyin)
Training Samples N=4	an quan – safety an quan – safely an quan – safety of an quan – safe

(a)



(b) (c)

Figure 8: (a) A cluster for the source phrase "an quan" and its training samples (phrase pairs). Note that the linguistic features for the samples are not demonstrated in this example. (b) The number of training samples for each cluster (phrases are extracted from 185K Chinese-English sentence pairs). (c) The statistics of the number of features against the number of training samples (phrases are extracted from 185K Chinese-English sentence pairs).

Statistics	Chinese	English
Sentence Pairs	216,250	
Running Words	9.76M	7.00M
Vocabulary Size	8,129	23,025

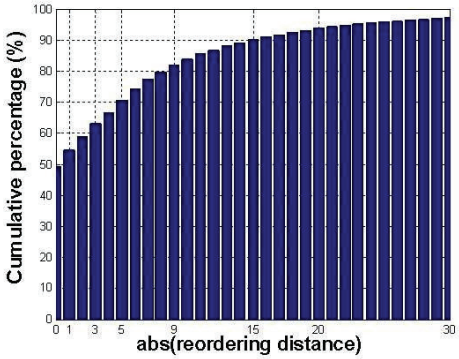


Figure 9: The data statistics for the *parallel texts of Hong Kong laws* corpus (left) and the statistics of phrase reordering distance d for all consistent phrase pairs (up to length 8) extracted from the corpus (right). The word alignments are provided by the word alignment toolkit *GIZA++*. The right figure shows that short distance phrase movements (i.e., $d < 4$) only take up 62% of the whole phrase movements.

Chinese-to-English task										
Orientations	Training set					Test set				
	20K	50K	100K	150K	185K	20K	50K	100K	150K	185K
$d<0$	0.17M	0.45M	0.82M	1.25M	1.63M	13K	16K	16K	17K	17K
$d=0$	0.41M	1.11M	2.10M	3.30M	4.04M	28K	33K	34K	38K	38K
$d>0$	0.12M	0.32M	0.61M	0.90M	1.11M	9K	10K	11K	11K	11K
$d\leq -5$	80K	0.20M	0.38M	0.56M	0.70M	6.0K	6.5K	7.3K	7.5K	7.4K
$-5<d<0$	90K	0.25M	0.44M	0.69M	0.83M	7.0K	9.5K	8.7K	9.5K	9.6K
$d=0$	4.1M	1.11M	2.10M	3.30M	4.04M	28K	33K	34K	38K	38K
$0<d<5$	40K	0.10M	0.20M	0.27M	0.31M	2.5K	2.8K	2.5K	2.4K	2.2K
$d\geq 5$	80K	0.22M	0.41M	0.63M	0.80M	6.5K	7.2K	8.5K	8.6K	8.8K

Table 4: The training and the test sizes (phrase pairs) for three-class setup (top) and five-class setup (bottom), where "K" indicates thousand and "M" indicates million.

sification performance and the computational efficiency among the three learning agents for DPR: SVM,[8] MMR and MMS, where the goal was to find the best learning agent for the MT experiments.

Two orientation classification tasks were carried out: one with three-class setup and one with five-class setup. We discarded points that had long distance reordering ($|d|>15$, representing less than 8% of the data) to avoid some alignment errors caused by *GIZA++*. This results in data sizes shown in Table 4. The classification performance was measured by an overall precision across all orientation classes and the class-specific F1 measures and the experiments were repeated three times to asses variance.

4.1.1 COMPARISON OF OVERALL PRECISIONS AND THE CLASS-SPECIFIC F1-SCORES

Figure 10 shows classification accuracies at different sizes of training sets, and we observed a monotonic increase with the amount of training data used. In addition, all discriminative models perform better than the generative LR model. The MMS approach achieves the best classification performance, with an absolute 8.5% average improvement with three-class setup and an absolute 8.7% average improvement with five classes. Similar improvements are observed when examining class-specific F1 scores on Table 5 and Table 6; the DPR model with the MMS learning agent achieves the best results. However, the DPR models with SVM and MMR techniques do not perform very well in the experiments, possibly due to the feature expression we used. Since constructing a kernel using the sparse feature expression usually results in a very sparse kernel matrix where little similarity between samples is presented, SVM and MMR might not extract adequate information for modelling phrase movements.

When the training sample size is large, the ME model performs better than all other learning agents except MMS, showing its good ability in exploiting features. But when the training sample size is small (e.g., 50K-sentence task), its results are worse than that of SVM, possibly due to the over-fitting on the training data. This reveals the importance of choosing the priors for the ME models: a simple prior may not be helpful while a complicated prior usually makes the training

8. The multi-class SVM model is trained by SVM-Multiclass (Tsochantaridis et al., 2004).

Orientations	Training Data	Generative model	Discriminative models			
		LR	MMR	SVM	ME	MMS
$d<0$	20K	57.2 ± 0.8	63.7 ± 0.6	64.1 ± 0.9	63.9 ± 0.5	$\mathbf{64.7}\pm0.6$
	50K	58.5 ± 0.1	65.6 ± 0.6	65.8 ± 0.7	65.9 ± 0.5	$\mathbf{67.4}\pm0.1$
	100K	61.6 ± 1.1	69.6 ± 1.4	70.6 ± 1.3	71.8 ± 1.3	$\mathbf{74.2}\pm0.3$
	150K	63.8 ± 0.6	72.3 ± 0.8	73.0 ± 0.6	75.3 ± 1.3	$\mathbf{76.5}\pm1.0$
	185K	63.3 ± 0.8	72.2 ± 1.2	73.1 ± 0.8	75.7 ± 1.0	$\mathbf{76.8}\pm1.0$
$d=0$	20K	80.1 ± 0.3	83.6 ± 0.1	84.3 ± 0.2	83.7 ± 0.2	$\mathbf{84.7}\pm0.2$
	50K	80.0 ± 0.1	83.4 ± 0.5	84.5 ± 0.2	84.5 ± 0.3	$\mathbf{85.5}\pm0.2$
	100K	81.7 ± 0.2	85.7 ± 0.6	87.0 ± 0.3	87.8 ± 0.3	$\mathbf{88.6}\pm0.3$
	150K	83.0 ± 0.3	86.8 ± 0.4	88.1 ± 0.3	89.0 ± 0.4	$\mathbf{89.9}\pm0.4$
	185K	82.9 ± 0.2	86.9 ± 0.2	88.2 ± 0.3	89.5 ± 0.3	$\mathbf{90.3}\pm0.2$
$d>0$	20K	44.2 ± 0.8	55.9 ± 0.7	56.6 ± 0.8	55.6 ± 0.6	$\mathbf{58.1}\pm1.0$
	50K	44.3 ± 0.3	54.9 ± 0.5	56.7 ± 0.2	56.1 ± 0.2	$\mathbf{59.3}\pm0.5$
	100K	48.4 ± 2.0	63.6 ± 0.6	65.1 ± 0.2	66.5 ± 0.1	$\mathbf{68.7}\pm0.1$
	150K	51.4 ± 0.6	64.7 ± 0.3	66.5 ± 0.2	68.5 ± 0.5	$\mathbf{70.8}\pm0.3$
	185K	49.2 ± 1.0	64.9 ± 1.5	66.5 ± 0.3	69.6 ± 1.5	$\mathbf{71.5}\pm1.6$

P-value in T-test				
Orientations	Generative model	Discriminative models		
	LR	MMR	SVM	ME
$d<0$	$1.02e-8$	$2.75e-5$	$7.27e-5$	$1.00e-4$
$d=0$	$8.66e-10$	$8.39e-7$	$1.55e-5$	$2.19e-6$
$d>0$	$1.97e-9$	$1.45e-6$	$7.19e-6$	$3.84e-9$

Table 5: Classification performance on the Chinese-English corpus: the class-specific F1-scores [%] for three-class setup. Bold numbers refer to the best results. P-values of T-test for statistical significance in the differences between MMS and other models are shown in the lower table.

time increase dramatically. Hence, how to choose the appropriate priors for ME in order to balance training speed and performance is often difficult. Alternatively, using the early stopping strategy DPR with MMS does not over-fit the training data, indicating that the PSL algorithm in company with early stopping already guards against over-fitting.

Figure 11 further demonstrates the average precision for each reordering distance d on the 185K-sentence task (five-class setup), using the results provided by LR, ME and DPR with MMS respectively. It shows that even for long distance reordering, the DPR model still performs well, while the LR baseline usually performs badly (more than half examples are classified incorrectly). With so many classification errors, the effect of this baseline in an SMT system is in doubt, even with a powerful language model. Meanwhile, we observed that results for forward phrase movements (i.e., $d<0$) are better than those for backward reorderings (i.e., $d>0$). We postulate this is because the reordering patterns for backward reorderings also depend on the orientation classes of the phrases nearby. For example, in Figure 3, the phrase "on a building" would be in "forward reordering" if it does not meet another "forward" phrase "a fire has taken place". This observation shows that a richer feature set including a potential orientation class of nearby phrases may help the reordering classification and will be investigated in our future work.

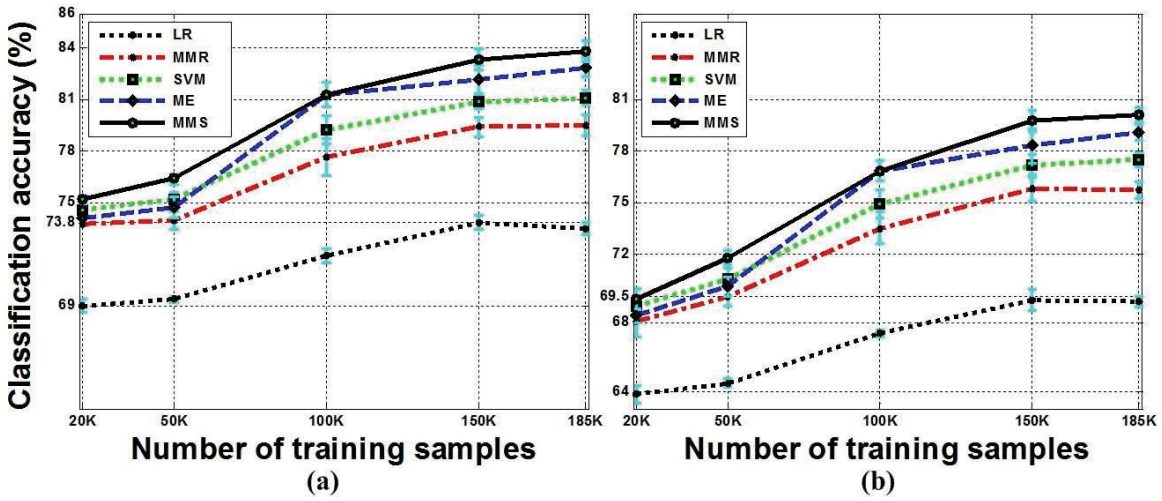


Figure 10: The overall classification precision of three-class setup (Figure (a)) and five-class setup (Figure (b)), where "*K*" indicates thousand and the error bars show the variances.

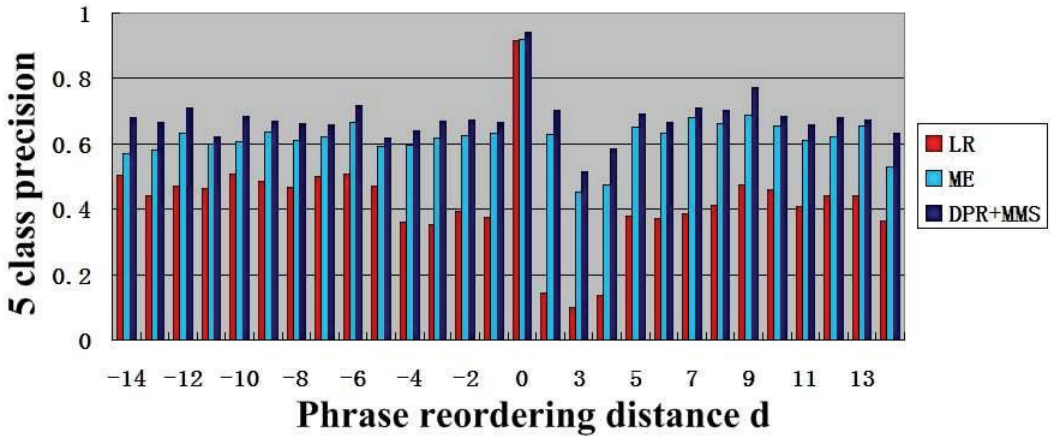


Figure 11: Five-class classification precision with respect to d on the 185K-sentence task. A similar trend is also observed on the three-class classification precision.

Orientations	Training Data	Generative model	Discriminative models			
		LR	MMR	SVM	ME	MMS
$d \leq -5$	20K	40.9 ± 2.4	46.2 ± 1.8	$\mathbf{47.2} \pm 2.4$	45.6 ± 1.9	47.0 ± 1.5
	50K	41.0 ± 0.2	46.5 ± 0.6	48.1 ± 0.4	47.5 ± 0.3	$\mathbf{49.6} \pm 0.7$
	100K	46.9 ± 0.1	54.7 ± 1.5	56.3 ± 0.8	57.3 ± 0.5	$\mathbf{58.7} \pm 0.8$
	150K	47.6 ± 0.9	57.1 ± 1.1	58.9 ± 1.4	60.5 ± 1.3	$\mathbf{62.1} \pm 1.1$
	185K	47.8 ± 0.3	57.6 ± 0.4	59.3 ± 0.3	61.8 ± 0.7	$\mathbf{63.4} \pm 0.7$
$-5 < d < 0$	20K	35.0 ± 1.5	44.6 ± 1.6	45.2 ± 1.3	46.6 ± 1.1	$\mathbf{47.6} \pm 1.0$
	50K	40.8 ± 1.5	52.3 ± 1.2	52.4 ± 0.8	53.8 ± 0.7	$\mathbf{55.5} \pm 0.2$
	100K	43.3 ± 0.5	55.3 ± 1.2	56.1 ± 1.5	58.7 ± 1.4	$\mathbf{60.9} \pm 0.5$
	150K	47.8 ± 1.7	60.8 ± 2.0	61.8 ± 2.0	65.1 ± 2.5	$\mathbf{66.1} \pm 2.0$
	185K	45.7 ± 1.5	59.2 ± 1.5	61.0 ± 1.5	64.8 ± 1.6	$\mathbf{66.0} \pm 1.5$
$d = 0$	20K	79.9 ± 0.3	83.6 ± 0.2	84.0 ± 0.2	83.9 ± 0.2	$\mathbf{84.7} \pm 0.3$
	50K	80.0 ± 0.1	83.7 ± 0.2	84.3 ± 0.2	84.4 ± 0.2	$\mathbf{85.5} \pm 0.2$
	100K	81.4 ± 0.1	86.0 ± 0.6	86.8 ± 0.5	87.6 ± 0.4	$\mathbf{88.6} \pm 0.4$
	150K	82.7 ± 0.3	87.2 ± 0.3	87.9 ± 0.3	88.8 ± 0.5	$\mathbf{89.8} \pm 0.3$
	185K	82.7 ± 0.2	87.2 ± 0.1	88.2 ± 0.2	89.5 ± 0.3	$\mathbf{90.4} \pm 0.2$
$0 < d < 5$	20K	13.4 ± 1.8	39.2 ± 3.0	42.8 ± 3.5	41.0 ± 3.0	$\mathbf{46.3} \pm 2.5$
	50K	22.0 ± 1.7	44.5 ± 1.0	47.6 ± 0.6	45.5 ± 0.4	$\mathbf{50.8} \pm 0.6$
	100K	19.2 ± 2.4	50.9 ± 0.9	53.6 ± 1.5	54.6 ± 1.3	$\mathbf{58.1} \pm 1.1$
	150K	23.8 ± 0.7	50.2 ± 0.9	54.4 ± 0.7	56.8 ± 1.7	$\mathbf{60.4} \pm 1.1$
	185K	19.6 ± 2.8	47.8 ± 2.8	51.4 ± 3.0	56.2 ± 3.7	$\mathbf{60.0} \pm 3.0$
$d \geq 5$	20K	41.4 ± 0.9	47.9 ± 3.5	50.7 ± 1.1	49.9 ± 1.0	$\mathbf{50.8} \pm 2.1$
	50K	39.4 ± 0.8	49.5 ± 0.2	50.9 ± 0.5	50.8 ± 0.4	$\mathbf{55.4} \pm 0.5$
	100K	47.0 ± 1.3	59.9 ± 0.1	61.2 ± 0.6	62.7 ± 0.6	$\mathbf{64.5} \pm 0.8$
	150K	48.8 ± 0.5	62.0 ± 0.1	63.8 ± 0.2	65.2 ± 0.6	$\mathbf{67.1} \pm 0.2$
	185K	49.4 ± 0.6	62.9 ± 1.3	64.9 ± 1.3	67.2 ± 1.4	$\mathbf{68.8} \pm 1.2$

P-value in T-test				
Orientations	Generative model	Discriminative models		
	LR	MMR	SVM	ME
$d \leq -5$	$6.19e-7$	$3.93e-5$	$9.30e-3$	$7.89e-10$
$-5 < d < 0$	$7.77e-10$	$3.07e-7$	$3.69e-7$	$2.02e-5$
$d = 0$	$1.14e-9$	$1.19e-6$	$3.50e-6$	$8.99e-11$
$0 < d < 5$	$2.36e-11$	$5.21e-8$	$8.50e-5$	$9.51e-9$
$d \geq 5$	$4.76e-8$	$4.25e-7$	$8.56e-4$	$1.00e-3$

Table 6: Classification performance on the Chinese-English corpus: the class-specific F1-scores [%] for five-class setup. Bold numbers refer to the best results. P-values of T-test for statistical significance in the differences between MMS and other models are shown in the lower table.

4.1.2 EXPLORING DPR WITH MMS

With the above general view, the DPR model with the MMS learning agent has shown to be the best classifier. Here we further explore its advantages by analysing more detailed results.

Figure 12 first illustrates the relative improvements of DPR with MMS over LR, ME and DPRs with MMR and SVM on the switching classes (i.e., $d \neq 0$), where we observed that the relative

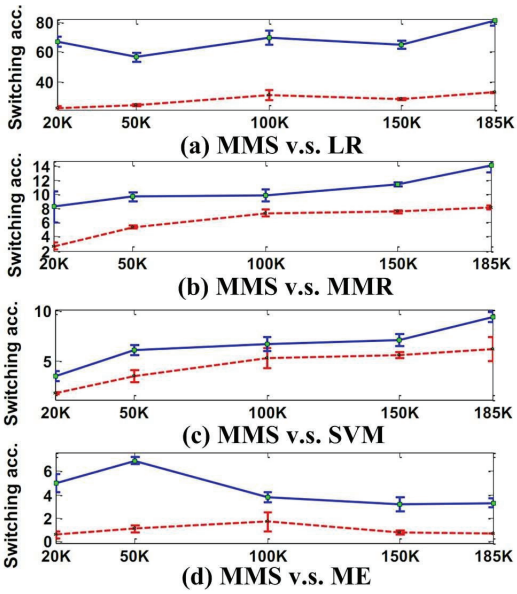


Figure 12: The average relative improvements of DPR with MMS over (a) LR, (b) DPR with MMR, (c) DPR with SVM and (d) ME on the switching classes (i.e., $d \neq 0$) for three-class setup (Red dashed lines) and five-class setup (Blue solid lines). "K" indicates thousand and the error bars show the variances.

improvement with five-class setup is usually greater than that with three-class setup. This implies the more orientation classes DPR has, the better performance MMS achieves compared with other models. This observation makes MMS the most promising learning agent in our future work where we expect to extend the orientation set further.

We then compared the DPR model with MMS with the LR and the ME models according to the overall precision of each cluster on Figure 13. Compared with the generative model LR, DPR performs better in many of the clusters, especially when given enough training samples (the black lines in the figure). This verifies the advantage of discriminative models. In particular, the number of larger circles which imply greater ambiguity in target translations is greater than that of larger rectangles; indicating MMS performs better in these ambiguous clusters, implying that the target translations also contain useful information about phrase movements.

Comparing the two discriminative models, the cluster improvement of DPR over ME is smaller than that over LR, represented by the reduced number of circles and the increased number of rectangles. However, DPR with MMS still achieves a stable improvement over ME. This is especially true when the training samples are not adequate (represented by more circles in Figure 13 (c) than in Figure 13 (d)), where the ME model is more likely to over-fit while the DPR with MMS still performs well.

Finally we illustrate three examples on Figure 14, where we observed a great improvement of DPR over LR. The first (top) example demonstrates the benefit from the target translations as by translating the Chinese source phrase "you guan" into different English words (i.e., "relating", "relates" or "relevant"), the phrase pairs usually have different but regular movements. The second

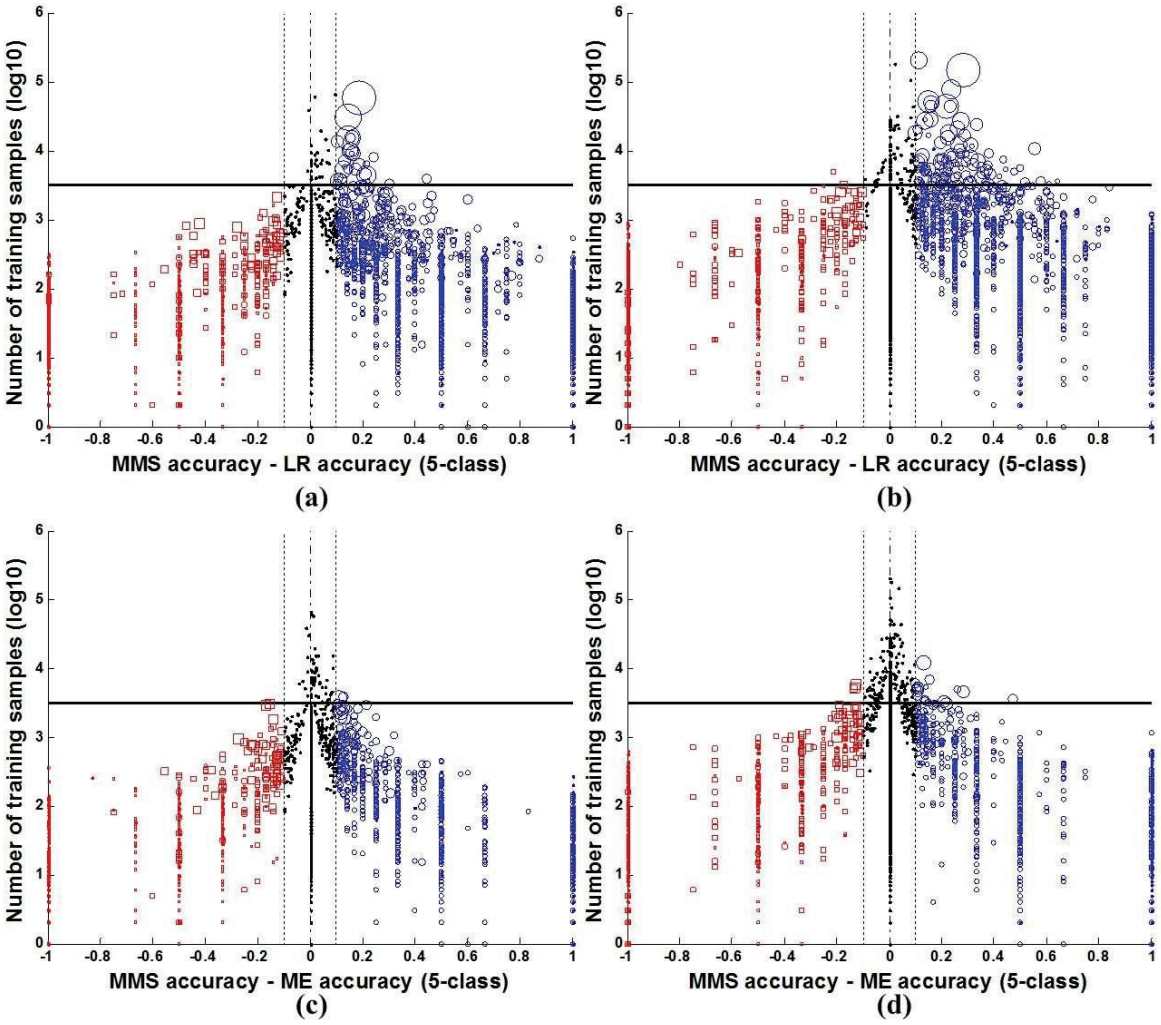


Figure 13: Scatter-plots comparing the cluster accuracies of DPR with MMS with the LR (top) and ME (bottom) models on 50K-sentence task (left) and 150K-sentence task (right). Each circle/rectangle/point represents a cluster that contains all phrase pairs with a unique source phrase (e.g., Figure 8 (a)). Those clusters for which the performance difference (x-axes) is greater than 0.1 are shown as rectangles and circles, the areas of which are proportional to the number of target translations in them. The y-axes show the number of training samples (in log 10 scale) for each cluster.

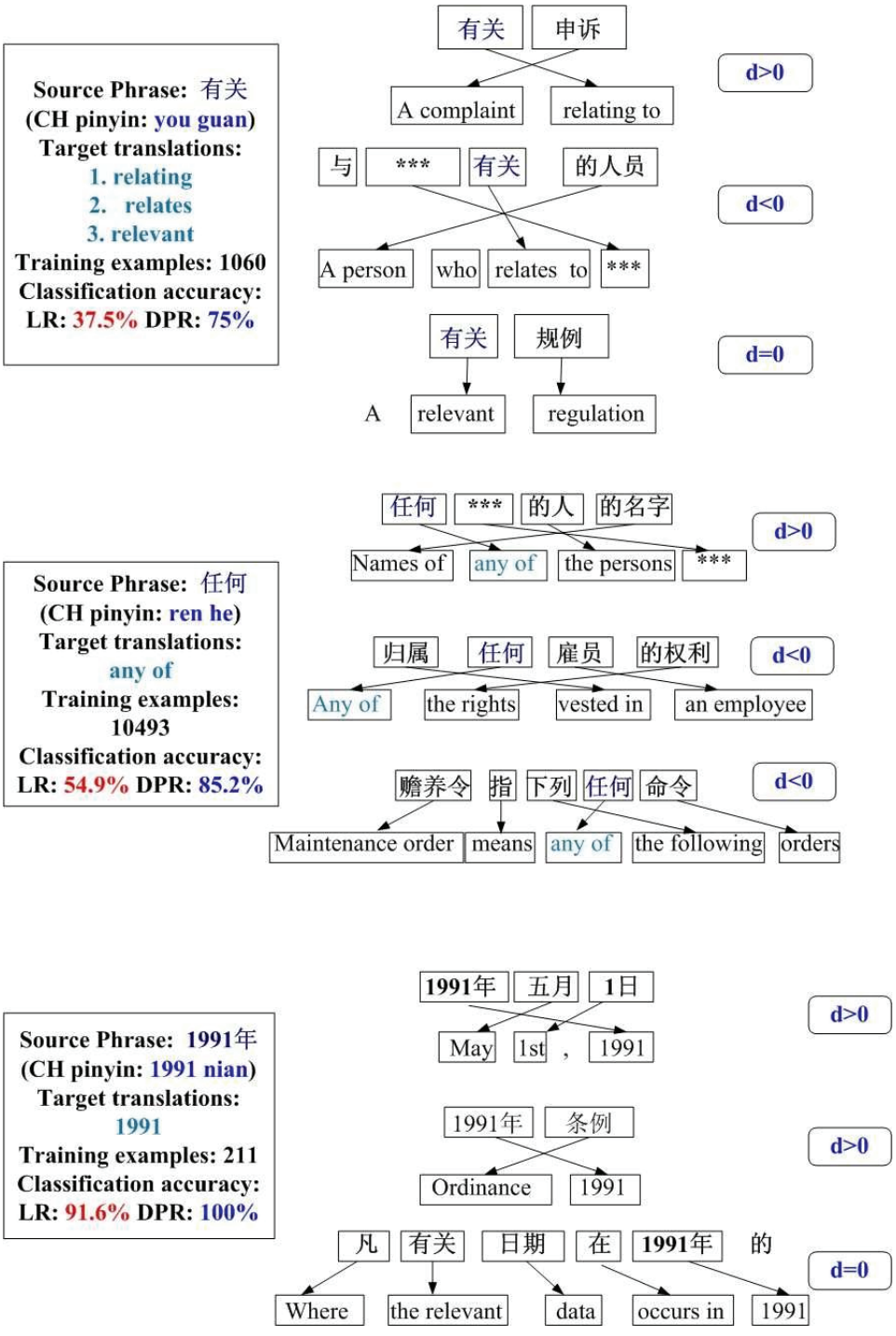


Figure 14: Phrase movements captured by the DPR model with MMS on the 50K-sentence task.

(middle) example shows a grammatical structure captured by the DPR model: in English the phrase "any of" usually stays in front of the subjects (or objects) it modifies. In general, when given enough training samples a discriminative model such as DPR is able to capture various grammatical structures (modelled by phrase movements) better than a generative model. The final (bottom) example depicts one type of phrase movements caused by the constant expressions in different languages (e.g., date expression). Although such expressions can be covered manually with a rule-based MT system, they can easily be captured by a DPR model as well. Hence, we conclude that the frequent phrase movements, whether caused by different grammatical structures or rule-based expressions, can be captured and the movement information is then passed on to an MT decoder to organise the target sentence structures.

4.1.3 A COMPARISON OF THE TRAINING TIME

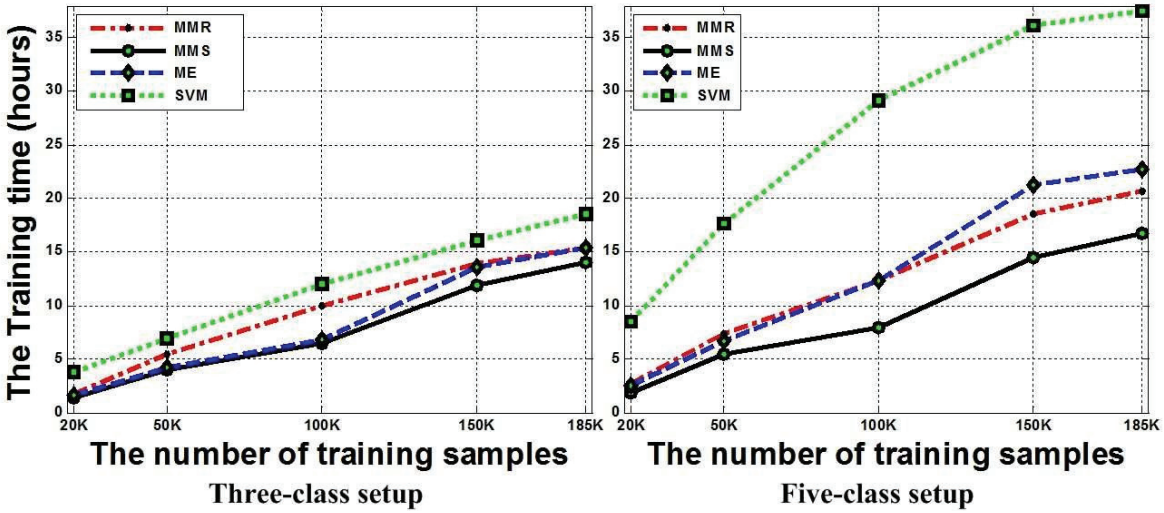


Figure 15: The training time of MMR, ME, MMS (coded in Python) and SVM (coded in C++) to reach the same training error tolerance.

As a comparison, we plot on Figure 15 the training time of MMS, MMR, ME and SVM to reach the same training error tolerance.[9] For the DPR model, MMS is the fastest as expected where in contrast the SVM technique is the slowest. Moreover, training a DPR model with MMS is faster than training an ME model, especially when the number of classes increase. This is because the *generalised iterative scaling* (GIS) algorithm for an ME model requires going through all samples twice at each round: one is for updating the conditional distributions $p(o|f_j, e_i)$ and the other is for updating $\{\mathbf{w}_o\}_{o \in O}$. Alternatively, the PSL algorithm only goes through all examples once at each round, making it faster and more applicable for larger data sets.

9. The MMS, MMR and ME models are coded in Python while SVM-multiclass is coded in C++.

4.2 Machine Translation Experiments

We now test the effectiveness of the DPR model in an MT system, using a state-of-the-art SMT system—MOSES (Koehn et al., 2005) that models phrase movements with the LR models as a baseline system. To keep the comparison fair, our MT system just replaces MOSES's LR models with DPR while sharing all other components (i.e., a phrase translation probability model, a 4-gram language model (Stolcke, 2002) and the beam search decoder). In addition, we also compared the DPR model with the ME model in Zens and Ney (2006) on the 50K-sentence MT task, where the results confirmed that DPR can lead to improvement performance.

We chose MMS as the learning agent for the DPR model in consideration of its prominent classification performance. In detail, all consistent phrase pairs (up to length 8) were extracted from the training sentence pairs and form the sample pool. The DPR model was then trained by the PSL algorithm and the function $h(z) = \exp(z)$ was applied to Equation (3) to transform the prediction scores.

To make use of the phrase reordering probabilities, two strategies were applied: one is to use the probabilities directly as the reordering cost (dotted line in Figure 1), which is also used in Xiong et al. (2006); Zens and Ney (2006); the other is to use them to adjust the word distance-based reordering cost (solid line in Figure 1), where the reordering cost of a sentence is computed as

$$h_d(\mathbf{f}^I, \mathbf{e}^I) = -\sum_{(f_{j_m}, e_{i_m}) \in (\mathbf{f}^I, \mathbf{e}^I)} \frac{d_m}{\beta p(o|f_{j_m}, e_{i_m})} \tag{7}$$

with tuning parameter β. Intuitively, if the DPR model has a large orientation set (i.e., the phrase movements are modelled in a precise way) and the orientation predictions are good enough, it is reasonable to use the reordering probabilities directly. However, as we experienced in Section 4.1, the DPR predictions with five-class setup still need improvement, especially for the switching orientations (i.e., $d \neq 0$). On the other hand, if the DPR model only uses a small orientation set (e.g., three-class setup), it is able to provide very good orientation predictions. But all long distance phrase movements will have the same reordering probabilities, which may mislead the SMT decoder and spoil the translations. In this case, the distance-sensitive expression (7) is able to fill the deficiency of a small-class setup of DPR by penalising long distance phrase movements. Hence in the MT experiments, we used the five-class phrase reordering probabilities directly while the three-class probabilities were used to adjust the word distance-based reordering cost.

For parameter tuning, minimum-error-rating training (MERT) (Och, 2003) is used to tune the parameters. Note that there are seven parameters which need tuning in MOSES's LR models, while there is only one for DPR. The translation performance is evaluated by four standard MT measurements, namely *word error rate* (WER) (Tillmann et al., 1997), *BLEU* (Papineni et al., 2002), *NIST* (Doddington, 2002) and *METEOR* (Banerjee and Lavie, 2005). In effect, WER and NIST weight more on word/phrase translation accuracy; BLEU biases towards translation fluency; and METEOR emphasises translation adequacy (i.e., word/phrase translation recall). The reader is referred to Callison-Burch et al. (2007) and Ni (2010) for detailed discussions on these measurements.

We first demonstrate on Table 7 a comparison of the DPR model with the LR and the ME models on the 50K-sentence Chinese-to-English MT task. The improvements on most evaluations over LR and ME are consistent with what are observed on the reordering classification experiments. However, the MT results show no difference between three-class setup and five-class setup, possibly due to the low classification accuracy of DPR with five-class setup (especially on the switching

Measure	MOSES	DPR		ME	
		3-class	5-class	3-class	5-class
WER [%]	**24.3** $\pm$ 0.6	24.6 $\pm$ 1.5	24.7 $\pm$ 1.1	25.3 $\pm$ 1.7	26.0 $\pm$ 2.1
BLEU [%]	44.5 $\pm$ 1.2	47.1 $\pm$ 1.3	**47.5** $\pm$ 1.2	46.17 $\pm$ 1.7	45.0 $\pm$ 2.5
NIST	8.73 $\pm$ 0.11	**9.04** $\pm$ 0.26	9.03 $\pm$ 0.32	8.72 $\pm$ 0.26	8.49 $\pm$ 0.49
METEOR [%]	66.1 $\pm$ 0.8	**66.4** $\pm$ 1.1	66.1 $\pm$ 1.1	65.0 $\pm$ 1.7	63.9 $\pm$ 2.6

Table 7: The comparison of the DPR model with the LR and the ME models on the 50K-sentence MT task.

classes). How to improve the classification accuracy of DPR with a large class setup is hence a main challenge for our future work.

Since the three-class DPR achieves the same translation quality but it is faster, for the other MT tasks we only used DPR with three-class setup as the phrase reordering model. Figure 16 illustrates the comparison of the DPR MT system with the baseline MOSES according to the four MT evaluations, where we observed consistent improvements on most evaluations. Furthermore, the larger the sample size is, the better results DPR will achieve. This again shows the learning ability of the DPR model when given enough samples. In particular, both systems produce similar predictions in sentence content (represented by similar WERs), but our MT system does better at phrase reordering and produces more fluent translations (represented by better BLEUs).

However, if the sample size is small (e.g., the 20K-sentence task), DPR is unable to collect adequate phrase reordering information. In this case the application of DPR to an MT system may involve a risk of a reduction in translation quality (represented by the low qualities on WER and METEOR).

5. Conclusions and Future Work

We have proposed a distance phrase reordering (DPR) model using a classification scheme and trained and evaluated it in a structured learning framework. The phrase reordering classification tasks have shown that DPR is better at capturing phrase movements over the LR and ME models. The MMS learning agent in particular, achieves outstanding performance in terms of classification accuracy and computational efficiency. An analysis of performance confirms that the proposed MMS method is shown to perform particularly well when there is a large amount of training data, and on translation examples with large ambiguity in the target language domain.

Machine translation experiments carried out on the Chinese-English corpus show that DPR gives more fluent translation results, which confirms its effectiveness. On the other hand, when training data is sparse, the process may involve a risk of a reduction in translation quality.

For future work, we aim to improve the prediction accuracy of the five-class setup before applying it to an MT system, as DPR can be more powerful if it is able to provide more precise phrase positions for the decoder. We also aim to formulate the phrase reordering problem as an ordinal regression problem rather than a classification one proposed in this paper. Furthermore, we will refine the learning framework of DPR by carefully designing or automatically learning the distance matrix $\triangle(o_n, o')$. A richer feature set to better characterise the grammar reorderings is also a direc-

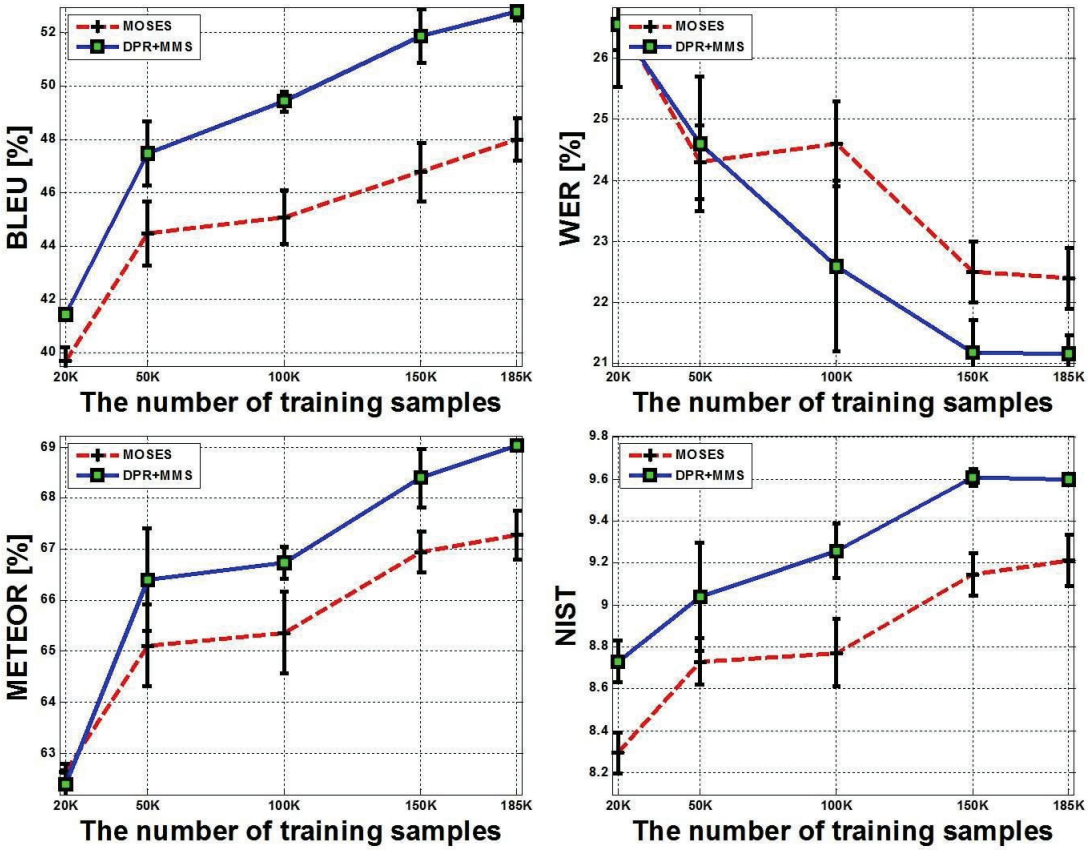


Figure 16: The translation evaluations.

tion of our current investigations. Finally we will try the DPR model on larger corpora (e.g., the NIST Chinese-English corpus), with the purpose of verifying its ability in scaling up to large data collections.

Acknowledgments

This work was funded from an European Community project SMART (FP7-033917). Y. Ni was funded by a scholarship from the School of Electronics and Computer Science, University of Southampton. Integration of the work as part of MOSES was funded by the PASCAL Network of Excellence. Particular thanks go to Prof Philip Koehn and Mr Hieu Hoang, University of Edinburgh, for help with implementation details of MOSES.

Appendix A.

In this appendix we infer the optimisation problem (6) from (4).

To consider adding a regularisation term, we upper bound the norm of each $\mathbf{w}_o$ by $\|\mathbf{w}_o\| \leq R$. Then minimising (5) with respect to $\{\mathbf{w}_o\}_{o\in O}$ is equivalent to solving the following optimisation problem

$$\min_{\|\mathbf{w}_o\|\leq R} L\big(\{\mathbf{w}_o\}_{o\in O}\big), \tag{8}$$

where the cumulative loss $L\big(\{\mathbf{w}_o\}_{o\in O}\big) = \sum_n \rho\big(o_n, f^n, e^n, \{\mathbf{w}_o\}_{o\in O}\big)$.

We can then express the sub maximisation problem $\max_{o\neq o_n}[\triangle(o_n,o)+\mathbf{w}_o^T\phi(f^n,e^n)]-\mathbf{w}_{o_n}^T\phi(f^n,e^n)$ in Equation (5) as a *linear programming* problem

$$\begin{array}{ll} \max\limits_{\mathbf{z}_n} & \sum\limits_{o\in O} z_n^o[\triangle(o_n,o)+\mathbf{w}_o^T\phi(f^n,e^n)] \\ s.t. & \sum\limits_{c} z_n^o = 0 \\ & z_n^{o_n} = -1 \\ & z_n^o \geq 0, o\in\{O\backslash o_n\} \end{array} . \tag{9}$$

Let $\mathcal{Z}_n$ denote the closed set of $\mathbf{z}_n$, $\mathbf{z}=\{\mathbf{z}_1,\ldots,\mathbf{z}_N\}$ and $\mathcal{Z}=\mathcal{Z}_1\times\ldots\times\mathcal{Z}_N$, substituting (9) into (8) yields a natural saddle-point form

$$\min_{\|\mathbf{w}_o\|\leq R}\max_{\mathbf{z}\in\mathcal{Z}} L(\{\mathbf{w}_o\}_{o\in O},\{\mathbf{z}_o\}_{o\in O})$$

with

$$\begin{array}{rl} L(\{\mathbf{w}_o\}_{o\in O},\{\mathbf{z}_o\}_{o\in O}) = & \sum\limits_{n=1}^{N}\max\Big\{0, \sum\limits_{o\in O} z_o^n\big(\triangle(o_n,o)+\mathbf{w}_o^T\phi(f^n,e^n)\big)\Big\} \\ s.t. & \left\{\begin{array}{ll} z_o^n=-1 & o=o_n \\ z_o^n\geq 0 & o\neq o_n \quad n=1,\ldots,N \\ \sum\limits_{o\in O} z_o^n = 0 & \end{array}\right. \end{array} ,$$

which is the optimisation problem (6).

Appendix B.

In this appendix, we describe more details about maximum margin regression (MMR).

To illuminate the background of the MMR, let the constraints

$$\begin{aligned} \text{s.t.} \quad & \sum_{o\in O} \varphi(o_n,o)\mathbf{w}_o^T\phi(f^n,e^n) \geq 1-\xi(f^n,e^n) \\ & \xi(f^n,e^n)\geq 0,\ \forall (f^n,e^n)\in\Upsilon \end{aligned}$$

be transformed into an inner product based form

$$\begin{aligned} & \sum_{o\in O} \varphi(o_n,o)\mathbf{w}_o^T\phi(f^n,e^n) \geq 1-\xi(f^n,e^n) \\ & \Rightarrow \langle \varphi(o_n), \mathbf{W}\phi(f^n,e^n)\rangle \geq 1-\xi(f^n,e^n) \end{aligned} \quad , \tag{10}$$

where the following short hand notations are applied: $\mathbf{W}=\{\mathbf{w}_o^T\}\in\mathbb{R}^{|O|\times dim(\phi)}$ is a matrix in which the rows correspond to the row vectors $\mathbf{w}_o^T$, and $\varphi(o_n)=(\varphi(o_n,o)),\ o\in O$ is a vector of the indicator values.

To give a possible interpretation to the inner product based constraints consider the well known cosine rule connecting the inner product and the distance via norms in a L_2 norm space, namely for every pair of vectors $\mathbf{a},\mathbf{b}$ of this space the following holds $\|\mathbf{a}-\mathbf{b}\|_2^2=\|\mathbf{a}\|_2^2+\|\mathbf{b}\|_2^2-2\langle\mathbf{a},\mathbf{b}\rangle$. Exploiting this equality the inner product based constraints (10) can be transformed into an equivalent, norm based one

$$\|\varphi(o_n)\|_2^2+\|\mathbf{W}\phi(f^n,e^n)\|_2^2-2+2\xi(f^n,e^n)\geq\|\varphi(o_n)-\mathbf{W}\phi(f^n,e^n)\|_2^2 \ . \tag{11}$$

Constraints (11) state that the squared distance between the vector valued representation of the outputs and the image of the input vectors with respect to the linear operator matrix $\mathbf{W}$ is bounded above by summing of the square norm of outputs, the norm of the input image and a tolerance given by the margin. Therefore if the Frobenius norm of $\mathbf{W}$ is minimised then the constraints force the distance between the outputs and the image of the inputs to be small. If the norm of all outputs are the same this minimisation works uniformly, otherwise larger distance error is allowed for outputs with greater norm.

The optimisation problem of MMR allows to use implicit representation not only for the inputs but also for the outputs. To see that one can introduce Lagrangian multipliers, $\alpha(f^n,e^n)$, to each constraints and write up the dual problem for MMR

$$\begin{aligned} \min \quad & \tfrac{1}{2}\sum_{(f^n,e^n)}\sum_{(f^m,e^m)}\alpha(f^n,e^n)\alpha(f^m,e^m)\overbrace{\langle\varphi(o_n),\varphi(o_m)\rangle}^{\kappa^{\varphi}_{nm}}\overbrace{\langle\phi(f^n,e^n),\phi(f^m,e^m)\rangle}^{\kappa^{\phi}_{nm}} \\ & -\sum_{(f^n,e^n)}\alpha(f^n,e^n) \\ \text{w.r.t} \quad & \alpha(f^n,e^n),\ \forall(f^n,e^n)\in\Upsilon \\ \text{s.t.} \quad & 0\leq\alpha(f^n,e^n)\leq C,\ \forall(f^n,e^n)\in\Upsilon \end{aligned} \quad ,$$

where κ^{ϕ}_{nm} and κ^{φ}_{nm} stand for the inner products between the input and the output pairs respectively. Therefore, to solve the dual problem requires only the knowledge of the values of inner products of the output pairs and not their concrete feature representation. See further motivations behind the formulation of MMR in Szedmak and Hussain (2009).

References

S. Banerjee and A. Lavie. Meteor: an automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and Summarisation*, pages 65–72, Ann Arbor, Michigan, June 2005.

W. S. Bennett and J. Slocum. The lrc machine translation system. *Computational Linguistics*, 11 (2-3):111–121, 1985.

A. Berger, S. D. Pietra, and V. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–72, March 1996.

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

C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June 2007.

M. Collins. Discriminative training methods for hidden markov models: theory and experiments with percepton algorithms. In *C. Sammut, A. G. Hoffmann (Eds.) Proceedings of the 19th International Conference on Machine Learning (ICML 2002)*, 2002.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research (HLT)*, 2002.

C. Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2:213–242, 2001.

J. Giménez and L. Màrquez. Context–aware discriminative phrase selection for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 159–166, Prague, June 2007.

J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

K. Knight. Decoding complexity in word replacement translation models. *Computational Linguistics*, 25(2):607–615, 1999.

P. Koehn. Pharaoh: a beam search decoder for phrase–based statistical machine translation models. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA 2004)*, pages 115–124, October 2004.

P. Koehn, A. Axelrod, A. B. Mayne, C. Callison-Burch, M. Osborne, and D. Talbot. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2005)*, Pittsburgh, PA, October 2005.

Y. Ni. *Beyond Multi-class–Structured Learning for Machine Translation*. PhD thesis, University of Southampton, Southampton, U.K., 2010.

Y. Ni, M. Niranjan, C. Saunders, and S. Szedmak. Distance phrase reordering for moses - user manual and code guide. Technical report, School of Electronics and Computer Science, University of Southampton, Southampton, UK, 2010a.

Y. Ni, C. Saunders, S. Szedmak, and M. Niranjan. The application of structured learning in natural language processing. *Machine Translation Journal*, 24(2):71–85, 2010b.

F. J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, Japan, September 2003.

F. J. Och, C. Tillmann, and H. Ney. Improved alignment models for statistical machine translation. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, University of Maryland, College Park, MD, June 1999.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, 2002.

A. Stolcke. Srilm - an extensible language modeling toolkit. In *J. H. L. Hansen and B. Pellom (Eds.) Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, Denver, Colorado, USA, September 2002.

S. Szedmak and Z. Hussain. A universal machine learning optimization framework for arbitrary outputs. Technical report, PASCAL, Southampton, UK, 2009.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: maximum margin regression; multiclass and multiview learning at one-class complexity. Technical report, PASCAL, Southampton, UK, 2006.

B. Taskar, C. Guestrin, and D. Koller. Max–margin markov networks. In *S. Thrun, L. K. Saul, B. Schölkopf (Eds.) Proceedings of 7th Annual Conference on Neural Information Processing Systems (NIPS 2003)*, Vancouver, Canada, 2003.

B. Taskar, S. Lacoste-Julien, and M. I. Jordan. Structured prediction, dual extragradient and bregman projections. *Journal of Machine Learning Research, "Special Topic on Machine Learning and Optimization"*, pages 1627–1653, 2006.

C. Tillmann. A block orientation model for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, Boston, MA, USA, 2004.

C. Tillmann and T. Zhang. A localized prediction model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 557–564, Ann Arbor, MI, June 2005.

C. Tillmann, S. Vogel, H. Ney, H. Sawaf, and A. Zubiaga. Accelerated dp based search for statistical translation. In *Proceedings of the 5th European Conference on Speech Communication and Technology*, 1997.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Russ Greiner, Dale Schuurmans (Eds.) Proceedings of the 21st International Machine Learning Conference (ICML 2004)*. ACM Press, 2004.

D. Vickrey, L. Biewald, M. Teyssier, and D. Koller. Word–sense disambiguation for machine translation. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT–EMNLP 2005)*, pages 771–778, 2005.

W. Weaver. Translation. In W. N. Locke and A. D. Booth, editors, *Machine Translation of Languages: Fourteen Essays*, pages 15–23, 1949.

D. Xiong, Q. Liu, and S. Lin. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, July 2006.

R. Zens and H. Ney. Discriminative reordering models for statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 55–63, New York City, June 2006.

R. Zens, O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. The rwth phrase–based statistical machine translation system. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2005)*, pages 155–162, Pittsburgh, PA, October 2005.

Journal of Machine Learning Research 12 (2011) 31-67 Submitted 11/09; Revised 9/10; Published 1/11

Improved Moves for Truncated Convex Models

M. Pawan Kumar PAWAN@CS.STANFORD.EDU
Computer Science Department
Stanford University
Stanford, CA 94305, USA

Olga Veksler OLGA@CSD.UWO.CA
Computer Science Department
University of Western Ontario
London, ON N6A 5B7, Canada

Philip H.S. Torr PHILIPTORR@BROOKES.AC.UK
Department of Computing
Oxford Brookes University
Oxford, OX33 1HX, UK

Editor: Tommi Jaakkola

Abstract

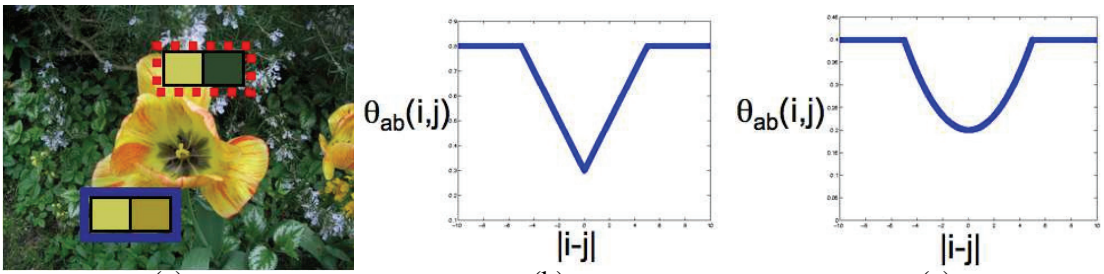
We consider the problem of obtaining an approximate maximum *a posteriori* estimate of a discrete random field characterized by pairwise potentials that form a truncated convex model. For this problem, we propose two *st*-MINCUT based move making algorithms that we call Range Swap and Range Expansion. Our algorithms can be thought of as extensions of $\alpha\beta$-Swap and α-Expansion respectively that fully exploit the form of the pairwise potentials. Specifically, instead of dealing with one or two labels at each iteration, our methods explore a large search space by considering a range of labels (that is, an interval of consecutive labels). Furthermore, we show that Range Expansion provides the same multiplicative bounds as the standard linear programming (LP) relaxation in polynomial time. Compared to previous approaches based on the LP relaxation, for example interior-point algorithms or tree-reweighted message passing (TRW), our methods are faster as they use only the efficient *st*-MINCUT algorithm in their design. We demonstrate the usefulness of the proposed approaches on both synthetic and standard real data problems.

Keywords: truncated convex models, move making algorithms, range moves, multiplicative bounds, linear programming relaxation

1. Introduction

Discrete pairwise random fields are a useful tool for concisely specifying the probability of a labeling (that is, an assignment of values) for a set of discrete random variables. Hence, they offer an elegant formulation for several problems in computer vision, from low-level tasks such as stereo reconstruction and image denoising (Szeliski et al., 2008) to high-level tasks such as pose estimation (Felzenszwalb and Huttenlocher, 2000) and scene segmentation (Shotton et al., 2006). Once formulated within this framework, the problem is typically solved by obtaining the maximum *a posteriori* (MAP) estimate, that is, finding the labeling that minimizes the corresponding Gibbs energy (hereby referred to as simply the energy). This is well-known to be an NP-hard problem and thus, requires us to come up with accurate approximation algorithms.

©2011 M. Pawan Kumar, Olga Veksler and Philip H.S. Torr.



(a) (b) (c)

Figure 1: (a) An example of a natural image that consists of smoothly varying intensities (for instance, the two enlarged pixels bounded in the solid box near the bottom of the image; shown in blue if viewed in color) and sharp edges (for instance the two enlarged pixels bounded in the dashed box near the top of the image; shown in red if viewed in color). The smooth variation is captured by the convex part of truncated convex models. The sharp edges are not over penalized due to the truncation, thereby making the potentials robust. (b)-(c) Two examples of truncated convex potentials that will be of interest to us in this work: truncated linear metric (b) and truncated quadratic semi-metric (c).

It is common practice in computer vision to specify an energy function with arbitrary unary potentials and truncated convex pairwise potentials (Boykov et al., 2001; Szeliski et al., 2008; Veksler, 1999). This is especially true in low-level vision where the use of truncated convex models is motivated by the fact that pixels belonging to the same *object* are similar in appearance—captured by the convex part of the pairwise potentials—while pixels belonging to different objects induce an edge in the image—captured by the truncated part (see Fig. 1). In other words, convexity encourages smoothness while truncation ensures that edges are not over penalized. Given their widespread use, the problem of MAP estimation for truncated convex models merits special attention.

In this work, we develop two approaches, called Range Swap and Range Expansion, that take advantage of the special form of the pairwise potentials to obtain an accurate MAP estimate. Specifically, our methods iteratively minimize the energy by searching over a subset of the possible labelings specified by the original problem. Each iteration is formulated as an *st*-MINCUT problem for which there exist several efficient algorithms (Boykov and Kolmogorov, 2004). Unlike other *st*-MINCUT based approaches (Boykov et al., 2001) that restrict the number of labels for each random variable at an iteration to at most 2, our methods explore a large search space by considering a range of labels (that is, an interval of consecutive labels). Our methods are both practically useful and theoretically interesting: in practice, they provide an improved performance (lower energy labelings); in theory, we show that Range Expansion provides the same guarantees as the standard linear programming (LP) relaxation in polynomial time. Specifically, it obtains the LP relaxation's multiplicative bounds for the truncated linear and truncated quadratic pairwise potentials. Note that this does not imply that it provides the same solution as the LP relaxation. However, as our experiments will demonstrate, they provide comparable results (the LP relaxation typically provides lower energy values but at a high computational cost).

Before proceeding further, we would like to note here that the algorithms presented in this paper can be trivially extended to *truncated submodular models*, where submodularity is as defined in Schlesinger and Flach (2006) and is a strict generalization of convexity (Ishikawa, 2003). However, we will restrict our discussion to truncated convex models as it makes the description and analysis of our methods simpler. For clarity of presentation, many of the proofs are reported in the Appendix.

Preliminary versions of this paper have appeared as Kumar and Torr (2008) and Veksler (2007). The project webpage is located at the following URL:

`http://ai.stanford.edu/~pawan/research/truncated-moves.html`.

2. Related Work

Given the popularity of truncated convex models, it is not surprising that the corresponding MAP estimation problem has been well-studied in the literature. For example, Felzenszwalb and Huttenlocher (2004) improved the efficiency of the popular max-product belief propagation (BP) algorithm Pearl, 1988 by using the special form of the pairwise potentials. Note that BP provides the exact MAP estimate for tree-structured random fields. However, for a general neighborhood structure, BP is not guaranteed to converge.

The results of Felzenszwalb and Huttenlocher (2004) can be used directly to speed-up the tree-reweighted message passing algorithm (TRW) (Wainwright et al., 2005) and its sequential variant TRW-S (Kolmogorov, 2006). Both TRW and TRW-S attempt to optimize the Lagrangian dual of the standard LP relaxation of the MAP estimation problem (Chekuri et al., 2005; Koster et al., 1998; Schlesinger, 1976; Wainwright et al., 2005). Unlike BP and TRW, TRW-S is guaranteed to converge. However, TRW-S and other related algorithms (Globerson and Jaakkola, 2007; Komodakis et al., 2007; Schlesinger and Giginyak, 2007a,b) suffer from the following problems: (i) An extensive comparison of energy minimization algorithms by Szeliski et al. (2008) revealed that TRW-S is slower than *st*-MINCUT based algorithms. Other approaches, such as dual coordinate ascent (Globerson and Jaakkola, 2007) or dual decomposition (Komodakis et al., 2007), are even slower than TRW-S in practice (even though, unlike TRW-s, dual decomposition is capable of escaping from the *weak tree agreement* local minimum). (ii) TRW-S and the related methods attempt to solve the dual of the LP relaxation. When the dual is not *decodable* (that is, when the LP relaxation is not tight for a specific instance of the problem), the primal solution is often obtained in a heuristic fashion (for example, by using monotonic chain decoding Meltzer et al., 2005).

Another way of solving the LP relaxation is to resort to interior point algorithms (Boyd and Vandenberghe, 2004) or iterative Bregman projections (Ravikumar et al., 2008). These approaches provide the primal (possibly fractional) solution of the LP relaxation, but at a high computational cost. In our experience, the existing software for interior point algorithms is unable to deal with energy minimization problems for moderately sized (620×480) images. However, if a primal solution can be obtained then certain randomized rounding schemes provide the following guarantees (Chekuri et al., 2005):

- For Potts model, a multiplicative bound of 2 is obtained by using the rounding scheme of Kleinberg and Tardos (1999).
- For the truncated linear metric, a multiplicative bound of $2+\sqrt{2}$ is obtained using the rounding scheme of Chekuri et al. (2005).

- For the truncated quadratic semi-metric, a multiplicative bound of $O(\sqrt{M})$ is obtained using the rounding scheme of Chekuri et al. (2005). Here M is the truncation factor.

The algorithms most related to our approach are the so-called move making methods that rely on solving a series of *st*-MINCUT problems. Move making algorithms start with an initial labeling and iteratively minimize the energy by moving to a better labeling. At each iteration, (a subset of) random variables have the option of either retaining their old label or taking a new label from a subset of the labels $\mathbf{l}$. For example, in the $\alpha\beta$-swap algorithm (Boykov et al., 2001) the variables currently labeled l_α or l_β can either retain their labels or swap them (that is, some variables labeled l_α can be relabeled as l_β and vice versa). In the α-expansion algorithm (Boykov et al., 2001), each variable can either retain its label or get assigned the label l_α during an iteration. Unlike $\alpha\beta$-swap, which has no guarantees on the quality of its solution, the α-expansion algorithm and its generalization using a primal-dual scheme (Komodakis and Tziritas, 2007) provide the following bounds:

- For the Potts model, a multiplicative bound of 2 is obtained using α-expansion (Boykov et al., 2001).
- For the truncated linear metric, a multiplicative bound of $2M$ is obtained using α-expansion (Boykov et al., 2001).
- For the truncated quadratic semi-metric, a multiplicative bound of $2M$ is obtained using the primal-dual scheme of Komodakis and Tziritas (2007).

It is also worth noting that we can obtain a bound of 2 for the related multiway cut problem (Vazirani, 2001) using the *st*-MINCUT algorithm.

Both $\alpha\beta$-swap and α-expansion only allow a variable to take one of two possible labels at each iteration. In other words, they are restricted to a small search space during each move. Gupta and Tardos (2000) extended the α-expansion algorithm for the truncated linear metric by considering a range of labels and provided a multiplicative bound of 4. However, their method is not applicable for the case of truncated quadratic semi-metric. Note that the bounds obtained by all the above move making algorithms are inferior to the bounds obtained by the LP relaxation for truncated convex models (as summarized in table 1). In fact, a recent result shows that the bounds obtained by Boykov et al. (2001) and Komodakis and Tziritas (2007) can also be achieved using the simple iterated conditional modes (ICM) algorithm Gould et al., 2009. However, despite providing inferior bounds, move making algorithms use only a single *st*-MINCUT at each iteration and hence, are often faster than interior point algorithms, TRW, TRW-S and BP.

3. Preliminaries

Before providing the details for Range Swap and Range Expansion, we set up the notation and briefly review some preliminary concepts that are used in the remainder of the paper.

3.1 Random Field

Given data $\mathbf{D}$ (for example, an image or a video), random fields model the probability of a set of random variables $\mathbf{v}$, that is, either the joint distribution of $\mathbf{v}$ and $\mathbf{D}$ as in the case of Markov random fields (MRF) (Besag, 1986) or the conditional distribution of $\mathbf{v}$ given $\mathbf{D}$ as in the case of conditional

	α-exp	PD	GT	LP	Our (Range Expansion)
Potts	2	2	-	2	-
Trunc. Linear	2M	2M	4	$2+\sqrt{2}$	$2+\sqrt{2}$
Trunc. Quad.	-	2M	-	$O(\sqrt{M})$	$O(\sqrt{M})$

Table 1: The multiplicative bounds obtained by various algorithms for the three commonly used truncated convex models. PD refers to the primal-dual method of Komodakis and Tziritas (2007), GT refers to the method by Gupta and Tardos (2000) and LP refers to the multiplicative bounds obtained by the LP relaxation. Note that, unlike our approach, previous move making algorithms provide inferior bounds compared to LP for truncated linear metric and truncated quadratic semi-metric.

random fields (CRF) (Lafferty et al., 2001). The word 'discrete' refers to the fact that each of the random variables $v_a \in \mathbf{v} = \{v_0, \cdots, v_{n-1}\}$ can take one label from a discrete set $\mathbf{l} = \{l_0, \cdots, l_{h-1}\}$. Throughout this paper, we will assume an MRF framework while noting that our results are equally applicable for a CRF.

An MRF defines a neighborhood relationship (denoted by $\mathcal{E}$) over the random variables such that $(a,b) \in \mathcal{E}$ if v_a and v_b are neighboring random variables. Given an MRF, a labeling refers to a function f such that

$$f : \{0, \cdots, n-1\} \longrightarrow \{0, \cdots, h-1\}.$$

In other words, the function f assigns the label $l_{f(a)} \in \mathbf{l}$ to each random variable $v_a \in \mathbf{v}$. The probability of the labeling is given by the following Gibbs distribution:

$$\Pr(f, \mathbf{D}|\theta) = \frac{1}{Z(\theta)} \exp(-Q(f, \mathbf{D}; \theta)), \tag{1}$$

where θ is the parameter vector of the MRF and $Z(\theta)$ is the partition function. Since we consider pairwise MRFs, the energy can be written as:

$$Q(f, \mathbf{D}; \theta) = \sum_{v_a \in \mathbf{v}} \theta_a(f(a)) + \sum_{(a,b) \in \mathcal{E}} \theta_{ab}(f(a), f(b)).$$

Here, $\theta_a(f(a))$ denotes unary potentials and $\theta_{ab}(f(a), f(b))$ denotes pairwise potentials, that is, $\theta_a(f(a))$ is the cost of assigning label $l_{f(a)}$ to variable v_a and $\theta_{ab}(f(a), f(b))$ is the cost of assigning labels $l_{f(a)}$ and $l_{f(b)}$ to variables v_a and v_b respectively. Using Equation (1) it follows that the labeling f that maximizes the posterior $\Pr(f, \mathbf{D}; \theta)$ (that is, the MAP estimate) can be obtained by minimizing the energy.

3.2 Truncated Convex Models

We consider the problem of MAP estimation of random fields where the pairwise potentials are defined by truncated convex models (Veksler, 1999). Formally speaking, the pairwise potentials are of the form

$$\theta_{ab}(f(a), f(b)) = w_{ab} \min\{d(f(a) - f(b)), M\},$$

where $w_{ab} \geq 0$ for all $(a,b) \in \mathcal{E}$, $d(\cdot)$ is a convex function and $M > 0$ is the truncation factor. Here, the term 'convex' is used according to the definition of Ishikawa (2003). Specifically, a function $d(\cdot)$ defined over integers is convex if, and only if,

$$d(x+1) - 2d(x) + d(x-1) \geq 0, \forall x \in \mathbb{Z}.$$

It is assumed that $d(x) = d(-x)$. Examples of pairwise potentials of this form include the truncated linear metric and the truncated quadratic semi-metric, that is,

$$\theta_{ab}(f(a), f(b)) = w_{ab} \min\{|f(a) - f(b)|, M\},$$
$$\theta_{ab}(f(a), f(b)) = w_{ab} \min\{(f(a) - f(b))^2, M\}.$$

An illustration of the above potentials is provided in Fig. 1(b)-(c).

3.3 Multiplicative Bounds

The worst case accuracy of a MAP estimation approach can be expressed using its multiplicative bound. Formally, let f be the labeling obtained by an algorithm A (randomized or deterministic) for an instance of the MAP estimation problem belonging to a particular class (in our case when the pairwise potentials form a truncated convex model). Let f^* be the optimal labeling. The algorithm A is said to achieve a multiplicative bound of σ if for every instance in the specific class the following holds true:

$$E\left(\frac{Q(f, \mathbf{D}; \theta)}{Q(f^*, \mathbf{D}; \theta)}\right) \leq \sigma,$$

where $E(\cdot)$ denotes the expectation and can be dropped from the above inequality if the algorithm is deterministic (as in our case).

3.4 The *st*-MINCUT Problem

Given a directed, non-negatively weighted graph with two terminal vertices s (the source) and t (the sink), an st-cut is defined as a partitioning of the vertices of the graph into two disjoint sets such that the first partition contains s while the second partition contains t. The st-MINCUT problem is to find the minimum cost st-cut, where the cost of a cut is measured as the sum of the weights of the edges whose starting point belongs to the first partition and ending point belongs to the second partition. It is well-known that the st-MINCUT problem can be formulated as a linear program (LP) (which is different but closely related to the LP relaxation for the general MAP estimation problem) with integer solutions. The st-MINCUT problem has several efficient polynomial and pseudo-polynomial solvers (Boykov and Kolmogorov, 2004; Dinic, 1970; Goldberg and Tarjan, 1988). In this work, we employ the pseudo-polynomial solver of Boykov and Kolmogorov (2004) that has been shown to have a linear complexity in practice for several computer vision tasks. The low complexity of this algorithm is responsible for making our iterative algorithm (which solves an st-MINCUT problem at each iteration) computationally efficient.

In order to help the reader follow the arguments of the paper, we provide the list of terms used throughout the paper along with their meanings in table 2.

$\mathbf{D}$	Data provided by the user (for example, an image or a video).
n	Number of random variables.
$\mathbf{v}$	Set of random variables $\{v_0, \cdots, v_{n-1}\}$.
$\mathcal{E}$	Set of neighboring random variables v_a and v_b (denoted by $(a,b) \in \mathcal{E}$).
h	Number of labels.
$\mathbf{l}$	Set of labels $\{l_0, \cdots, l_{h-1}\}$.
I_m	Interval of consecutive labels $[i_m + 1, j_m]$.
L	Length of the interval, that is, $L = j_m - i_m$.
I_r	Set of intervals $\{[0,r],[r+1,r+L], \cdots, [.,h-1]\}$.
f	Labeling of the random field (v_a takes a label $l_{f(a)}$).
f^*	An optimal (MAP) labeling of the random field.
$\theta_a(i)$	Unary potential of assigning label l_i to v_a.
w_{ab}	Weight for neighboring random variables $(a,b) \in \mathcal{E}$.
$d(\cdot)$	Convex function used to define the distance between two labels.
$\hat{d}(\cdot)$	$\hat{d}(x) = d(x+1) - d(x) - d(1) + d(0)/2$.
M	Truncation factor.
κ_{ab}	Constant $w_{ab} d(L)$.
$\theta_{ab}(i,j)$	The pairwise potential $w_{ab} \min\{d(i-j), M\}$ of assigning labels l_i and l_j to neighboring random variables v_a and v_b respectively.
θ	Parameter vector of the discrete random field.
$Q(f; \mathbf{D}, \theta)$	Energy of the labeling f given the data $\mathbf{D}$ and parameters θ.
S	Index for a subset of random variables $S \subseteq \{0, 1, \cdots, n_1\}$.
$\mathbf{v}(S)$	$\{v_a \in \mathbf{v}, a \in S\}$.
$\mathcal{A}(S)$	$\{(a,b) \in \mathcal{E}, a \in S, b \in S\}$.
$\mathcal{B}_1(S)$	$\{(a,b) \in \mathcal{E}, a \in S, b \notin S\}$.
$\mathcal{B}_1(S)$	$\{(a,b) \in \mathcal{E}, a \notin S, b \in S\}$.
$\mathcal{B}(S)$	$\mathcal{B}_1(S) \cup \mathcal{B}_2(S)$.
$\mathbf{v}(f, I_m)$	$\{v_a \in \mathbf{v}, f(a) \in I_m\}$.
$\mathcal{A}(f, I_m)$	$\{(a,b) \in \mathcal{E}, f(a) \in I_m, f(b) \in I_m\}$.
$\mathcal{B}_1(f, I_m)$	$\{(a,b) \in \mathcal{E}, f(a) \in I_m, f(b) \notin I_m\}$.
$\mathcal{B}_2(f, I_m)$	$\{(a,b) \in \mathcal{E}, f(a) \notin I_m, f(b) \in I_m\}$.
$\mathcal{B}(f, I_m)$	$\mathcal{B}_1(f, I_m) \cup \mathcal{B}_2(f, I_m)$.
$\mathcal{G}_m$	Graph corresponding to I_m over which an *st*-MINCUT problem is defined.
$\mathcal{V}_m$	Set of vertices a_k and $b_{k'}$ for $\mathcal{G}_m$ such that $k, k' \in I_m$.
$\mathcal{E}_m$	Set of edges $(a_k, b_{k'})$ for $\mathcal{G}_m$.

Table 2: List of the various terms used throughout the paper.

4. Why Range Moves?

As mentioned earlier, our methods differ from previous move making approaches that deal with only 1 or 2 (not necessarily consecutive) labels at each iteration by considering a range of labels. In other words, we obtain a local minimum labeling with respect to a large search space defined by intervals of consecutive labels. To motivate our choice of using a range of labels, we show that an algorithm that obtains the local minimum with respect to *smooth labelings* provides a small multiplicative bound and hence, a tight approximation. Before proceeding further, we require the following definitions.

Definition 1: Let $S \subseteq \{0, \cdots, n-1\}$ be a subset of the indices of the random variables. A labeling f is said to be *smooth* with respect to S if, and only if, for each $(a,b) \in \mathcal{E}$ such that $a \in S$ and $b \in S$, there exists a path $a_0 = a, a_1, \cdots, a_q = b$ such that $(a_i, a_{i+1}) \in \mathcal{E}$, $a_i \in S$ and $d(f(a_i) - f(a_{i+1})) \leq M$ for all $i = 0, 1, \cdots, q-1$. In other words, the pairwise potential for each edge in the path lies in the convex part (indicating the lack of a discontinuity, hence the name smooth labeling). Note that this does not necessarily imply that $d(f(a), f(b)) \leq M$.

Definition 2: A labeling $\hat{f}$ is said to be a local minimum over smooth labelings if the energy cannot be reduced further by changing the labels of any subset of random variables, say defined by S, such that the new labeling f is smooth with respect to S. In other words, if $f(a) = \hat{f}(a)$ for all $a \notin S$ and f is smooth with respect to S, then $Q(\hat{f}, \mathbf{D}; \theta) \leq Q(f, \mathbf{D}; \theta)$, for all $S \subseteq \{0, \cdots, n-1\}$.

Using the above definitions, we can state the following theorem.

Theorem 1: An algorithm that provides a local minimum over smooth labelings achieves a multiplicative bound of 2 (Proof in Appendix A).

Note that a multiplicative bound of 2 is superior to the best known approximation guarantees (obtained by the LP relaxation). However, an algorithm that provides the desired local minimum labeling would be computationally infeasible. To see why, consider a random field with three variables v_a, v_b and v_c that are neighbors of each other. Suppose there exists a labeling f such that $d(f(a) - f(b)) \leq M, d(f(b) - f(c)) \leq M$ and $d(f(a) - f(c)) > M$. Note that this labeling is smooth since we can find a path from v_a to v_c via v_b such that the edges in the path lie in the convex part. In order to obtain a local minimum over smooth labelings, an algorithm needs to be able to search over such labelings f (that is, provide the optimal move over all smooth labelings). This implies that the algorithm should be able to solve the problem of MAP estimation in the presence of truncation (since $\theta_{ac}(f(a), f(c))$ would lie in the truncated part). Since MAP estimation in truncated convex models is an NP-hard problem, such an algorithm would not be computationally feasible unless P = NP.

Although the above argument shows that we will not be able to design an algorithm that provides a local minimum over smooth labelings, it serves to demonstrate the benefit of allowing each random variable to choose from a range of labels. Even though the range cannot be large enough to cover all smooth labelings, we should at least explore as large a subset of labelings as is computationally feasible. Clearly, this is an issue that is not considered in previous move making approaches. In order to alleviate this deficiency, we develop two algorithms that consider a large range of labels for each random variable. Table 3 describes the main steps involved in both the algorithms. The two methods differ in the way they move from one labeling to the next. In the next two sections, we provide a detailed description and analysis of our methods.

Initialization
• Initialize the labeling to some function f_1. For example, $f_1(a) = 0$ for all $v_a \in \mathbf{v}$.
Iteration
• Set $i_m = 0$ (where i_m indexes the interval to be used).
• While $i_m < h$
— Define interval $I_m = [i_m + 1, j_m]$ where $j_m = \min\{i_m + L, h - 1\}$ and $d(L) \geq M$.
— Move from current labeling f_m to a new labeling f_{m+1} using st-MINCUT such that
(i) if $f_{m+1}(a) \neq f_m(a)$ then $f_{m+1}(a) \in I_m$,
(ii) $Q(f_{m+1}, \mathbf{D}; \theta) \leq Q(f_m, \mathbf{D}; \theta)$.
— $i_m \leftarrow i_m + 1$.
Termination
• Stop if energy does not decrease for any interval I_m, otherwise repeat **Iteration**.

Table 3: As is typical with move making methods, our methods iteratively move from one labeling to the next by solving an st-MINCUT problem. They are said to converge when there remain no moves that reduce the energy further. The two algorithms, Range Swap and Range Expansion, differ in the way they choose the new labeling f_{m+1}. Specifically, they construct different graphs for the corresponding st-MINCUT problem.

5. The Range Swap Algorithm

Range Swap can be thought of as an appropriate modification of the αβ-swap algorithm of Boykov et al. (2001) for truncated convex models. At an iteration m, the Range Swap algorithm only considers the random variables v_a whose current labeling $f_m(a)$ lies in the interval $I_m = [i_m + 1, j_m]$ of length L.[1] In order to simplify the explanation of the algorithm, we will begin by assuming that $d(L) = M$ and later relax this condition such that $d(L) \geq M$. Keeping the labels of all other random variables fixed, Range Swap provides the option for random variables with $f_m(a) \in I_m$ to change their labels to $f_{m+1}(a) \in I_m$ (or retain their current label). In order to provide a concrete description of the algorithm, we define a set $S_m = \{a | f_m(a) \in I_m\}$. Using S_m we define the set of random variables $\mathbf{v}(S_m)$ and the set of edges $\mathcal{A}(S_m)$, $\mathcal{B}_1(S_m)$, $\mathcal{B}_2(S_m)$ and $\mathcal{B}(S_m)$ as follows:

$$
\begin{aligned}
&\mathbf{v}(S_m) = \{v_a | a \in S_m\}, \\
&\mathcal{A}(S_m) = \{(a,b) | (a,b) \in \mathcal{E}, a \in S_m, b \in S_m\}, \\
&\mathcal{B}_1(S_m) = \{(a,b) | (a,b) \in \mathcal{E}, a \in S_m, b \notin S_m\}, \\
&\mathcal{B}_2(S_m) = \{(a,b) | (a,b) \in \mathcal{E}, a \notin S_m, b \in S_m\}, \\
&\mathcal{B}(S_m) = \mathcal{B}_1(S_m) \bigcup \mathcal{B}_2(S_m).
\end{aligned} \tag{2}
$$

1. In what follows, we will assume that $j_m = i_m + L$ instead of $j_m = \min\{i_m + L, h - 1\}$. In other words, the length of the interval will always be L. However, all the arguments can be trivially extended to the case where the length of the interval is less than L.

At iteration m, the Range Swap algorithm moves from labeling f_m to f_{m+1} such that

$$\begin{aligned}
&Q(f_{m+1},\mathbf{D};\theta) \leq Q(f_m,\mathbf{D};\theta),\\
&f_{m+1}(a) \in I_m, \forall v_a \in \mathbf{v}(S_m),\\
&f_{m+1}(a) = f_m(a), \forall v_a \in \mathbf{v} - \mathbf{v}(S_m),
\end{aligned}$$

where $\mathbf{v} - \mathbf{v}(S_m)$ denotes all the variables that are not present in the set $\mathbf{v}(S_m)$. The new labeling f_{m+1} is obtained by constructing a graph such that every st-cut on the graph corresponds to a labeling f of the random variables that satisfies:

$$\begin{aligned}
&f(a) \in I_m, \forall v_a \in \mathbf{v}(S_m),\\
&f(a) = f_m(a), \forall v_a \in \mathbf{v} - \mathbf{v}(S_m).
\end{aligned}$$

The new labeling f_{m+1} is computed by solving for the minimum cost cut in this graph. We provide the details of the graph construction below.

5.1 Graph Construction

The Range Swap algorithm relies on a graph construction that is capable of exactly modeling arbitrary unary potentials and convex pairwise potentials. Such a graph construction was first proposed by Ishikawa (2003). As will be seen shortly, in this work we use a simpler graph construction (that does not require any *out-of-bounds* edges used in Ishikawa 2003). However, it is worth noting that the graph construction of Ishikawa (2003) may also be employed without affecting any property of the algorithm.

At each iteration of our algorithm, we are given an interval $I_m = [i_m + 1, j_m]$ of L labels (that is, $j_m = i_m + L$) where $d(L) = M$. We also have the current labeling f_m for all the random variables. We construct a directed weighted graph (with non-negative weights) $\mathcal{G}_m = \{\mathcal{V}_m, \mathcal{E}_m, c_m(\cdot,\cdot)\}$ such that for each $v_a \in \mathbf{v}(S_m)$, we define vertices $\{a_{i_m+1}, a_{i_m+2}, \cdots, a_{j_m}\} \in \mathcal{V}_m$. In addition, as is the case with every st-MINCUT problem, there are two additional vertices called terminals which we denote by s (the source) and t (the sink).

The edges $e \in \mathcal{E}_m$ with capacity (weight) $c_m(e)$ are defined to represent the following three types of potentials: (i) the unary potential $\theta_a(k)$ for random variable $v_a \in \mathbf{v}(S_m)$ taking the label k specified by an st-cut in the graph; (ii) the pairwise potential $\theta_{ab}(k, f_m(b))$ where $(a,b) \in \mathcal{B}_1(S_m)$ and the pairwise potential $\theta_{ab}(f_m(a), k)$ where $(a,b) \in \mathcal{B}_2(S_m)$, that is, pairwise potentials where one random variable is fixed to take its previous label; and (iii) the pairwise potential $\theta_{ab}(k,k')$ where $(a,b) \in \mathcal{A}(S_m)$, that is, pairwise potentials where no random variable is fixed to take its previous label. Note that all other potentials that specify the energy of the labeling are fixed during the iteration.

5.1.1 REPRESENTING UNARY POTENTIALS

For all random variables $v_a \in \mathbf{v}(S_m)$, we define the following edges that belong to the set $\mathcal{E}_m$:

- For all $k \in [i_m + 1, j_m)$, edges (a_k, a_{k+1}) have capacity $c_m(a_k, a_{k+1}) = \theta_a(k)$, that is, the cost of assigning label l_k to variable v_a.
- For all $k \in [i_m + 1, j_m)$, edges (a_{k+1}, a_k) have capacity $c_m(a_{k+1}, a_k) = \infty$.

- Edges (a_{j_m}, t) have capacity $c_m(a_{j_m}, t) = \theta_a(j_m)$.
- Edges (t, a_{j_m}) have capacity $c_m(t, a_{j_m}) = \infty$.
- Edges (s, a_{i_m+1}) have capacity $c_m(s, a_{i_m+1}) = \infty$.
- Edges (a_{i_m+1}, s) have capacity $c_m(a_{i_m+1}, s) = \infty$.

Fig. 2 shows the above edges together with their capacities for one random variable v_a. Note that there are two types of edges in the above set: (i) with finite capacity; and (ii) with infinite capacity. Any st-cut with finite cost contains only one of the finite capacity edges for each random variable v_a. This is because if an st-cut included more than one finite capacity edge, then by construction it must include at least one infinite capacity edge thereby making its cost infinite (Ishikawa, 2003). We interpret a finite cost st-cut as a relabeling of the random variables as follows:

$$f(a) = \begin{cases} k & \text{if } st\text{-cut includes edge } (a_k, a_{k+1}) \text{ where } k \in [i_m + 1, j_m), \\ j_m & \text{if } st\text{-cut includes edge } (a_{j_m}, t). \end{cases} \tag{3}$$

Note that the sum of the unary potentials for the labeling f is exactly equal to the cost of the st-cut over the edges defined above.

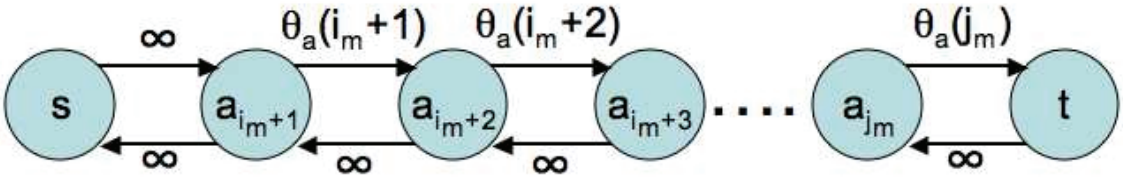


Figure 2: Part of the graph $\mathcal{G}_m$ containing the terminals and the vertices corresponding to the variable v_a. The edges that represent the unary potential of the new labeling are also shown.

5.1.2 REPRESENTING PAIRWISE POTENTIALS WITH ONE FIXED VARIABLE

We describe the case where $(a,b) \in \mathcal{B}_1(S_m)$. The other case where $(a,b) \in \mathcal{B}_2(S_m)$ can be handled similarly. Since $f_{m+1}(b)$ is fixed to $f_m(b)$, the pairwise potential $\theta_{ab}(i, f_{m+1}(b)) = \theta_{ab}(i, f_m(b))$ can be effectively treated as a unary potential of v_a. Hence, similar to unary potentials, it can be formulated using the following edge in set $\mathcal{E}_m$:

- For all $k \in [i_m + 1, j_m)$, edges (a_k, a_{k+1}) have capacity $c_m(a_k, a_{k+1}) = \theta_{ab}(k, f_m(b))$, that is, the cost of assigning label l_k to variable v_a and keeping the label of v_b fixed to $f_m(b)$.
- For all $k \in [i_m + 1, j_m)$, edges (a_{k+1}, a_k) have capacity $c_m(a_{k+1}, a_k) = \infty$.
- Edges (a_{j_m}, t) have capacity $c_m(a_{j_m}, t) = \theta_{ab}(j_m, f_m(b))$.
- Edges (t, a_{j_m}) have capacity $c_m(t, a_{j_m}) = \infty$.
- Edges (s, a_{i_m+1}) have capacity $c_m(s, a_{i_m+1}) = \infty$.
- Edges (a_{i_m+1}, s) have capacity $c_m(a_{i_m+1}, s) = \infty$.

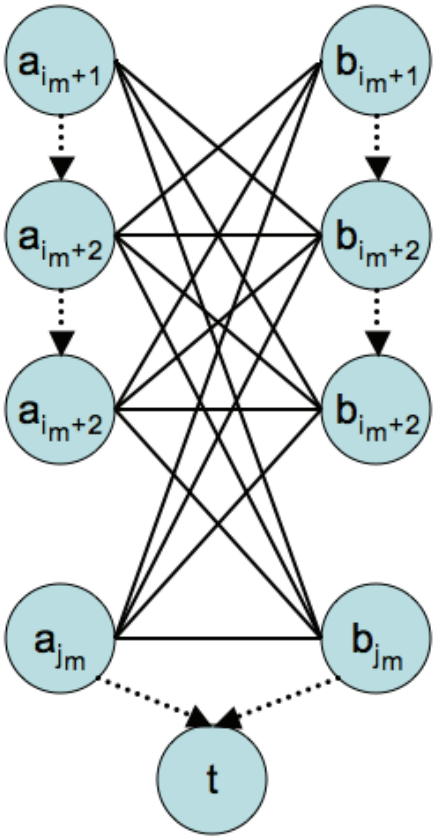


Figure 3: Edges that are used to represent the pairwise potentials of two neighboring random variables v_a and v_b such that $(a,b) \in \mathcal{A}(S_m)$ are shown. Undirected edges indicate that there are directed edges in both directions with equal capacity (as given by Equation 4). Directed dashed edges, with capacities shown in Equation (5), are added to ensure that the graph models the convex pairwise potentials correctly.

5.1.3 REPRESENTING PAIRWISE POTENTIALS WITH NO FIXED VARIABLES

For all random variables v_a and v_b such that $(a,b) \in \mathcal{A}(S_m)$, we define edges $(a_k, b_{k'}) \in \mathcal{E}_m$ where either one or both of k and k' belong to the set $(i_m+1, j_m]$ (that is, at least one of them is not i_m+1). The capacity of these edges is given by

$$c_m(a_k, b_{k'}) = \frac{w_{ab}}{2}\left(d(k-k'+1) - 2d(k-k') + d(k-k'-1)\right). \tag{4}$$

The RHS of the above equation is guaranteed to be non-negative due to the fact that $w_{ab} \geq 0$ and $d(\cdot)$ is convex. It is worth noting that, for the special cases when $d(\cdot)$ is linear or quadratic, the above capacity has a simple form. Specifically, when $d(\cdot)$ is linear the above capacity is equal to w_{ab} if $k = k'$ and 0 otherwise. When $d(\cdot)$ is quadratic the above capacity is a constant w_{ab} for all values of

k and k'. In addition to the edges described in Equation (4), we also specify the following edges:

$$c_m(a_k, a_{k+1}) = \frac{w_{ab}}{2}\left(d(L-k+i_m)+d(k-i_m)\right), \forall (a,b) \in \mathcal{E}, k \in [i_m+1, j_m),$$
$$c_m(b_{k'}, b_{k'+1}) = \frac{w_{ab}}{2}\left(d(L-k'+i_m)+d(k'-i_m)\right), \forall (a,b) \in \mathcal{E}, k' \in [i_m+1, j_m),$$
$$c_m(a_{j_m}, t) = c_m(b_{j_m}, t) = \frac{w_{ab}}{2} d(L), \forall (a,b) \in \mathcal{E}. \tag{5}$$

Fig. 3 provides an illustration of the above edges. The following Lemma shows that these edges model convex pairwise potentials exactly (up to an additive constant).

Lemma 1: For the capacities defined in Equations (4) and (5), the cost of the st-cut which includes the edges (a_k, a_{k+1}) and $(b_{k'}, b_{k'+1})$ (that is, v_a and v_b take labels l_k and $l_{k'}$ respectively) is given by $w_{ab}d(k-k') + \kappa_{ab}$, where the constant $\kappa_{ab} = w_{ab}d(L)$ (Proof in Appendix B).

This completes our graph construction. Given the graph $\mathcal{G}_m$ we solve the st-MINCUT problem, which provides us with a labeling f_{m+1} (using Equation (3)). We note that, since the cost of the st-cut exactly models the convex pairwise potential plus a constant, the above graph (together with the edges representing unary potentials) can be used to find the exact MAP estimate of the random field with convex pairwise potentials. In other words, ours is a somewhat modified, easy to follow graph construction for the method of Ishikawa (2003).

5.2 Generalizing Range Swap

In the previous subsection, we had assumed that the length of the interval L was chosen such that $d(L) = M$. We now relax this assumption such that $d(L) \geq M$. In this case, we define the set S_m such that

$$S_m = \{a | f_m(a) \in I_m, d(f_m(a), f_m(b)) \leq M, \forall (a,b) \in \mathcal{E}, f_m(b) \in I_m\}.$$

In other words, S_m consists of those random variables whose current label belongs to the interval I_m and whose pairwise potential with all its neighboring random variables v_b such that $f_m(b) \in I_m$ lies in the convex part of the truncated convex model. Using S_m the subset of random variables $\mathbf{v}(S_m)$ and the subset of edges $\mathcal{A}(S_m)$, $\mathcal{B}_1(S_m)$, $\mathcal{B}_2(S_m)$ and $\mathcal{B}(S_m)$ are defined as in Equation (2). The graph over which the st-MINCUT is performed is constructed as described in the previous subsection. As will be seen in § 5.4, the above definition of S_m would be useful in proving that the Range Swap algorithm monotonically improves the energy of the labeling from one iteration to the next.

5.3 Properties of the Graph

The following properties relating an st-cut with the corresponding labeling f hold true for the graph construction described in the previous subsection.

Property 1: The cost of the st-cut exactly represents the sum of the unary potentials for all variables in $\mathbf{v}(S_m)$, that is, $\sum_{v_a \in \mathbf{v}(S_m)} \theta_a(f(a))$.

Property 2: For $(a,b) \in \mathcal{B}_1(S_m)$, the cost of the st-cut exactly represents the pairwise potential $\theta_{ab}(f(a), f_m(b))$. Similarly, for $(a,b) \in \mathcal{B}_2(S_m)$, the cost of the st-cut exactly represents the pairwise potential $\theta_{ab}(f_m(a), f(b))$.

Property 3: For $(a,b) \in \mathcal{A}(S_m)$, if $f(a) \in I_m$ and $f(b) \in I_m$ such that

$$d(f(a) - f(b)) \leq M,$$

then the cost of the *st*-cut exactly represents the pairwise potential $\theta_{ab}(f(a), f(b))$ plus a constant κ_{ab}, that is,

$$w_{ab}d(f(a) - f(b)) + \kappa_{ab}.$$

This property follows directly from Lemma 1.

Property 4: For $(a,b) \in \mathcal{A}(S_m)$, if $f(a) \in I_m$ and $f(b) \in I_m$ such that

$$d(f(a) - f(b)) > M,$$

then the cost of the *st*-cut incorrectly represents the pairwise potential $\theta_{ab}(f(a), f(b))$, being

$$w_{ab}d(f(a) - f(b)) + \kappa_{ab},$$

which is an overestimation of the correct value (that is, $w_{ab}M$ plus the constant κ_{ab}). This follows from the fact that our graph construction overestimates the truncation part by the convex function $w_{ab}d(\cdot)$.

In summary, property 1 tells us that the cost of the *st*-cut exactly models the sum of the unary potentials. Properties 2 and 3 specify the cases where the cost of the *st*-cut exactly models the pairwise potentials, while property 4 specifies the remaining case where the cost of the *st*-cut overestimates the pairwise potentials. Since the potentials are either modeled exactly or are overestimated, it follows that the energy of the labeling f_{m+1} is less than or equal to the cost of the *st*-MINCUT on $\mathcal{G}_m$. The only free parameter in the Range Swap algorithm is the length of the interval L. Next, we discuss how to choose the value of this parameter.

5.4 Length of the Interval

We begin by considering the case when L satisfies $d(L) = M$. Note that in this case, property 4 no longer needs to be considered. This implies that the cost of the *st*-cut exactly models the energy of the corresponding labeling. Hence, the *st*-MINCUT provides the optimal move f_{m+1}. Next, we consider the case when the length of the interval satisfies $d(L) > M$. We show that this interval provides a labeling that is at least as good as the labeling obtained by considering any of its subsets for which the optimal move can be computed. Formally, let f_{m+1} be the labeling obtained by using an interval of length L such that $d(L) > M$ and let f'_{m+1} be the labeling obtained by using a subset of the interval of length L' such that $d(L') = M$. Then the following holds true.

Observation 1: The energy of f_{m+1} is less than or equal to the energy of f'_{m+1}.

Proof: When we use the interval of length L, one of the cuts in the graph would correspond to f'_{m+1}. Since $d(L') = M$, it follows that the cost of the cut would be equal to $Q(f'_{m+1}, \mathbf{D}; \theta)$. Furthermore, the cost of the cut corresponding to f_{m+1} is at least equal to $Q(f_{m+1}, \mathbf{D}; \theta)$. Using the fact that f_{m+1} corresponds to the minimum cost cut, we see that

$$Q(f_{m+1}, \mathbf{D}; \theta) \leq Q(f'_{m+1}, \mathbf{D}; \theta).$$

∎

The above observation shows that we do not lose any accuracy by considering non-optimal moves on large intervals (compared to optimal moves on smaller subsets of the interval). However, the larger the value of L the bigger the corresponding graph on which we need to compute the *st*-MINCUT. Thus, in practice the value of L should be chosen according to the available computational resources.

5.5 Analysis of Range Swap

Regardless of whether $d(L) = M$ (that is, the Range Swap algorithm described in § 5.1) or $d(L) > M$ (its generalization described in § 5.2), it is worth noting that the corresponding graph construction ensures that the cut corresponding to the labeling f_m exactly models the energy $Q(f_m, \mathbf{D}; \theta)$ up to a constant. This implies that the energy of the new labeling f_{m+1} is less than or equal to the energy of f_m, that is,

$$Q(f_{m+1}, \mathbf{D}; \theta) \leq Q(f_m, \mathbf{D}; \theta).$$

This follows from the fact that the cost of the *st*-MINCUT is less than or equal to the energy of the labeling f_m but is greater than or equal to the energy of f_{m+1}. In other words, the Range Swap algorithm monotonically improves the energy of the labeling from one iteration to the next.

It is worth noting that, unlike previous move making algorithms, Range Swap is not guaranteed to compute the optimal move other than in the special case when $d(L) = M$ (where $L = j_m - i_m$ is the length of the interval). In other words, for the case where $d(L) > M$, if in the m^{th} iteration we move from label f_m to f_{m+1} then it is possible that there exists another labeling f'_{m+1} such that

$$\begin{aligned}
&Q(f'_{m+1}, \mathbf{D}; \theta) \leq Q(f_{m+1}, \mathbf{D}; \theta), \\
&f'_{m+1}(a) \in I_m, \forall v_a \in \mathbf{v}(S_m), \\
&f'_{m+1}(a) = f_m(a), \forall v_a \in \mathbf{v} - \mathbf{v}(S_m).
\end{aligned}$$

This is due to the fact that the graph construction overestimates certain pairwise potentials (see Property 4). However, as Observation 1 shows, the improvement in the energy obtained by a (potentially non-optimal) move when $d(L) > M$ is at least as much as the improvement obtained by the optimal move when $d(L) = M$.

6. The Range Expansion Algorithm

Range Expansion is a suitable modification of the α-expansion algorithm of Boykov et al. (2001) for truncated convex models. Unlike Range Swap, at an iteration m it considers all the random variables v_a regardless of whether their current labeling $f_m(a)$ lies in the interval I_m. It provides the option for each random variable v_a to either retain its old label $f_m(a)$ or change its label to $f_{m+1}(a) \in I_m$. Formally, the Range Expansion algorithm moves from labeling f_m to f_{m+1} such that

$$\begin{aligned}
&Q(f_{m+1}, \mathbf{D}; \theta) \leq Q(f_m, \mathbf{D}; \theta), \\
&f_{m+1}(a) = f_m(a) \text{ OR } f_{m+1}(a) \in I_m, \forall v_a \in \mathbf{v}.
\end{aligned}$$

Similar to Range Swap, Range Expansion does not compute the optimal move at each iteration. In other words, if in the m^{th} iteration we move from label f_m to f_{m+1} then it is possible that there exists another labeling f'_{m+1} such that

$$\begin{aligned}
&Q(f'_{m+1}, \mathbf{D}; \theta) < Q(f_{m+1}, \mathbf{D}; \theta), \\
&f'_{m+1}(a) = f_m(a) \text{ OR } f'_{m+1}(a) \in I_m, \forall v_a \in \mathbf{v}.
\end{aligned}$$

However, our analysis in § 6.3 shows that we will still be able to reduce the energy sufficiently at each iteration so as to obtain the best known multiplicative bounds upon convergence. As in the

case of Range Swap, we move from labeling f_m to f_{m+1} by constructing a graph such that every *st*-cut on the graph corresponds to a labeling f of the random variables that satisfies:

$$f(a) = f_m(a) \text{ OR } f(a) \in I_m, \forall v_a \in \mathbf{v}.$$

The new labeling f_{m+1} is obtained in two steps: (i) we obtain a labeling f that corresponds to the *st*-MINCUT on our graph; and (ii) we choose the new labeling f_{m+1} as

$$f_{m+1} = \begin{cases} f & \text{if } Q(f,\mathbf{D};\theta) \leq Q(f_m,\mathbf{D};\theta), \\ f_m & \text{otherwise.} \end{cases} \tag{6}$$

Note that, unlike Range Swap, step (ii) is required in Range Expansion since the labeling f obtained in step (i) may have greater energy than f_m. This is due to the approximations involved in the graph construction described below.

6.1 Graph Construction

We construct a directed weighted graph (with non-negative weights) $\mathcal{G}_m = \{\mathcal{V}_m, \mathcal{E}_m, c_m(\cdot,\cdot)\}$ such that $\mathcal{V}_m$ contains the source s, the sink t and the vertices $\{a_{i_m+1}, a_{i_m+2}, \cdots, a_{j_m}\}$ for each random variable $v_a \in \mathbf{v}$. The edges $e \in \mathcal{E}_m$ with capacity $c_m(e)$ are of two types: (i) those that represent the unary potentials of a labeling corresponding to an *st*-cut in the graph and; (ii) those that represent the pairwise potentials of the labeling.

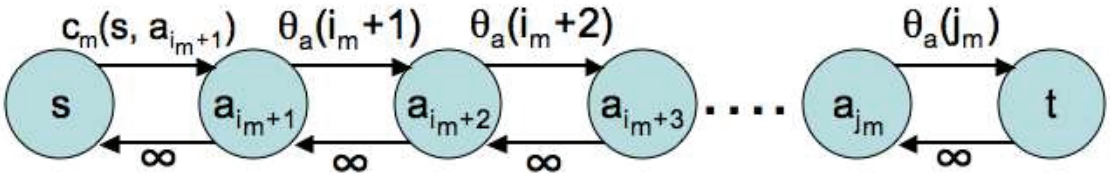


Figure 4: Part of the graph $\mathcal{G}_m$ containing the terminals and the vertices corresponding to the variable v_a. The edges that represent the unary potential of the new labeling are also shown. The term $c_m(s, a_{i_m+1})$ is defined in Equation (7).

6.1.1 REPRESENTING UNARY POTENTIALS

The unary potentials are represented in a similar manner to the graph construction used in Range Swap. The notable difference is that now we have to model the unary potential for the case when a variable v_a retains its old label that does not lie in the interval I_m. To this end, we change the capacity of the edge (s, a_{i_m+1}) to

$$c_m(s, a_{i_m+1}) = \begin{cases} \theta_a(f_m(a)) & \text{if } f_m(a) \notin I_m, \\ \infty & \text{otherwise.} \end{cases} \tag{7}$$

Fig. 4 shows all the edges specified for representing the unary potential of one random variable v_a. We interpret a finite cost *st*-cut as a relabeling of the random variables as follows:

$$f(a) = \begin{cases} k & \text{if } st\text{-cut includes edge } (a_k, a_{k+1}) \text{ where } k \in [i_m+1, j_m), \\ j_m & \text{if } st\text{-cut includes edge } (a_{j_m}, t), \\ f_m(a) & \text{if } st\text{-cut includes edge } (s, a_{i_m+1}). \end{cases} \tag{8}$$

Note that the sum of the unary potentials for the labeling f is exactly equal to the cost of the st-cut over the edges defined above.

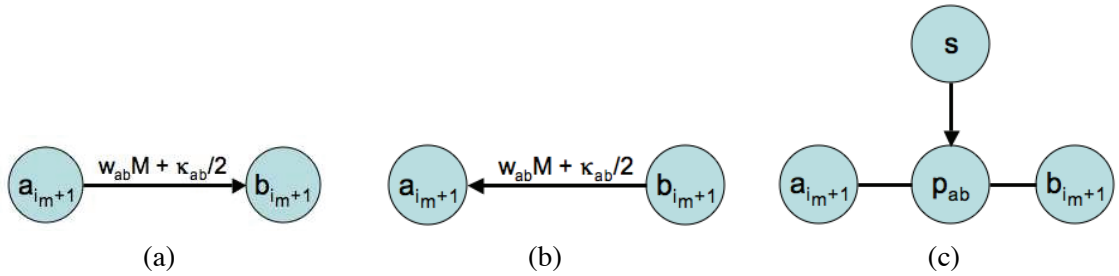


Figure 5: Additional edges that are added to the graph shown in Fig. 3 for representing pairwise potentials. **(a)** When $f_m(a) \in I_m$ and $f_m(b) \notin I_m$. Here, $\kappa_{ab} = w_{ab}d(L)$. **(b)** When $f_m(a) \notin I_m$ and $f_m(b) \in I_m$. **(c)** When $f_m(a) \notin I_m$ and $f_m(b) \notin I_m$. Undirected edges indicate the presence of opposing edges with equal capacity. The capacities of all five edges are specified in Equation (9).

6.1.2 REPRESENTING PAIRWISE POTENTIALS

For each pair of neighboring random variables $(a,b) \in \mathcal{E}$ we will use the edges defined for the graph of Range Swap for representing pairwise potentials, that is, all the edges shown in Fig. 3. However, we also have to consider the cases where at least one of the neighboring random variables retains its previous label and that label is not present in the interval I_m. In order to model these cases, we incorporate the following additional edges:

- If $f_m(a) \in I_m$ and $f_m(b) \notin I_m$ then we add an edge (a_{i_m+1}, b_{i_m+1}) with capacity $w_{ab}M + \kappa_{ab}/2$ (see Fig. 5(a)).

- If $f_m(a) \notin I_m$ and $f_m(b) \in I_m$ then we add an edge (b_{i_m+1}, a_{i_m+1}) with capacity $w_{ab}M + \kappa_{ab}/2$ (see Fig. 5(b)).

- If $f_m(a) \notin I_m$ and $f_m(b) \notin I_m$, we introduce a new vertex p_{ab}.[2] Using this vertex p_{ab}, five edges are defined with the following capacities (see Fig. 5(c)):

$$
\begin{aligned}
c_m(a_{i_m+1}, p_{ab}) &= c_m(p_{ab}, a_{i_m+1}) = w_{ab}M + \kappa_{ab}/2, \\
c_m(b_{i_m+1}, p_{ab}) &= c_m(p_{ab}, b_{i_m+1}) = w_{ab}M + \kappa_{ab}/2, \\
c_m(s, p_{ab}) &= \theta_{ab}(f_m(a), f_m(b)) + \kappa_{ab}.
\end{aligned}
$$

This completes our graph construction. Given the graph $\mathcal{G}_m$ we solve the st-MINCUT problem, which provides us with a labeling f as described in Equation (8). The new labeling f_{m+1} is obtained using Equation (6).

2. We note here that an equivalent graph can be constructed without adding the vertex p_{ab} using the method of Schlesinger and Flach (2006). However, the vertex p_{ab} helps make the analysis easier.

6.2 Properties of the Graph

We now describe the properties of the above graph construction, with the aim of facilitating the analysis of our algorithm for the case of truncated linear and truncated quadratic models.

Property 5: The cost of the *st*-cut exactly represents the sum of the unary potentials associated with the corresponding labeling f, that is, $\sum_{v_a \in \mathbf{v}} \theta_a(f(a))$.

Property 6: For $(a,b) \in \mathcal{E}$, if $f(a) = f_m(a) \notin I_m$ and $f(b) = f_m(b) \notin I_m$ then the cost of the *st*-cut exactly represents the pairwise potential $\theta_{ab}(f(a), f(b))$ plus a constant κ_{ab}. This is due to the fact that the *st*-cut contains the edge (s, p_{ab}) whose capacity is $\theta_{ab}(f_m(a), f_m(b)) + \kappa_{ab}$. Note that in this case p_{ab} belongs to the partition containing the sink t. This can be easily verified by observing that the cost of the *st*-cut would increase if p_{ab} belonged to the partition containing the source s (since this would include edge (p_{ab}, a_{i_m+1}) and (p_{ab}, b_{i_m+1}) in the *st*-cut).

Property 7: For $(a,b) \in \mathcal{E}$, if $f(a) \in I_m$ and $f(b) \in I_m$ such that

$$d(f(a) - f(b)) \leq M,$$

then the cost of the *st*-cut exactly represents the pairwise potential $\theta_{ab}(f(a), f(b))$ plus a constant κ_{ab}, that is,

$$w_{ab} d(f(a) - f(b)) + \kappa_{ab}.$$

This follows from the fact that in this case the pairwise potential lies in the convex part of the truncated convex model, which is modeled exactly (see Lemma 1).

Property 8: For $(a,b) \in \mathcal{E}$, if $f(a) \in I_m$ and $f(b) \in I_m$ such that

$$d(f(a) - f(b)) > M,$$

then the cost of the *st*-cut incorrectly represents the pairwise potential $\theta_{ab}(f(a), f(b))$, being

$$w_{ab} d(f(a) - f(b)) + \kappa_{ab},$$

which is an overestimation of the correct value (that is, $w_{ab}M$ plus the constant κ_{ab}). This follows from the fact that our graph construction overestimates the truncation part by the convex function $w_{ab} d(\cdot)$ (see Lemma 1).

Property 9: For $(a,b) \in \mathcal{E}$, if $f(a) \in I_m$ and $f(b) = f_m(b) \notin I_m$ then the cost of the *st*-cut incorrectly represents the pairwise potential $\theta_{ab}(f(a), f(b))$, being

$$w_{ab} d(f(a) - (i_m + 1)) + w_{ab} \hat{d}(f(a) - (i_m + 1)) + w_{ab} M + \kappa_{ab}, \tag{9}$$

where $\hat{d}(\cdot)$ denotes the following function:

$$\hat{d}(x) = d(x+1) - d(x) - d(1) + \frac{d(0)}{2}, \forall x \geq 0. \tag{10}$$

Note that $\hat{d}(\cdot)$ is only defined for a non-negative argument. Clearly, the argument of $\hat{d}(\cdot)$ in Equation (9) is non-negative since $f(a) \in [i_m + 1, j_m]$. The function $\hat{d}(x) = 0$ when $d(\cdot)$ is a linear metric and $\hat{d}(x) = 2x$ when $d(\cdot)$ is the quadratic semi-metric.

Similarly, if $f(a) = f_m(a) \notin I_m$ and $f(b) \in I_m$ then the cost of the *st*-cut incorrectly represents the pairwise potential $\theta_{ab}(f(a), f(b))$, being

$$w_{ab} d(f(b) - (i_m + 1)) + w_{ab} \hat{d}(f(b) - (i_m + 1)) + w_{ab} M + \kappa_{ab}.$$

The above property can be shown to be true using the following Lemma.

Lemma 2: For the graph described in § 6.1, property 9 holds true (Proof in Appendix C).

In summary, property 5 tells us that the cost of the st-cut exactly models the sum of the unary potentials. Properties 6 and 7 specify the cases where the cost of the st-cut exactly models the pairwise potentials, while properties 8 and 9 specify the remaining cases where the cost of the st-cut overestimates the pairwise potentials. In other words, the energy of the labeling f, and hence the energy of f_{m+1}, is less than or equal to the cost of the st-MINCUT on $\mathcal{G}_m$.

Note that our graph construction is similar to that of Gupta and Tardos (2000) with two notable exceptions: (i) we can handle any general truncated convex model and not just truncated linear as in the case of Gupta and Tardos (2000); and (ii) we have the freedom to choose the value of L, while Gupta and Tardos (2000) fixed this value to M. A logical choice would be to use that value of L that minimizes the worst case multiplicative bound for a particular class of problems. The following analysis obtains the desired value of L for both the truncated linear and the truncated quadratic models. Our worst case multiplicative bounds are exactly those achieved by the LP relaxation (see Chekuri et al., 2005).

6.3 Multiplicative Bounds

In order to obtain multiplicative bounds for the Range Expansion algorithm, we will make use of the fact that the algorithm only terminates once we are unable to reduce the energy for any interval I_m. In other words, we stop once we have reached the local minimum of the large neighborhood defined by the intervals. We exploit this fact in the following manner. First, we establish a lower bound on how much the energy is reduced for a given interval (see Lemma 3 below). To this end, we extensively use the properties of the graph described in the previous subsection. As our final labeling f is a local minimum over the intervals, it follows that once the algorithm terminates the above mentioned lower bound will be less than or equal to zero (otherwise it would be possible to reduce the energy further). This observation provides us with an expression for the upper bound of the energy of f. Next, we simplify this expression for both truncated linear metric (see Theorem 2) and truncated quadratic semi-metric (see Theorem 3) and show that our bounds match those of the LP relaxation.

Before we proceed with the details, we require the following definitions. Let $r \in [0, L-1]$ be a uniformly distributed random integer. Using r we define the following set of intervals

$$I_r = \{[0,r],[r+1,r+L],[r+L+1,r+2L],\cdots,[.,h-1]\},$$

where $h = |\mathbf{l}|$ is the total number of labels associated with the MRF. We denote an optimal labeling of the MRF by f^*. Given such a labeling f^* and an interval $I_m = [i_m+1, j_m] \in I_r$, we define the following sets:

$$\begin{aligned}
&\mathbf{v}(f^*,I_m) = \{v_a | v_a \in \mathbf{v}, f^*(a) \in I_m\},\\
&\mathcal{A}(f^*,I_m) = \{(a,b) | (a,b) \in \mathcal{E}, f^*(a) \in I_m, f^*(b) \in I_m\},\\
&\mathcal{B}_1(f^*,I_m) = \{(a,b) | (a,b) \in \mathcal{E}, f^*(a) \in I_m, f^*(b) \notin I_m\},\\
&\mathcal{B}_2(f^*,I_m) = \{(a,b) | (a,b) \in \mathcal{E}, f^*(a) \notin I_m, f^*(b) \in I_m\},\\
&\mathcal{B}(f^*,I_m) = \mathcal{B}_1(f^*,I_m) \cup \mathcal{B}_2(f^*,I_m).
\end{aligned}$$

In other words, $\mathbf{v}(f^*,I_m)$ contains all the random variables that take an optimal labeling in I_m, $\mathcal{A}(f^*,I_m)$ contains the set of all edges in the graphical model of the MRF whose endpoints take an

optimal labeling in the interval I_m, and $\mathcal{B}(f^*, I_m)$ contains edges where only one endpoint takes an optimal labeling in I_m.

Clearly, the following equation holds true:

$$\sum_{v_a \in \mathbf{v}} \theta_a(f^*(a)) = \sum_{I_m \in I_r} \sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f^*(a)), \tag{11}$$

since $f^*(a)$ belongs to exactly one interval in I_r for all $v_a \in \mathbf{v}$. In order to make the analysis less cluttered, we introduce the following shorthand notation for some terms:

- For $(a,b) \in \mathcal{A}(f^*, I_m)$, we denote $w_{ab} d(f^*(a) - f^*(b))$ by e^m_{ab}.
- For $(a,b) \in \mathcal{B}_1(f^*, I_m)$, we denote $w_{ab} d(f^*(a) - (i_m + 1)) + w_{ab} \hat{d}(f^*(a) - (i_m + 1)) + w_{ab} M$ by e^m_a.
- For $(a,b) \in \mathcal{B}_2(f^*, I_m)$, we denote $w_{ab} d(f^*(b) - (i_m + 1)) + w_{ab} \hat{d}(f^*(b) - (i_m + 1)) + w_{ab} M$ by e^m_b.

We are now ready to prove our main results, starting with the following Lemma.

Lemma 3: At an iteration of our algorithm, given the current labeling f_m and an interval $I_m = [i_m + 1, j_m]$, the new labeling f_{m+1} obtained by solving the *st*-MINCUT problem reduces the energy by at least the following:

$$\sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f_m(a)) + \sum_{(a,b) \in \mathcal{A}(f^*, I_m) \bigcup \mathcal{B}(f^*, I_m)} \theta_{ab}(f_m(a), f_m(b))$$
$$- \left(\sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f^*(a)) + \sum_{(a,b) \in \mathcal{A}(f^*, I_m)} e^m_{ab} + \sum_{(a,b) \in \mathcal{B}_1(f^*, I_m)} e^m_a + \sum_{(a,b) \in \mathcal{B}_2(f^*, I_m)} e^m_b \right).$$

Here f^* refers to an optimal labeling for the given MRF (Proof in Appendix D).

Let f be the final labeling obtained using our algorithm. Since f is a local optimum with respect to all intervals I_m, it follows that the above term should be non-positive for all I_m (otherwise the energy could be further reduced thereby contradicting the fact that f is a local optimum labeling). In other words,

$$\sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f(a)) + \sum_{(a,b) \in \mathcal{A}(f^*, I_m) \bigcup \mathcal{B}(f^*, I_m)} \theta_{ab}(f(a), f(b))$$
$$\leq \left(\sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f^*(a)) + \sum_{(a,b) \in \mathcal{A}(f^*, I_m)} e^m_{ab} + \sum_{(a,b) \in \mathcal{B}_1(f^*, I_m)} e^m_a + \sum_{(a,b) \in \mathcal{B}_2(f^*, I_m)} e^m_b \right), \forall I_m.$$

We sum the above inequality over all $I_m \in I_r$. The summation of the LHS is at least $Q(f, \mathbf{D}; \theta)$. Furthermore, using Equation (11), the summation of the above inequality can be written as

$$Q(f, \mathbf{D}; \theta) \leq \sum_{v_a \in \mathbf{v}} \theta_a(f^*(a)) +$$
$$\sum_{I_m \in I_r} \left(\sum_{(a,b) \in \mathcal{A}(f^*, I_m)} e^m_{ab} + \sum_{(a,b) \in \mathcal{B}_1(f^*, I_m)} e^m_a + \sum_{(a,b) \in \mathcal{B}_2(f^*, I_m)} e^m_b \right).$$

We now take the expectation of the above inequality over the uniformly distributed random integer $r \in [0, L-1]$. The LHS of the inequality and the first term on the RHS (that is, $\sum \theta_a(f^*(a))$) are constants with respect to r. Hence, we get

$$Q(f,\mathbf{D};\theta) \leq \sum_{v_a \in \mathbf{v}} \theta_a(f^*(a)) + \frac{1}{L}\sum_r \sum_{I_m \in I_r} \left(\sum_{(a,b)\in \mathcal{A}(f^*,I_m)} e^m_{ab} + \sum_{(a,b)\in \mathcal{B}_1(f^*,I_m)} e^m_a + \sum_{(a,b)\in \mathcal{B}_2(f^*,I_m)} e^m_b \right). \quad (12)$$

We conclude by observing that this is the same bound that is obtained by the LP relaxation. Thus, using the analysis of Chekuri et al. (2005) we obtain the following results.

Lemma 4: When $d(\cdot)$ is linear, that is, $d(x) = |x|$, the following inequality holds true:

$$\frac{1}{L}\sum_r \sum_{I_m \in I_r} \left(\sum_{(a,b)\in \mathcal{A}(f^*,I_m)} e^m_{ab} + \sum_{(a,b)\in \mathcal{B}_1(f^*,I_m)} e^m_a + \sum_{(a,b)\in \mathcal{B}_2(f^*,I_m)} e^m_b \right) \leq \left(2 + \max\left\{\frac{2M}{L}, \frac{L}{M}\right\}\right) \sum_{(a,b)\in\mathcal{E}} \theta_{ab}(f^*(a)f^*(b)).$$

(Proof in Appendix E).

Theorem 2: For the truncated linear metric, our algorithm obtains a multiplicative bound of $2+\sqrt{2}$ using $L = \sqrt{2}M$.

The proof of the above theorem follows by substituting $L = \sqrt{2}M$ in the above inequality and simplifying inequality (12). Note that this bound is better than those obtained by α-expansion (Boykov et al., 2001) ($2M$) and its generalization (Gupta and Tardos, 2000) (4). In fact, the bound of Gupta and Tardos (2000) can be obtained directly from the above analysis by using the non-optimal assignment of $L = M$.

Similarly, using Theorem 4 of Chekuri et al. (2005), we obtain the following multiplicative bound for the truncated quadratic semi-metric.

Theorem 3: For the truncated quadratic semi-metric, our algorithm obtains a multiplicative bound of $O(\sqrt{M})$ using $L = \sqrt{M}$.

Note that both α-expansion and the approach of Gupta and Tardos provide no bounds for the above case. The primal-dual method of Komodakis and Tziritas (2007) obtains a bound of $2M$, which is clearly inferior to our guarantees. Finally, we note that a slight modification of Theorem 3.7 of Gupta and Tardos (2000) shows that the above guarantees can be obtained in a polynomial number of iterations. Since each iteration itself is of polynomial complexity, it follows that the Range Expansion algorithm provides LP multiplicative bounds on polynomial time.

Theorem 4: If the Range Expansion algorithm is run for $O(h/L)(\log Q(f_1,\mathbf{D};\theta) + \log \varepsilon^{-1})$ iterations (where f_1 is the initial labeling, and $\varepsilon > 0$), then the expected value of the energy would be at most $(2+\sqrt{2}+\varepsilon)Q(f^*,\mathbf{D};\theta)$ for the truncated linear metric and $(O(\sqrt{M})+\varepsilon)Q(f^*,\mathbf{D};\theta)$ for the truncated quadratic semi-metric (where f^* is an optimal labeling).

Although theoretically interesting, the practical implications of this result are minimal since in most scenarios we will be able to run our methods for a sufficient number of iterations so as to end up in a local minimum over all intervals I_m. For instance, in all our experiments we reached a local minimum in less than 5 iterations.

7. Experiments

We tested both our range move algorithms using both synthetic and standard real data, and compared it with several state of the art methods. We do not include a comparison with interior point algorithms for the primal LP due to their high computation cost.

7.1 Synthetic Data

We begin with experiments that use synthetically generated data. In the following, we provide the details of the experimental setup and present the results obtained.

7.1.1 EXPERIMENTAL SETUP

We used 100 random fields for both the truncated linear and truncated quadratic models. The variables $\mathbf{v}$ and neighborhood relationship $\mathcal{E}$ of the random fields described a 4-connected grid graph of size 50×50. Note that 4-connected grid graphs are widely used to model several problems in computer vision (Szeliski et al., 2008). Each variable was allowed to take one of 20 possible labels, that is, $\mathbf{l} = \{l_0, l_1, \cdots, l_{19}\}$. The parameters of the random field were generated randomly. Specifically, the unary potentials $\theta_a(i)$ were sampled uniformly from the interval $[0, 10]$ while the weights w_{ab}, which determine the pairwise potentials, were sampled uniformly from $[0, 5]$. The parameter M was also chosen randomly while taking care that $d(5) \leq M \leq d(10)$.

7.1.2 RESULTS

Fig. 6 shows the results obtained by our methods and four other state of the art algorithms: $\alpha\beta$-swap, α-expansion, BP and TRW-S. We used publicly available code for all previously proposed approaches.[3] As can be seen from the figure, the most accurate move making approaches are the methods proposed in this paper. As expected, both our algorithms are slower than $\alpha\beta$-swap and α-expansion (since each iteration computes an *st*-MINCUT on a larger graph). However, they are faster than TRW-S, which attempts to minimize the LP relaxation, and BP. We note here that our implementation does not use any clever tricks to speed up the max-flow algorithm (such as those described by Alahari et al., 2008) that can potentially decrease the running time by orders of magnitude.

7.2 Real Data - Stereo Reconstruction

Given two *epipolar rectified* images $\mathbf{D}_1$ and $\mathbf{D}_2$ of the same scene, the problem of stereo reconstruction is to obtain a correspondence between the pixels of the images. This problem can be modeled using a random field whose variables correspond to pixels of one image (say $\mathbf{D}_1$) and take labels from a set of *disparities* $\mathbf{l} = \{0, 1, \cdots, h-1\}$. A disparity value i for a random variable a denoting pixel (x, y) in $\mathbf{D}_1$ indicates that its corresponding pixel lies in location $(x+i, y)$ in the second image.

For the above random field formulation, the unary potentials were obtained using the method described by Birchfield and Tomasi (1998) and were truncated at 15. As is typically the case, we chose the neighborhood relationship $\mathcal{E}$ to define a 4-neighborhood grid graph. The number of

3. When using α-expansion with the truncated quadratic semi-metric, all edges with negative capacities in the graph construction were removed, similar to the experiments in Szeliski et al. (2008).

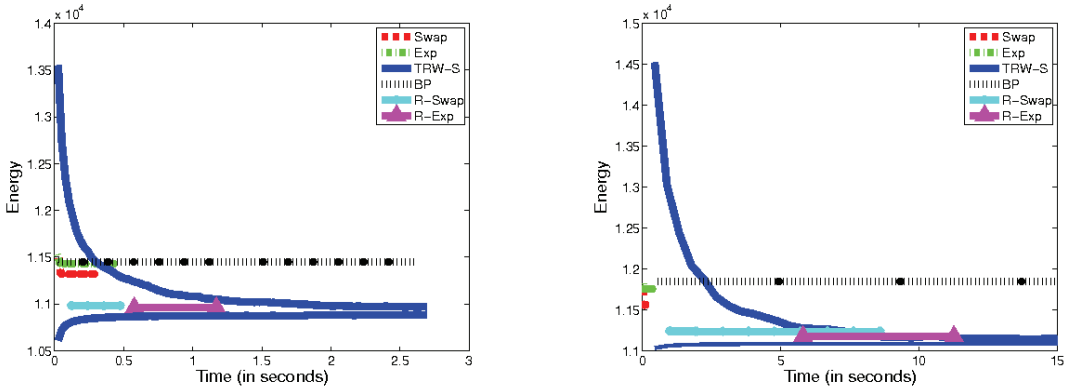


Figure 6: Results of the synthetic experiment. **(a)** Truncated linear metric. **(b)** Truncated quadratic semi-metric. The x-axis shows the time taken in seconds. The y-axis shows the average energy obtained over all 100 random fields using the six algorithms. The lower blue curve is the value of the dual obtained by TRW-S. In both the cases, our methods provide more accurate solutions than previous move making algorithms ($\alpha\beta$-swap and α-expansion) and are faster than message passing approaches (TRW-S and BP). The labelings obtained by our methods always have a lower energy than those obtained by BP and are comparable to the energy obtained using TRW-S.

disparities h was set to 20. We experimented using the following truncated convex potentials:

$$\theta_{ab}(i,j) = 50\min\{|i-j|, 10\},$$
$$\theta_{ab}(i,j) = 50\min\{(i-j)^2, 100\}.$$

The above form of pairwise potentials encourage neighboring pixels to take similar disparity values, which corresponds to our expectations of finding smooth surfaces in natural images. Truncation of pairwise potentials is essential to avoid over smoothing, as observed in Boykov et al. (2001). Note that using spatially varying weights w_{ab} provides better results. However, the main aim of this experiment is to demonstrate the accuracy and speed of our approach and not to design the best possible energy. Fig. 7 shows the results obtained using various algorithms when using the truncated linear metric on a standard stereo pair (Tsukuba). Table 4 provides the value of the energy and the total time taken by all the approaches for three stereo pairs. Similar to the synthetic experiments, the range move algorithms provide accurate solutions while taking less time than TRW-S and BP. Range expansion does marginally better than range swap but is computationally more expensive.

8. Concluding Remarks

Summary. We proposed the Range Swap and Range Expansion algorithms for obtaining an approximate MAP estimate of discrete random fields with truncated convex pairwise potentials. Our methods consider a range of labels at each iteration and hence, explore a larger search space compared to previous *st*-MINCUT based approaches. Due to the use of only the *st*-MINCUT problem

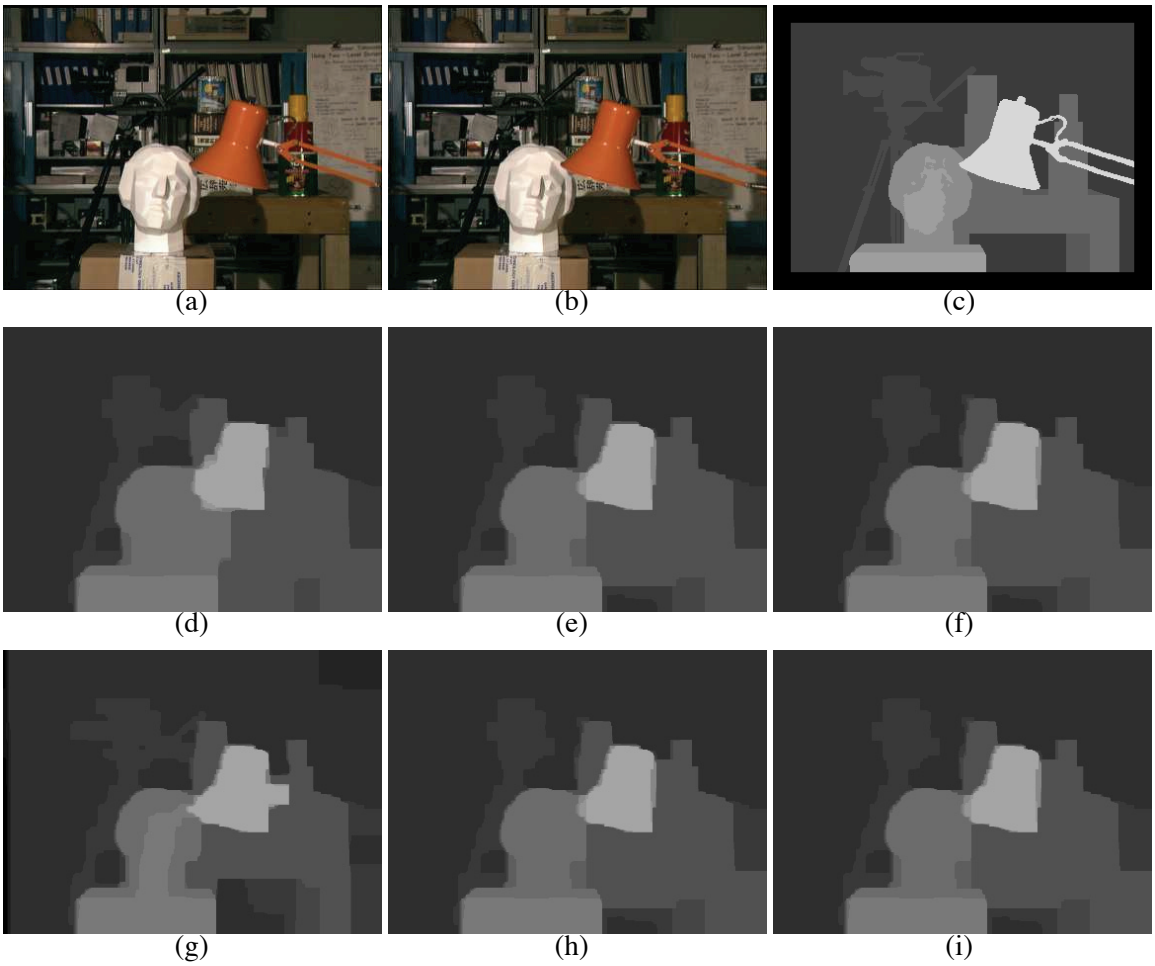


Figure 7: Tsukuba stereo pair. **(a)** First image. **(b)** Second image. **(c)** Ground truth disparity map. **(d)-(i)** Results obtained using various algorithms: in the above order, $\alpha\beta$-swap algorithm, α-expansion, TRW-S, BP, range swap and range expansion.

in their design, both the methods are faster than previous message passing approaches such as BP and TRW-S. Experiments on synthetic and real data problems demonstrate the effectiveness of our methods compared to several state of the art algorithms.

The two algorithms differ in the *st*-MINCUT problem that they solve at each iteration to move from one labeling to the next. The Range Swap algorithm guarantees that at each iteration the energy of the new labeling obtained by the *st*-MINCUT algorithm is less than or equal to the energy of the previous labeling. However, this monotonic improvement in the energy comes at the price of considering only a subset of the random variables at each iteration. In practice, solving the smaller problem (defined on a subset of random variables) at each iteration makes the Range Swap algorithm computationally efficient. In contrast, the graph construction employed by the Range Expansion algorithm does not guarantee a monotonic improvement in the energy. In other words, the new

Algorithm	Energy-1	Time-1(s)	Energy-2	Time-2(s)
αβ-swap	645227	28.86	709120	20.04
α-expansion	634931	9.52	723360	9.78
TRW-S	**634720**	94.86	**651696**	226.07
BP	662108	170.67	2155759	244.71
Range Swap	**634720**	39.75	**651696**	80.40
Range Expansion	**634720**	66.13	**651696**	80.70

(a)

Algorithm	Energy-1	Time-1(s)	Energy-2	Time-2(s)
αβ-swap	1056109	35.00	1198029	52.98
α-expansion	1052860	15.16	1320088	11.95
TRW-S	1053341	142.19	1057371	339.02
BP	1117782	180.65	2443796	368.14
Range Swap	**1052762**	100.49	**1057041**	168.28
Range Expansion	**1052762**	129.30	**1057041**	155.98

(b)

Algorithm	Energy-1	Time-1(s)	Energy-2	Time-2(s)
αβ-swap	3678200	18.48	3707268	20.25
α-expansion	3677950	11.73	3687874	8.79
TRW-S	3677578	131.65	3679563	332.94
BP	3789486	272.06	5180705	331.36
Range Swap	3686844	97.23	**3679552**	141.78
Range Expansion	**3613003**	120.14	**3679552**	191.20

(c)

Table 4: The energy obtained and the time taken by the algorithms used in the stereo reconstruction experiment. Columns 2 and 3 : truncated linear metric. Columns 4 and 5: truncated quadratic semi-metric. **(a)** Tsukuba. **(b)** Venus. **(c)** Teddy. The lowest energy obtained in each case is indicated using bold font.

labeling may have a higher energy than the previous labeling (in which case the new labeling is discarded and the previous labeling is retained). However, the graph construction has the advantage of considering all the random variables at each iteration. The larger search space enables the Range Expansion algorithm to improve the multiplicative bound for the truncated linear metric compared to Boykov et al. (2001) and Gupta and Tardos (2000) and provide the best known bound for the truncated quadratic semi-metric. In practice, the Range Expansion algorithm is computationally more expensive than Range Swap (since the size of the graph is bigger), while providing comparable labelings (with slightly less energy).

Discussion. The speed of both Range Swap and Range Expansion can be further improved by using clever techniques such as those described by Kolmogorov and Shioura (2007) and/or Alahari et al. (2008) for convex and arbitrary unary potentials respectively.

Although we restricted our discussion to truncated convex models for simplicity, our method can easily be extended to handle truncated submodular models by using the graph construction of Schlesinger and Flach (2006) for the submodular part.

The analysis in Section 6.3 shows that, for the truncated linear and truncated quadratic models, the bound achieved by Range Expansion over intervals of any length L is equal to that of rounding the LP relaxation's optimal solution using the same intervals (Chekuri et al., 2005). This equivalence also extends to the Potts model (in which case α-expansion provides the same bound as the LP relaxation with the rounding scheme of Kleinberg and Tardos, 1999) and general metric potentials (in which case the recent method of Kumar and Koller, 2009 provides the same bound as the LP relaxation when using the rounding scheme of Chekuri et al., 2005). This raises the question about the relationship between move making algorithms and the rounding schemes used in convex relaxations. Note that despite recent efforts (Komodakis and Tziritas, 2007) analyzing certain move making algorithms in the context of primal-dual approaches for the LP relaxation, not many results are known about their connection with randomized rounding schemes. Although the discussion in Section 6.3 cannot be trivially generalized to all random fields, it offers a step towards answering this question. We believe that further exploration in this direction would help design efficient move making algorithms for more complex relaxations such as those described in Kumar et al. (2007) and Sontag and Jaakkola (2007).

Acknowledgments

M. Pawan Kumar is funded by NSF under grant IIS 0917151, MURI contract N000140710747, and the Boeing Corporation. Olga Veksler would like to acknowledge support provided by NSERC, CFI and ERA grants. Philip Torr thanks EPSRC and PASCAL. Philip Torr is in receipt of a Royal Society Wolfson Research Merit Award, and would like to acknowledge support from the Royal Society and Wolfson foundation.

Appendix A. Proof of Theorem 1

Theorem 1: An algorithm that provides a local minimum over smooth labelings achieves a multiplicative bound of 2.

Proof: We denote an optimum labeling by f^* (that is, $Q(f^*,\mathbf{D};\theta) \leq Q(f,\mathbf{D};\theta)$ for all labelings f) and a local minimum over smooth labelings by $\hat{f}$. Given f^*, we define a partitioning of the random variables into p subsets using $S_i \subseteq \{0,\cdots,n-1\}$ for $i = 0,1,\cdots,p-1$ such that $\bigcup_i S_i = \{0,1,\cdots,n-1\}$ and $S_i \cap S_j = \phi$ (that is, the null set) for all $i \neq j$. In other words, S_i define a disjoint and complete partitioning of the random variables. Furthermore, the subsets S_i are restricted such that f^* is a smooth labeling with respect to S_i. Note that, for any f^*, such a partitioning must exist. This can be seen by observing that the trivial partitioning where each partition consists of only one random variable satisfies the properties described above. In fact, there may be numerous distinct partitionings of the random variables into subsets S_i. Let us take the partitioning that has

the smallest number of subsets, that is p is as small as possible. Note that there are several such minimal partitionings, however the one we shall select does not alter what follows.

Since the subsets S_i define a minimal partitioning, it follows that for any $a \in S_i$ and $b \in S_j$ such that $i \neq j$ and $(a,b) \in \mathcal{E}$, $d(f^*(a) - f^*(b)) > M$. This can easily be proved by contradiction: if there exist $a \in S_i$ and $b \in S_j$ such that $d(f^*(a) - f^*(b)) \leq M$, then we can obtain a smaller partitioning by replacing S_i and S_j by their union. For each subset S_i, we define the following sets

$$\begin{aligned}
&\mathbf{v}(S_i) = \{v_a | a \in S_i\}, \\
&\mathcal{A}(S_i) = \{(a,b) | (a,b) \in \mathcal{E}, a \in S_i, b \in S_i\}, \\
&\mathcal{B}_1(S_i) = \{(a,b) | (a,b) \in \mathcal{E}, a \in S_i, b \notin S_i\}, \\
&\mathcal{B}_2(S_i) = \{(a,b) | (a,b) \in \mathcal{E}, a \notin S_i, b \in S_i\}, \\
&\mathcal{B}(S_i) = \mathcal{B}_1(S_i) \cup \mathcal{B}_2(S_i).
\end{aligned}$$

In other words, $\mathbf{v}(S_i)$ contains all the random variables specified by the subset S_i, $\mathcal{A}(S_i)$ contains the set of all edges in the graphical model of the MRF whose endpoints belong to the set $\mathbf{v}(S_i)$ and $\mathcal{B}(S_i)$ contains the set of all edges where only one endpoint belongs to $\mathbf{v}(S_i)$. For each S_i, we also define a labeling f_i such that

$$f_i(a) = \begin{cases} f^*(a) & \text{if} \quad a \in S_i, \\ \hat{f}(a) & \text{otherwise.} \end{cases}$$

Since $\hat{f}$ is a local minimum over smooth labelings, it follows from definition 2 that $Q(\hat{f}, \mathbf{D}; \theta) \leq Q(f_i, \mathbf{D}; \theta)$. By canceling out the common terms, we see that

$$\begin{aligned}
& \sum_{v_a \in \mathbf{v}(S_i)} \theta_a(\hat{f}(a)) + \sum_{(a,b) \in \mathcal{A}(S_i) \cup \mathcal{B}(S_i)} \theta_{ab}(\hat{f}(a), \hat{f}(b)) \\
\leq & \sum_{v_a \in \mathbf{v}(S_i)} \theta_a(f_i(a)) + \sum_{(a,b) \in \mathcal{A}(S_i) \cup \mathcal{B}(S_i)} \theta_{ab}(f_i(a), f_i(b)) \\
\leq & \sum_{v_a \in \mathbf{v}(S_i)} \theta_a(f^*(a)) + \sum_{(a,b) \in \mathcal{A}(S_i) \cup \mathcal{B}(S_i)} \theta_{ab}(f^*(a), f^*(b)).
\end{aligned}$$

The last expression holds true because: (i) $\theta_a(f_i(a)) = \theta_a(f^*(a))$ for all $v_a \in \mathbf{v}(S_i)$; (ii) $\theta_{ab}(f_i(a)), f_i(b)) = \theta_{ab}(f^*(a), f^*(b))$ for all $(a,b) \in \mathcal{A}(S_i)$; and (iii) $\theta_{ab}(f^*(a), f^*(b)) = w_{ab}M$ for all $(a,b) \in \mathcal{B}(S_i)$ (since $d(f^*(a) - f^*(b)) > M$). Summing the above inequality over all $i = 0, \cdots, p-1$ and using the fact that the LHS is at least $Q(\hat{f}, \mathbf{D}; \theta)$ we obtain

$$Q(\hat{f}, \mathbf{D}; \theta) \leq \sum_{v_a \in \mathbf{v}} \theta_a(f^*(a)) + 2 \sum_{(a,b) \in \mathcal{E}} \theta_{ab}(f^*(a), f^*(b)).$$

The factor 2 in the above inequality appears because the pairwise potential for each $(a,b) \notin \bigcup_i \mathcal{A}(S_i)$ will be counted twice (because it belongs to both $\mathcal{B}_1(S_i)$ and $\mathcal{B}_2(S_j)$ for some i and j). This proves the theorem. ∎

Appendix B. Proof of Lemma 1

Lemma 1: For the capacities defined in Equations (4) and (5), the cost of the st-cut which includes the edges (a_k, a_{k+1}) and $(b_{k'}, b_{k'+1})$ (that is, v_a and v_b take labels l_k and $l_{k'}$ respectively) is given by $w_{ab}d(k-k') + \kappa_{ab}$, where the constant $\kappa_{ab} = w_{ab}d(L)$.

Proof: We start by observing that due to the presence of the infinite capacity edges representing unary potentials, the *st*-cut will consist of only the following edges:

$$(a_k, a_{k+1}) \cup (b_{k'}, b_{k'+1}) \cup \{(a_{i'}, b_{j'}), i_m + 1 \leq i' \leq k, k'+1 \leq j' \leq j_m\}$$
$$\cup \{(a_{i'}, b_{j'}), k+1 \leq i' \leq k, i_m + 1 \leq j' \leq k'\}.$$

Using Equations (4) and (5) to sum the capacities of the above edges, we obtain the following expression:

$$\begin{aligned}
&\frac{w_{ab}}{2}\left(d(L-k+i_m)+d(k-i_m)\right)+\frac{w_{ab}}{2}\left(d(L-k'+i_m)+d(k'-i_m)\right)\\
&+\sum_{i'=i_m+1}^{k}\sum_{j'=k'+1}^{j_m}\frac{w_{ab}}{2}\left(d(i'-j'+1)-2d(i'-j')+d(i'-j'-1)\right)\\
&+\sum_{i'=k+1}^{j_m}\sum_{j'=i_m+1}^{k'}\frac{w_{ab}}{2}\left(d(i'-j'+1)-2d(i'-j')+d(i'-j'-1)\right). \qquad (13)
\end{aligned}$$

In order to simplify this expression, consider

$$\begin{aligned}
&\sum_{j'=k'+1}^{j_m}\left(d(i'-j'+1)-2d(i'-j')+d(i'-j'-1)\right)\\
=\;& d(i'-k')-2d(i'-k'-1)+d(i'-k'-2)\\
&+d(i'-k'-1)-2d(i'-k'-2)+d(i'-k'-3)\\
&\vdots\\
&+d(i'-j_m+2)-2d(i'-j_m+1)+d(i'-j_m)\\
&+d(i'-j_m+1)-2d(i'-j_m)+d(i'-j_m-1)\\
=\;& d(i'-k')-d(i'-k'-1)-d(i'-j_m)+d(i'-j_m+1). \qquad (14)
\end{aligned}$$

Hence, it follows that

$$\begin{aligned}
&\sum_{i'=i_m+1}^{k}\sum_{j'=k'+1}^{j_m}\left(d(i'-j'+1)-2d(i'-j')+d(i'-j'-1)\right)\\
=\;& d(i_m+1-k')-d(i_m-k')-d(i_m-j_m+1)+d(i_m-j_m)\\
&+d(i_m+2-k')-d(i_m+1-k')-d(i_m-j_m+2)+d(i_m-j_m+1)\\
&\vdots\\
&+d(k-k'-1)-d(k-k'-2)-d(k-j_m-1)+d(k-j_m-2)\\
&+d(k-k')-d(k-k'-1)-d(k-j_m)+d(k-j_m-1)\\
=\;& d(k-k')-d(j_m-k)-d(k'-i_m)+d(j_m-i_m)\\
=\;& d(k-k')-d(L-k+i_m)-d(i_m-k')+d(L), \qquad (15)
\end{aligned}$$

where the last expression holds because $L = j_m - i_m$. Note that we also use the fact that $d(x) = d(-x)$. Similarly, it can be shown that

$$\begin{aligned}
&\sum_{i'=k+1}^{j_m}\sum_{j'=i_m+1}^{k'}\left(d(i'-j'+1)-2d(i'-j')+d(i'-j'-1)\right)\\
=\;& d(k-k')-d(L-k'+i_m)-d(i_m-k)+d(L). \qquad (16)
\end{aligned}$$

Substituting Equations (15) and (16) into expression (13), we obtain the cost of the st-cut as

$$\begin{aligned} & \frac{w_{ab}}{2}\left(d(L-k+i_m)+d(k-i_m)\right)+\frac{w_{ab}}{2}\left(d(L-k'+i_m)+d(k'-i_m)\right) \\ & +\frac{w_{ab}}{2}\left(d(k-k')-d(L-k+i_m)-d(i_m-k')+d(L)\right) \\ & +\frac{w_{ab}}{2}\left(d(k-k')-d(L-k'+i_m)-d(i_m-k)+d(L)\right) \\ = \quad & w_{ab}d(k-k')+\kappa_{ab}. \end{aligned}$$

This proves that the capacities in Equations (4) and (5) model convex pairwise potentials exactly up to an additive constant. ∎

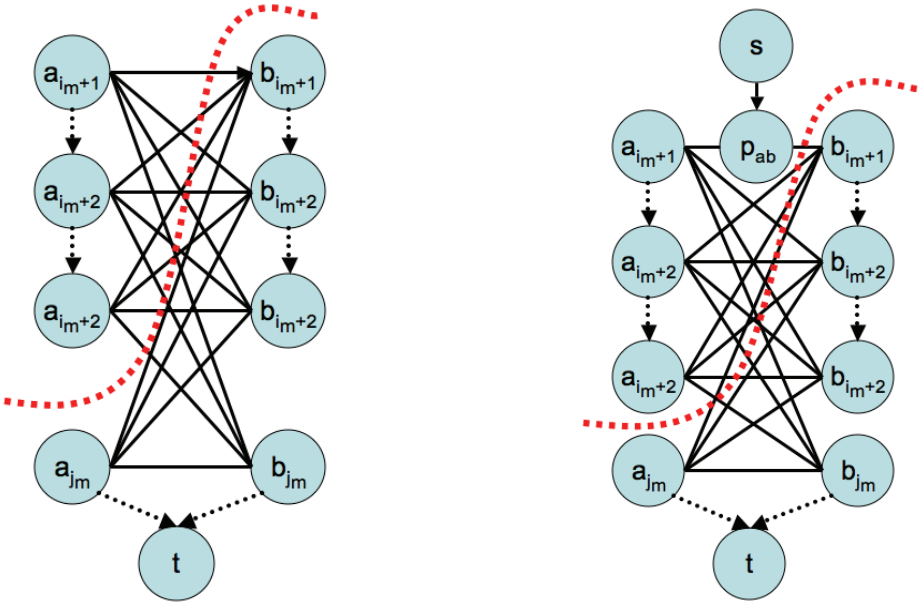


Figure 8: The st-cut (the dashed curve between the two sets of nodes $\{a_{i_m+1},\cdots,a_{j_m}\}$ and $\{b_{i_m+1},\cdots,b_{j_m}\}$; shown in red if viewed in color) that assigns $f(a)\in I_m$ and $f(b)\notin I_m$. Undirected edges represents directed edges in both directions (with the same capacity). **(a)** $f_m(a)\in I_m$ and $f_m(b)\notin I_m$. In this case, we introduce a directed edge from a_{i_m+1} to b_{i_m+1} that is included in the st-cut.**(b)** $f_m(a)\notin I_m$ and $f_m(b)\notin I_m$. In this case, we introduce an auxiliary variable p_{ab} that belongs to the source set in the st-cut.

Appendix C. Proof of Lemma 2

Lemma 2: For the graph described in § 6.1, property 9 holds true.
Proof: We will show the proof for $f(a)\in I_m$ and $f(b)=f_m(b)\notin I_m$. The proof for $f(a)=f_m(a)\notin I_m$ and $f(b)\in I_m$ can be obtained from the following arguments trivially.

There are two possible cases to be considered: (i) $f_m(a) \in I_m$; and (ii) $f_m(a) \notin I_m$. In the first case, the edges that specify the *st*-cut are given by (see Fig. 8(a))

$$\begin{aligned}&(a_{f(a)}, a_{f(a)+1}) \cup \{(a_{i'}, b_{j'}), i_m + 2 \leq i' \leq f(a), i_m + 1 \leq j' \leq j_m\} \\ &\cup \{(a_{i_m+1}, b_{j'}), i_m + 2 \leq j' \leq j_m\} \cup (a_{i_m+1}, b_{i_m+1}).\end{aligned} \tag{17}$$

In the second case, the *st*-cut is specified by (see Fig. 8(b))

$$\begin{aligned}&(a_{f(a)}, a_{f(a)+1}) \cup \{(a_{i'}, b_{j'}), i_m + 2 \leq i' \leq f(a), i_m + 1 \leq j' \leq j_m\} \\ &\cup \{(a_{i_m+1}, b_{j'}), i_m + 2 \leq j' \leq j_m\} \cup (p_{ab}, b_{i_m+1}).\end{aligned}$$

Note that in this case p_{ab} belongs to the same partition as the source s. This can be shown easily by observing that the cost of the *st*-cut increases if p_{ab} belongs to the partition containing the sink t (since this would include edges (a_{i_m+1}, p_{ab}) and (s, p_{ab}) in the *st*-cut). The two cases differ only in that the first includes the edge (a_{i_m+1}, b_{i_m+1}) and the second includes the edge (p_{ab}, b_{i_m+1}). However, the capacity of both these edges is equal to $w_{ab}M + \kappa_{ab}/2$. Hence it follows that the cost of the *st*-cut in both the cases is the same. Therefore it is sufficient to show that the Lemma holds true for the first case.

The cost of the *st*-cut for the edges in Equation (17) is given by

$$\begin{aligned}&\frac{w_{ab}}{2}\left(d(L - f(a) + i_m) + d(f(a) - i_m)\right) \\ &+ \sum_{i'=i_m+2}^{f(a)} \sum_{j'=i_m+1}^{j_m} \frac{w_{ab}}{2}\left(d(i' - j' + 1) - 2d(i' - j') + d(i' - j' - 1)\right) \\ &+ \sum_{j'=i_m+2}^{j_m} \frac{w_{ab}}{2}\left(d(i_m - j' + 2) - 2d(i_m - j' + 1) + d(i_m - j')\right) \\ &+ w_{ab}M + \frac{\kappa_{ab}}{2}.\end{aligned} \tag{18}$$

In order to simplify the above expression, we begin by observing that

$$\begin{aligned}&\sum_{j'=i_m+1}^{j_m} \left(d(i' - j' + 1) - 2d(i' - j') + d(i' - j' - 1)\right) \\ = \quad & d(i' - i_m) - d(i' - i_m - 1) - d(i' - j_m) + d(i' - j_m - 1).\end{aligned}$$

The above equation is obtained by substituting $k' = i_m$ in Equation (14). It follows that

$$\begin{aligned}&\sum_{i'=i_m+2}^{f(a)} \sum_{j'=i_m+1}^{j_m} \frac{w_{ab}}{2}\left(d(i' - j' + 1) - 2d(i' - j') + d(i' - j' - 1)\right) \\ = \quad & d(2) - d(1) - d(i_m - j_m + 2) + d(i_m - j_m + 1) \\ & + d(3) - d(2) - d(i_m - j_m + 3) + d(i_m - j_m + 2) \\ & \vdots \\ & + d(f(a) - i_m - 1) - d(f(a) - i_m - 2) - d(f(a) - j_m - 1) + d(f(a) - j_m - 2) \\ & + d(f(a) - i_m) - d(f(a) - i_m - 1) - d(f(a) - j_m) + d(f(a) - j_m - 1) \\ = \quad & d(f(a) - i_m) - d(j_m - f(a)) - d(1) + d(j_m - i_m - 1) \\ = \quad & d(f(a) - i_m) - d(L - f(a) + i_m) - d(1) + d(L - 1),\end{aligned} \tag{19}$$

where the last expression is obtained using $L = j_m - i_m$. Once again, we use the fact that $d(x) = d(-x)$. Similarly, by substituting $k' = i_m + 1$ in Equation (14), we get

$$\begin{aligned} & \sum_{j'=i_m+2}^{j_m} \frac{w_{ab}}{2} \left(d(i_m - j' + 2) - 2d(i_m - j' + 1) + d(i_m - j') \right) \\ = \quad & d(0) - d(1) - d(j_m - i_m - 1) + d(j_m - i_m) \\ = \quad & d(0) - d(1) - d(L-1) + d(L). \end{aligned} \tag{20}$$

By simplifying expression (18) using Equations (19) and (20), the cost of the *st*-cut is given by

$$\begin{aligned} & \frac{w_{ab}}{2} \left(d(L - f(a) + i_m) + d(f(a) - i_m) \right) \\ & + \frac{w_{ab}}{2} \left(d(f(a) - i_m) - d(L - f(a) + i_m) - d(1) + d(L-1) \right) \\ & + \frac{w_{ab}}{2} \left(d(0) - d(1) - d(L-1) + d(L) \right) \\ & + w_{ab} M + \frac{\kappa_{ab}}{2} \\ = \quad & w_{ab} d(f(a) - (i_m + 1)) + w_{ab} \hat{d}(f(a) - (i_m + 1)) + w_{ab} M + \kappa_{ab}, \end{aligned}$$

where the last expression is obtained using the definition of $\hat{d}(\cdot)$ in Equation (10) and the fact that $\kappa_{ab} = w_{ab} d(L)$. This proves the Lemma. ∎

Appendix D. Proof of Lemma 3

Lemma 3: At an iteration of our algorithm, given the current labeling f_m and an interval $I_m = [i_m + 1, j_m]$, the new labeling f_{m+1} obtained by solving the *st*-MINCUT problem reduces the energy by at least the following:

$$\begin{aligned} & \sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f_m(a)) + \sum_{(a,b) \in \mathcal{A}(f^*, I_m) \bigcup \mathcal{B}(f^*, I_m)} \theta_{ab}(f_m(a), f_m(b)) \\ & - \left(\sum_{v_a \in \mathbf{v}(f^*, I_m)} \theta_a(f^*(a)) + \sum_{(a,b) \in \mathcal{A}(f^*, I_m)} e^m_{ab} + \sum_{(a,b) \in \mathcal{B}_1(f^*, I_m)} e^m_a + \sum_{(a,b) \in \mathcal{B}_2(f^*, I_m)} e^m_b \right). \end{aligned}$$

Proof: From the arguments in § 6.2, it is clear that the energy of the new labeling f_{m+1} is bounded from above by the cost of the *st*-MINCUT. The cost of the *st*-MINCUT itself is bounded from above by the cost of any other *st*-cut in the graph $\mathcal{G}_m$. Consider one such *st*-cut that results in the following labeling:

$$f(a) = \begin{cases} f^*(a) & \text{if} \quad v_a \in \mathbf{v}(f^*, I_m) \\ f_m(a) & \text{otherwise.} \end{cases}$$

We will now derive the cost of this *st*-cut using the properties in § 6.2. We consider the following six cases:

- For random variables $v_a \notin \mathbf{v}(f^*, I_m)$ it follows from Property 5 that the cost of the *st*-cut will include the unary potentials associated with such variables exactly, that is,

$$\sum_{v_a \notin \mathbf{v}(f^*, I_m)} \theta_a(f_m(a)). \tag{21}$$

- For neighboring random variables $(a,b) \notin \mathcal{A}(f^*,I_m) \bigcup \mathcal{B}(f^*,I_m)$ it follows from Property 6 that the cost of the *st*-cut will include the pairwise potentials associated with such neighboring variables exactly up to a constant κ_{ab}, that is,

$$\sum_{(a,b)\notin \mathcal{A}(f^*,I_m) \bigcup \mathcal{B}(f^*,I_m)} (\theta_{ab}(f_m(a),f_m(b)) + \kappa_{ab}) . \tag{22}$$

- For random variables $v_a \in \mathbf{v}(f^*,I_m)$, it follows from Property 5 that the cost of the *st*-cut will include the unary potentials associated with such variables exactly, that is,

$$\sum_{v_a \in \mathbf{v}(f^*,I_m)} \theta_a(f^*(a)). \tag{23}$$

- For neighboring random variables $(a,b) \in \mathcal{A}(f^*,I_m)$ it follows from Properties 7 and 8 that the cost of the *st*-cut will include the following:

$$\sum_{(a,b)\in \mathcal{A}(f^*,I_m)} (e^m_{ab} + \kappa_{ab}) . \tag{24}$$

- For neighboring random variables $(a,b) \in \mathcal{B}_1(f^*,I_m)$ it follows from Property 9 that the cost of the *st*-cut will include the following:

$$\sum_{(a,b)\in \mathcal{B}_1(f^*,I_m)} (e^m_{a} + \kappa_{ab}) . \tag{25}$$

- For neighboring random variables $(a,b) \in \mathcal{B}_2(f^*,I_m)$ it follows from Property 9 that the cost of the *st*-cut will include the following:

$$\sum_{(a,b)\in \mathcal{B}_2(f^*,I_m)} (e^m_{b} + \kappa_{ab}) . \tag{26}$$

The energy of f (that is, $Q(f,\mathbf{D};\theta)$), and hence $Q(f_{m+1},\mathbf{D};\theta)$, is less than or equal to the sum of terms (21)-(26) minus $\sum_{(a,b)\in\mathcal{E}} \kappa_{ab}$. It follows that the difference between the energy of the current labeling f_m and the new labeling f_{m+1}, that is, $Q(f_m,\mathbf{D};\theta) - Q(f_{m+1},\mathbf{D};\theta)$, is at least

$$\begin{aligned} &\sum_{v_a\in\mathbf{v}(f^*,I_m)} \theta_a(f_m(a)) + \sum_{(a,b)\in\mathcal{A}(f^*,I_m)\bigcup\mathcal{B}(f^*,I_m)} \theta_{ab}(f_m(a),f_m(b)) \\ &- \left(\sum_{v_a\in\mathbf{v}(f^*,I_m)} \theta_a(f^*(a)) + \sum_{(a,b)\in\mathcal{A}(f^*,I_m)} e^m_{ab} + \sum_{(a,b)\in\mathcal{B}_1(f^*,I_m)} e^m_a + \sum_{(a,b)\in\mathcal{B}_2(f^*,I_m)} e^m_b \right). \end{aligned}$$

This proves the Lemma. ∎

Appendix E. Proof of Lemma 4

Lemma 4: When $d(\cdot)$ is linear, that is, $d(x) = |x|$, the following inequality holds true:

$$\begin{aligned} &\frac{1}{L} \sum_r \sum_{I_m\in I_r} \left(\sum_{(a,b)\in\mathcal{A}(f^*,I_m)} e^m_{ab} + \sum_{(a,b)\in\mathcal{B}_1(f^*,I_m)} e^m_a + \sum_{(a,b)\in\mathcal{B}_2(f^*,I_m)} e^m_b \right) \\ &\leq \left(2 + \max\left\{ \frac{2M}{L}, \frac{L}{M} \right\} \right) \sum_{(a,b)\in\mathcal{E}} \theta_{ab}(f^*(a)f^*(b)). \end{aligned} \tag{27}$$

Proof: The following is a slight modification of the proof of Lemma 4.5 of Chekuri et al. (2005) and is presented here for the sake of completeness. Since we are dealing with the truncated linear metric, the terms e^m_{ab}, e^m_a and e^m_b can be simplified as

$$e^m_{ab} = w_{ab}|f^*(a) - f^*(b)|, e^m_a = w_{ab}(f^*(a) - i_m - 1 + M), e^m_b = w_{ab}(f^*(b) - i_m - 1 + M).$$

We begin by observing that the LHS of inequality (27) can be rewritten as

$$\frac{1}{L}\sum_{(a,b)\in\mathcal{E}}\left(\sum_{\mathcal{A}(f^*,I_m)\ni(a,b)} e^m_{ab} + \sum_{\mathcal{B}_1(f^*,I_m)\ni(a,b)} e^m_a + \sum_{\mathcal{B}_2(f^*,I_m)\ni(a,b)} e^m_b\right). \tag{28}$$

In order to prove the Lemma, we consider the following three cases for two neighboring random variables $(a,b)\in\mathcal{E}$.

Case I: $d(f^*(a),f^*(b)) = |f^*(a)-f^*(b)| \leq L$ and hence, $\theta_{ab}(f^*(a),f^*(b)) = w_{ab}M$.

In this case, it is clear that $(a,b)\notin\mathcal{A}(f^*,I_m)$ for all intervals I_m since the length of each interval is L. Furthermore, the conditions for $(a,b)\in\mathcal{B}_1(f^*,I_m)$ and $(a,b)\in\mathcal{B}_2(f^*,I_m)$ are given by

$$\begin{aligned}(a,b)\in\mathcal{B}_1(f^*,I_m) &\Longleftrightarrow i_m\in[f^*(a)-L, f^*(a)-1],\\ (a,b)\in\mathcal{B}_2(f^*,I_m) &\Longleftrightarrow i_m\in[f^*(b)-L, f^*(b)-1].\end{aligned}$$

In order to prove inequality (27), we observe that

$$\begin{aligned}
&\sum_{\mathcal{A}(f^*,I_m)\ni(a,b)} e^m_{ab} + \sum_{\mathcal{B}_1(f^*,I_m)\ni(a,b)} e^m_a + \sum_{\mathcal{B}_2(f^*,I_m)\ni(a,b)} e^m_b\\
= \;& w_{ab}\left(\sum_{i_m=f^*(a)-L}^{f^*(a)-1}(M+f^*(a)-i_m-1) + \sum_{i_m=f^*(b)-L}^{f^*(b)-1}(M+f^*(b)-i_m-1)\right)\\
= \;& w_{ab}\left(2LM + \sum_{i_m=f^*(a)-L}^{f^*(a)-1}(f^*(a)-i_m-1) + \sum_{i_m=f^*(a)-L}^{f^*(a)-1}(f^*(a)-i_m-1)\right)\\
\leq \;& w_{ab}\left(2LM+2L^2\right)\\
= \;& L\left(2+\frac{L}{M}\right)\theta_{ab}(f^*(a),f^*(b)),
\end{aligned} \tag{29}$$

where the last expression is obtained using the fact that $\theta_{ab}(f^*(a),f^*(b)) = w_{ab}M$.

Case II: $M \leq d(f^*(a),f^*(b)) = |f^*(a)-f^*(b)| < L$ and hence, $\theta_{ab}(f^*(a),f^*(b)) = w_{ab}M$.

We will assume, without loss of generality, that $f^*(a)\leq f^*(b)$. In this case, the conditions for $(a,b)\in\mathcal{A}(f^*,I_m)$, $(a,b)\in\mathcal{B}_1(f^*,I_m)$ and $(a,b)\in\mathcal{B}_2(f^*,I_m)$ are given by

$$\begin{aligned}(a,b)\in\mathcal{A}(f^*,I_m) &\Longleftrightarrow i_m\in[f^*(b)-L, f^*(a)-1],\\ (a,b)\in\mathcal{B}_1(f^*,I_m) &\Longleftrightarrow i_m\in[f^*(a)-L, f^*(b)-L-1],\\ (a,b)\in\mathcal{B}_2(f^*,I_m) &\Longleftrightarrow i_m\in[f^*(a), f^*(b)-1].\end{aligned}$$

Again, in order to prove inequality (27), we observe that

$$\begin{aligned}
&\sum_{\mathcal{A}(f^*,I_m)\ni(a,b)} e^m_{ab} + \sum_{\mathcal{B}_1(f^*,I_m)\ni(a,b)} e^m_a + \sum_{\mathcal{B}_2(f^*,I_m)\ni(a,b)} e^m_b \\
= \quad & w_{ab}\left(\sum_{i_m=f^*(b)-L}^{f^*(a)-1}(f^*(b)-f^*(a)) + \sum_{i_m=f^*(a)-L}^{f^*(b)-L-1}(M+f^*(a)-i_m-1)\right. \\
& \left. + \sum_{i_m=f^*(a)}^{f^*(b)-1}(M+f^*(b)-i_m-1)\right) \\
\leq \quad & w_{ab}\,(2L+2M-(f^*(b)-f^*(a)))\,(f^*(b)-f^*(a)) \\
\leq \quad & w_{ab}L(2M+L) \\
= \quad & L\left(2+\frac{L}{M}\right)\theta_{ab}(f^*(a),f^*(b)), \qquad (30)
\end{aligned}$$

where the last expression is obtained using the fact that $\theta_{ab}(f^*(a),f^*(b)) = w_{ab}M$.

Case III: $d(f^*(a),f^*(b)) = |f^*(a)-f^*(b)| \leq M$ and hence, $\theta_{ab}(f^*(a),f^*(b)) = w_{ab}|f^*(a)-f^*(b)|$.

We will assume, without loss of generality, that $f^*(a) \leq f^*(b)$. Similar to case II, the conditions for $(a,b) \in \mathcal{A}(f^*,I_m)$, $(a,b) \in \mathcal{B}_1(f^*,I_m)$ and $(a,b) \in \mathcal{B}_2(f^*,I_m)$ are given by

$$\begin{aligned}
&(a,b) \in \mathcal{A}(f^*,I_m) \Longleftrightarrow i_m \in [f^*(b)-L, f^*(a)-1], \\
&(a,b) \in \mathcal{B}_1(f^*,I_m) \Longleftrightarrow i_m \in [f^*(a)-L, f^*(b)-L-1], \\
&(a,b) \in \mathcal{B}_2(f^*,I_m) \Longleftrightarrow i_m \in [f^*(a), f^*(b)-1].
\end{aligned}$$

Once again, we consider

$$\begin{aligned}
&\sum_{\mathcal{A}(f^*,I_m)\ni(a,b)} e^m_{ab} + \sum_{\mathcal{B}_1(f^*,I_m)\ni(a,b)} e^m_a + \sum_{\mathcal{B}_2(f^*,I_m)\ni(a,b)} e^m_b \\
= \quad & w_{ab}\left(\sum_{i_m=f^*(b)-L}^{f^*(a)-1}(f^*(b)-f^*(a)) + \sum_{i_m=f^*(a)-L}^{f^*(b)-L-1}(M+f^*(a)-i_m-1)\right. \\
& \left. + \sum_{i_m=f^*(a)}^{f^*(b)-1}(M+f^*(b)-i_m-1)\right) \\
\leq \quad & w_{ab}\,(2L+2M-(f^*(b)-f^*(a)))\,(f^*(b)-f^*(a)) \\
\leq \quad & w_{ab}(2L+2M)(f^*(b)-f^*(a)) \\
= \quad & L\left(2+\frac{2M}{L}\right)\theta_{ab}(f^*(a),f^*(b)), \qquad (31)
\end{aligned}$$

where the last expression is obtained using the fact that $\theta_{ab}(f^*(a),f^*(b)) = w_{ab}(f^*(b)-f^*(a))$.

Substituting inequalities (29), (30) and (31) in expression (28) and dividing both sides by L for all $(a,b) \in \mathcal{E}$, we obtain inequality (27). This proves the Lemma. ∎

References

K. Alahari, P. Kohli, and P. H. S. Torr. Reduce, reuse & recycle: Efficiently solving multi-label MRFs. In *CVPR*, 2008.

J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 48:259–302, 1986.

S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. In *ICCV*, pages 1073–1080, 1998.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–1137, 2004.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, 2001.

C. Chekuri, S. Khanna, J. Naor, and L. Zosin. A linear programming formulation and approximation algorithms for the metric labelling problem. *SIAM Journal on Discrete Mathematics*, 18(3):606–635, 2005.

E. Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. *Soviet Math. Dokl.*, 11:1277–1280, 1970.

P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *CVPR*, pages II: 66–73, 2000.

P. Felzenszwalb and D. Huttenlocher. Efficient belief propagation for early vision. In *CVPR*, pages I: 261–268, 2004.

A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing for MAP LP-relaxations. In *NIPS*, 2007.

A. Goldberg and R. Tarjan. A new approach to the maximum-flow problem. *Journal of ACM*, 35 (4):921–940, 1988.

S. Gould, F. Amat, and D. Koller. Alphabet SOUP: A framework for approximate energy minimization. In *CVPR*, 2009.

A. Gupta and E. Tardos. A constant factor approximation algorithm for a class of classification problems. In *STOC*, 2000.

H. Ishikawa. Exact optimization for Markov random fields with convex priors. *PAMI*, 25(10): 1333–1336, October 2003.

J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. In *STOC*, pages 14–23, 1999.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28 (10):1568–1583, 2006.

V. Kolmogorov and A. Shioura. New algorithms for the dual of the convex cost network flow problem with applications to computer vision. Technical report, University College London, 2007.

N. Komodakis and G. Tziritas. Approximate labeling via graph-cuts based on linear programming. *PAMI*, 2007.

N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, 2007.

A. Koster, C. van Hoesel, and A. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3-5):89–97, 1998.

M. P. Kumar and D. Koller. MAP estimation of semi-metric MRFs via hierarchical graph cuts. In *UAI*, 2009.

M. P. Kumar and P. H. S. Torr. Improved moves for truncated convex models. In *NIPS*, 2008.

M. P. Kumar, V. Kolmogorov, and P. H. S. Torr. An analysis of convex relaxations for MAP estimation. In *NIPS*, 2007.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labelling sequence data. In *ICML*, 2001.

T. Meltzer, C. Yanover, and Y. Weiss. Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In *ICCV*, 2005.

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

P. Ravikumar, A. Agarwal, and M. Wainwright. Message-passing for graph-structured linear programs: Proximal projections, convergence and rounding schemes. In *ICML*, 2008.

D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. Technical Report TUD-F106-01, Dresden University of Technology, 2006.

M. Schlesinger. Sintaksicheskiy analiz dvumernykh zritelnikh singnalov v usloviyakh pomekh (syntactic analysis of two-dimensional visual signals in noisy conditions). *Kibernetika*, 4:113–130, 1976.

M. Schlesinger and V. Giginyak. Solution to structural recognition (MAX,+)-problems by their equivalent transformations. Part 1. *Control Systems and Computers*, 1:3–15, 2007a.

M. Schlesinger and V. Giginyak. Solution to structural recognition (MAX,+)-problems by their equivalent transformations. Part 2. *Control Systems and Computers*, 2:3–18, 2007b.

J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, pages I: 1–15, 2006.

D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *NIPS*, 2007.

R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *PAMI*, 2008.

V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.

O. Veksler. *Efficient Graph-Based Energy Minimization Methods in Computer Vision*. PhD thesis, Cornell University, 1999.

O. Veksler. Graph cut based optimization for MRFs with truncated convex priors. In *CVPR*, 2007.

M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on trees: Message passing and linear programming. *IEEE Trans. on Information Theory*, 51(11):3697–3717, 2005.

Journal of Machine Learning Research 12 (2011) 69-73 Submitted 5/09; Revised 8/10; Published 1/11

CARP: Software for Fishing Out Good Clustering Algorithms

Volodymyr Melnykov VOLODYMYR.MELNYKOV@NDSU.EDU
Department of Statistics
North Dakota State University
Fargo, ND 58102, USA

Ranjan Maitra MAITRA@IASTATE.EDU
Department of Statistics and Statistical Laboratory
Iowa State University
Ames, IA 50011, USA

Editor: Cheng Soon Ong

Abstract

This paper presents the CLUSTERING ALGORITHMS' REFEREE PACKAGE or CARP, an open source GNU GPL-licensed C package for evaluating clustering algorithms. Calibrating performance of such algorithms is important and CARP addresses this need by generating datasets of different clustering complexity and by assessing the performance of the concerned algorithm in terms of its ability to classify each dataset relative to the true grouping. This paper briefly describes the software and its capabilities.

Keywords: CARP, MIXSIM, clustering algorithm, Gaussian mixture, overlap

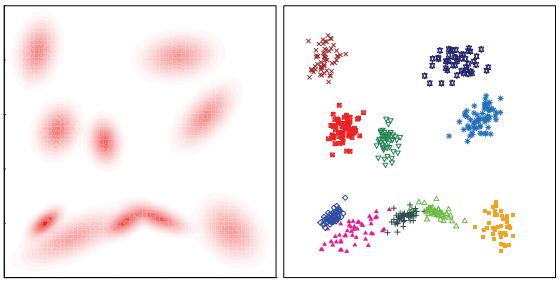
1. Introduction

There are many clustering algorithms available, but no uniformly clear winner. Thus, calibrating the performance of each algorithm in different situations is important. CARP provides software to evaluate performance of any user-provided clustering technique under simulated datasets of specified clustering complexity. At its heart is software that implements Maitra and Melnykov's (2010) algorithm to simulate datasets from Gaussian mixtures of different clustering difficulty. CARP provides an integrated software tool which generates datasets using the above, uses the clustering algorithm being evaluated on each dataset and compares the derived grouping relative to the true via indices such as the Adjusted Rand ($\mathcal{R}$) index (Hubert and Arabie, 1985). This paper discusses usage, applicability and flexibility of CARP.

2. CARP: An Integrated Tool for Evaluating Clustering Algorithms

There is some software available for simulating datasets for evaluating performance of clustering algorithms, for example, Qiu and Joe's (2006) open-source R package CLUSTERGENERATION (formerly, GENCLUS) and Steinley and Henson's (2005) publicly unavailable OCLUS code for use only in the proprietary MATLAB package. Beyond the algorithmic and other limitations (Maitra and Melnykov, 2010) underlying both these packages, none of them provide an integrated tool to evaluate clustering algorithms. CARP addresses this shortcoming by seamlessly integrating three stages. The first stage generates datasets given user-specified measures for desired clustering complexity

©2011 Volodymyr Melnykov and Ranjan Maitra.

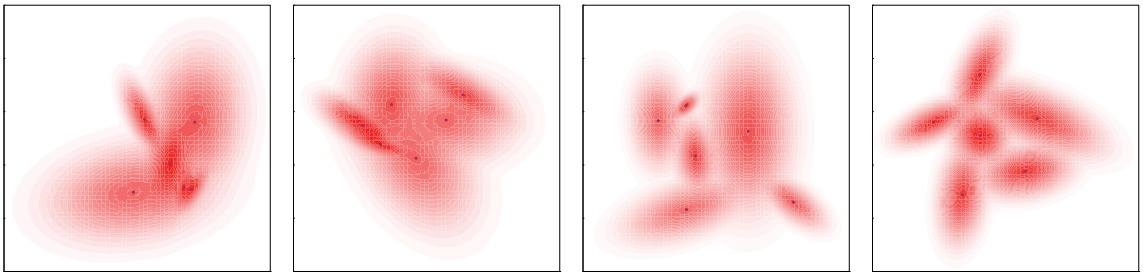


(a) (b)

Figure 1: (a) Sample 10-component bivariate mixture distribution. (b) Corresponding sample dataset generated at the first stage of CARP.

following the definitions of Maitra and Melnykov (2010). The next stage clusters each dataset using the user-provided algorithm in executable form. The final phase evaluates performance—in terms of the default $\mathcal{R}$ or some other index supplied by the user in executable form—of this clustering algorithm by comparing its derived partitioning with the true grouping. The package is specifically designed to be flexible enough to allow the user to provide clustering algorithms and evaluation measures in his/her preferred programming language. We now detail the three stages of CARP.

Stage I: Simulating Datasets of Given Clustering Complexity: The first stage of CARP implements Maitra and Melnykov's (2010) algorithms to generate datasets from Gaussian mixtures of different numbers of components, dimensions and dispersions characterized through summaries of the pairwise overlap that serves as a surrogate measure for clustering complexity. For any two components (say i, j) of a Gaussian mixture density $g(x) = \sum_{k=1}^{K} \pi_k \phi(x;\mu_k,\Sigma_k)$, the overlap is defined as $\omega_{ij} = \omega_{i|j} + \omega_{j|i}$, where $\omega_{j|i}$ is the probability that an observation from the ith component is misclassified to be from the jth one, with $\omega_{i|j}$ defined similarly. Analytical expressions for $\omega_{i|j}$ and $\omega_{j|i}$ being impractical, we use Theorem 2.1 in Maitra and Melnykov (2010) to calculate them numerically and efficiently. For a K-component mixture, there are $\binom{K}{2}$ ω_{ij}s, so clustering complexity is characterized using the average (ω) or maximum ($\breve{\omega}$) pairwise overlap measures. The software implements Algorithm 2.2.1 of Maitra and Melnykov (2010) to simulate Gaussian mixtures corresponding to a given ω or $\breve{\omega}$, as in the sample ten-component bivariate mixture distribution satisfying $\breve{\omega} = 0.1$ of Figure 1a. For a more comprehensive summary of clustering complexity, CARP uses their Algorithm 2.2.2 to simulate mixtures according to the pair $(\omega, \breve{\omega})$, as illustrated in Figure 2. Note that Figures 2a-b have the same ω but in the first case, this value is driven by the overlap between a few pairs of components. In the second case, many more components contribute. Similarly



(a) $\omega = 0.05, \breve{\omega} = 0.198$ (b) $\omega = 0.05, \breve{\omega} = 0.135$ (c) $\omega = 0.001, \breve{\omega} = 0.0036$ (d) $\omega = 0.001, \breve{\omega} = 0.0027$

Figure 2: Sample 6-component bivariate mixture distributions for different $(\omega, \breve{\omega})$s.

p	1	2	3	5	10	20	50	100
$K = 10$	12.051	8.993	5.360	0.635	0.147	0.206	0.774	4.202
$K = 100$	1236.965	1003.951	574.208	67.567	13.668	19.978	71.023	373.375

Table 1: Median time (secs) to simulate 25 K-component p-dimensional Gaussian mixtures.

for Figures 2c-d. CARP can simulate mixture models of many components (K) and dimensions (p): Table 1 summarizes the median time over 25 samples to obtain realizations of Gaussian mixtures at each setting on a desktop workstation with dual quad-core Intel® Xeon® X5482 @ 3.20 GHz processors running the Fedora 11 Linux distribution with the `2.6.30.9-102.fc11.x86_64` kernel and the `gcc` 4.4.1 suite of compilers. Note that an increase in K affects running time more than p.

The first phase of CARP thus involves simulating mixture distributions corresponding to user-specified K, p and clustering complexity in the form of $\breve{\omega}$ and/or ω. Given the desired sample size (n) of the dataset, this stage then obtains n simulated observations from each realized mixture model, as in the 500-observations sample displayed in Figure 1b. Beyond Gaussian mixtures, datasets from more general-shaped clusters with desired approximate clustering complexity are possible to generate using the inverse multivariate Box-Cox transform on the simulated Gaussian mixtures. This stage is also designed to allow for generating datasets with noisy variables and/or containing scatter/outliers. CARP can be used standalone, without calls to the next two stages, to generate only these distributions and datasets if so desired—the R package MIXSIM provides an additional interface to this stage of CARP.

Stage II: Clustering Simulated Datasets: The second phase of CARP uses the clustering algorithm(s) being evaluated to partition the datasets simulated in the first stage. Code for these algorithm(s) is submitted by the user in executable form, with no requirement on this code having to be in a specific programming language. This stage is designed to interface easily with other clustering tools, as illustrated in the manual.

Stage III: Evaluating the Performance of Clustering Algorithms: The third stage compares the partitionings provided by each clustering algorithm under investigation to the true. The default evaluation metric is $\mathcal{R}$, but other measures, including those with user-supplied code in executable form, can be used: see the manual for examples.

In summary, CARP provides distributions of the desired performance measure for the clustering method(s) being evaluated for given n, K, p and the desired $\breve{\omega}$, ω or $(\omega, \breve{\omega})$.

2.1 Demonstrating the Utility of CARP

For illustration, we consider a simple example where the goal is to evaluate performance of different algorithms in partitioning datasets with $(n, K, p) = (100, 7, 5)$ and varying clustering complexity characterized solely in terms of $\breve{\omega}$. (The exact CARP commands used to conduct this study are in Section 5 of the CARP software manual.) Table 2 summarizes performance of each algorithm on 25 datasets, each with different $\breve{\omega}$ and obtained using Stage I of CARP. Clustering algorithms are used here via external calls to the corresponding software. Evaluations are in terms of the default $\mathcal{R}$ provided with CARP. Clearly, performance degrades across the board with higher $\breve{\omega}$. Hierarchical clustering with Ward's and single linkages are, respectively, the best and worst performers in many cases, with k-means being second-best. Although a small-scale experiment, it highlights CARP's utility in summarizing and evaluating performance of different clustering algorithms.

	$\breve{\omega}$	0.500	0.400	0.300	0.200	0.150	0.100	0.050	0.010	0.005	0.001
W	$\mathcal{R}_{0.5}$	0.082	0.199	0.385	0.582	0.715	0.793	0.931	0.980	1.000	1.000
	$I_{\mathcal{R}}$	0.061	0.146	0.197	0.300	0.210	0.161	0.105	0.044	0.024	0.000
C	$\mathcal{R}_{0.5}$	0.063	0.164	0.327	0.477	0.596	0.714	0.787	0.956	0.991	1.000
	$I_{\mathcal{R}}$	0.052	0.125	0.216	0.257	0.231	0.216	0.182	0.145	0.052	0.005
A	$\mathcal{R}_{0.5}$	0.015	0.098	0.227	0.433	0.564	0.689	0.812	0.958	0.990	1.000
	$I_{\mathcal{R}}$	0.022	0.176	0.242	0.205	0.261	0.179	0.133	0.141	0.111	0.000
S	$\mathcal{R}_{0.5}$	0.002	0.003	0.005	0.017	0.058	0.234	0.490	0.820	0.833	0.897
	$I_{\mathcal{R}}$	0.007	0.007	0.010	0.202	0.321	0.442	0.449	0.116	0.148	0.170
K	$\mathcal{R}_{0.5}$	0.086	0.230	0.388	0.621	0.671	0.821	0.912	0.978	0.992	1.000
	$I_{\mathcal{R}}$	0.076	0.149	0.212	0.201	0.212	0.122	0.094	0.029	0.051	0.000
P	$\mathcal{R}_{0.5}$	0.083	0.209	0.341	0.560	0.677	0.811	0.901	0.977	0.982	1.000
	$I_{\mathcal{R}}$	0.059	0.147	0.237	0.208	0.217	0.192	0.102	0.043	0.027	0.000

Table 2: Performance of hierarchical algorithms with Ward's (W), complete (C), average (A), single (S) linkages, k-means (K), and partitioning around medoids (P) algorithms on clustering 5-dimensional datasets of size 100 with 7 groups. $\mathcal{R}_{0.5}$ and $I_{\mathcal{R}}$ represent the median and inter-quartile range of $\mathcal{R}$ over 100 replications.

2.2 Implementation

CARP is implemented in ANSI/ISO-compliant C and also available at `http://www.mloss.org`. Complete details on usage, parameters and examples are provided in the package's manual and README file. CARP can be used standalone in order to only simulate Gaussian mixtures and corresponding datasets. This standalone functionality is also provided by the R package MIXSIM which is publicly available from `http://www.R-project.org`.

3. Conclusions

CARP is a powerful and user-friendly open source software package for evaluating clustering algorithms. It realizes mixture models of pre-specified clustering complexity, and from there, datasets of desired sample sizes that are then partitioned using the clustering algorithms being appraised. Performance is summarized by comparing the obtained groupings with the true. CARP can also be used to evaluate semi-supervised clustering algorithms, or on simulated datasets with noisy variables or containing scatter/outliers. Clusters should each ideally be Gaussian-distributed, though the package also uses transformations such as the multivariate Box-Cox to simulate grouped datasets from more general distributions. CARP can also calculate the pairwise overlap between identified groups in clustered or classified datasets. It is only designed for cases involving continuous variables and not, in general, able to evaluate algorithms that cluster on manifolds and the like.

Acknowledgments

We acknowledge partial support for this project from the National Science Foundation (NSF) CAREER grant #DMS-0437555.

References

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

R. Maitra and V. Melnykov. Simulating data to study performance of finite mixture modeling and clustering algorithms. *Journal of Computational and Graphical Statistics*, 19(2):354–376, 2010. doi: 10.1198/jcgs.2009.08054.

W. Qiu and H. Joe. Generation of random clusters with specified degree of separation. *Journal of Classification*, 23:315–334, 2006.

D. Steinley and R. Henson. OCLUS: An analytic method for generating clusters with known overlap. *Journal of Classification*, 22:221–250, 2005.

Journal of Machine Learning Research 12 (2011) 75-110 Submitted 4/09; Revised 10/10; Published 1/11

Multitask Sparsity via Maximum Entropy Discrimination

Tony Jebara JEBARA@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
New York, NY 10027, USA

Editor: Francis Bach

Abstract

A multitask learning framework is developed for discriminative classification and regression where multiple large-margin linear classifiers are estimated for different prediction problems. These classifiers operate in a common input space but are coupled as they recover an unknown shared representation. A maximum entropy discrimination (MED) framework is used to derive the multitask algorithm which involves only convex optimization problems that are straightforward to implement. Three multitask scenarios are described. The first multitask method produces multiple support vector machines that learn a shared sparse feature selection over the input space. The second multitask method produces multiple support vector machines that learn a shared conic kernel combination. The third multitask method produces a pooled classifier as well as adaptively specialized individual classifiers. Furthermore, extensions to regression, graphical model structure estimation and other sparse methods are discussed. The maximum entropy optimization problems are implemented via a sequential quadratic programming method which leverages recent progress in fast SVM solvers. Fast monotonic convergence bounds are provided by bounding the MED sparsifying cost function with a quadratic function and ensuring only a constant factor runtime increase above standard independent SVM solvers. Results are shown on multitask data sets and favor multitask learning over single-task or tabula rasa methods.

Keywords: meta-learning, support vector machines, feature selection, kernel selection, maximum entropy, large margin, Bayesian methods, variational bounds, classification, regression, Lasso, graphical model structure estimation, quadratic programming, convex programming

1. Introduction

In applied domains ranging from biology to vision, inter-related data is collected by researchers for varying scientific purposes. While there are some concerted efforts to ensure that data sets are collected and labeled in consistent ways, it is often the case that many heterogeneous data sets over a given input domain are collected and labeled for different tasks. Most machine learning approaches take a *single-task* perspective where one large homogeneous repository of uniformly collected *iid* (independent and identically distributed) samples is given and labeled consistently. A more realistic, *multitask learning* approach is to combine data from multiple smaller sources and synergistically leverage heterogeneous labeling or annotation efforts.

Consider a group of biologists that are investigating the gene regulatory pathways of a simple species such as yeast. Each biologist may measure the expression levels of a different subset of genes under particular perturbation conditions of interest. In addition, the biologists may annotate or label the gene expression data they collect in different ways. Clearly, each data set has dependen-

©2011 Tony Jebara.

cies and redundancies when compared to another data set. Single-task learning from each data set in isolation (in a tabula rasa inductive manner) provides only a narrow view of the phenomenon at hand. Meanwhile, multitask learning (or inductive transfer) uses the collection of data sets simultaneously to exploit the related nature of the problems. For example, a multitask learning approach may involve algorithms that discover shared representations that are useful across several data sets and tasks. For instance, consider a group of doctors each interested in predicting the presence or absence of a particular disease from a set of medical tests that can be performed on a patient. Since medical tests may be invasive and expensive, the doctors may wish to find a small subset of medical tests (the shared representation) that can be performed on a patient once and for all such that each disease of interest can be accurately predicted.

This article explores maximum entropy discrimination approaches to multitask problems and is organized as follows. Section 2 reviews previous work in multitask learning, support vector machine feature selection and support vector machine kernel selection. Section 3 sets up the general multitask problem as learning from data that has been sampled from a set of generative models that are dependent given data observations yet become independent given a shared representation. Section 4 migrates the standard Bayesian treatment of the problem into a large-margin discriminative setting using maximum entropy. The log-linear model, the main classifier of interest in this article, is explicated in Section 5. Section 6 explicates the case where the shared representation is a binary feature selection that removes certain input space features in a consistent manner for all linear classification tasks. Section 7 extends the shared representation such that it explores any conic kernel combination with multiple linear classifiers. Section 8 extends the framework to adaptive data pooling problems. Section 9 illustrates the corresponding derivations in a multitask (scalar) regression setting. Section 10 briefly describes the sequential quadratic programming method which is to be applied to the convex programs derived in the various preceding sections. Experimental results are provided in Section 11. An extension that permits the approach to perform sparse graph structure estimation is described in Section 12 and Section 13 then concludes with a brief summary. The Appendix provides the derivation of a bound which converts all the necessary optimization steps into quadratic programming with a proof of fast convergence for the resulting sequential quadratic programming procedure. The Appendix also discusses connections to other sparse regression methods.

2. Previous Work

Since this article involves the combination of the three research areas, we review previous work in multitask learning, support vector machine (SVM) feature selection and SVM kernel selection.

Multitask learning has many names and incarnations including learning-to-learn, meta-learning, lifelong learning, and inductive transfer (Baxter, 1995; Thrun and Pratt, 1997; Caruana, 1997; Thrun, 1995). It goes beyond the usual assumptions in most learning methods which focus on learning a model from a single training data set. Instead, multitask learning couples multiple models and their individual training sets and tasks. The hope is that the models can benefit from each other synergistically if their tasks are inter-related (predicting if a face is male or female may help when predicting if a face belongs to an adult or a child), the distributions of the training sets are related (transformed versions of each other) or the general domains of the tasks are similar (for instance all tasks involve images of outdoor scenery). Early implementations of multitask learning primarily investigated neural network or nearest neighbor learners (Thrun, 1995; Baxter, 1995; Caruana, 1997). In addition to neural approaches, Bayesian methods have been explored that implement multitask

learning by assuming dependencies between the various models and tasks (Heskes, 1998, 2004). For instance, tasks can be clustered via a hierarchical mixture of Gaussians which couples their parameters. In addition, some theoretical arguments for the benefits of multitask learning have been made (Baxter, 2000) showing that the average error of M tasks can potentially decrease inversely with M. More recently, improved generalization guarantees for *each* individual task were provided if the classifiers are related and share a common structure (Ben-David and Schuller, 2003).

Concurrently, kernel methods (Schölkopf and Smola, 2001) and large-margin support vector machines are highly successful in single-task settings and are good candidates for multitask extensions. While multiclass variants of binary classifiers have been extensively explored (Crammer and Singer, 2001), multitask classification differs in that it often involves distinct sets of input data for each task. Furthermore, the concept of shared representation has been less practical to implement for kernel methods and support vector machines. For example, constraining the representation by performing SVM feature selection in a single-task setting may require extensions beyond standard quadratic programming (Jebara and Jaakkola, 2000; Weston et al., 2000). Similarly, constraining a representation to perform SVM kernel selection is also more involved in a single-task setting and requires second-order cone programming or semidefinite programming (Cristianini et al., 2001; Lanckriet et al., 2002).

This article focuses on multitask extensions of both feature selection and kernel selection with support vector machines. The derivations here will closely follow previous work which migrated maximum entropy to single-task SVMs (Jaakkola et al., 1999), to sparse SVMs (Jebara and Jaakkola, 2000) and to multitask SVMs (Jebara, 2003, 2004).[1] This maximum entropy framework led to one of the first convex large margin multitask classification approaches (Jebara, 2004). Convexity was subsequently explored in other multitask frameworks (Argyriou et al., 2008). The present article extends the derivations in the maximum entropy discrimination multitask approach, provides a straightforward iterative quadratic programming implementation and uses tighter bounds for improved runtime efficiency. Other related multitask SVM approaches have also been promising including novel kernel construction techniques to couple tasks (Evgeniou et al., 2005). These permit standard SVM learning algorithms to perform multitask learning while the multitask issues are handled primarily by the kernel itself. Even more recently, online algorithms have been proposed (Dekel et al., 2006) for multitask learning with margin-based predictors and provide interesting worst-case guarantees. Extensions to handle unlabeled data in multitask settings have also been promising (Ando and Zhang, 2005) and enjoyed theoretical generalization guarantees. An alternative perspective to multitask feature and kernel selection can be explored by performing joint covariate or subspace selection for multiple classification problems (Obozinski et al., 2010). Furthermore, feature selection and kernel selection can be seen as sparsity inducing methods. While a survey of sparsity is out of the scope of this article, one of the most popular implementations of sparsity or selection in regression settings is the ℓ_1 regularized Lasso method and its variants (Tibshirani, 1996; Tropp, 2006). Therein, sparsity is usually explored in a single-task setting and is used to remove unnecessary features in a regression problem (although sparsity is equally relevant in classification problems Jebara and Jaakkola, 2000). The multitask extension to such sparse estimation techniques is known as the Group Lasso and allows sparsity to be explored over predefined subsets of variables (Turlach et al., 2005; Yuan and Lin, 2006). Consistency arguments and connections between the Group Lasso and multiple kernel learning were also provided (Bach et al., 2004; Bach, 2008). Spar-

1. This article is the long version of a conference paper (Jebara, 2004).

sity and its connection to maximum entropy discrimination and so-called Laplace Markov networks was also recently explored (Zhu et al., 2008). This article provides another contact point between sparsity, large margins, multitask learning and kernel selection. The next sections formulate the general probabilistic setup for such multitask problems and convert traditional Bayesian solutions into a discriminative large-margin setting using the maximum entropy framework (Jaakkola et al., 1999).

3. Multitask Learning

The general multitask learning setup is as follows. We are given a collection of data sets $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ covering $m = 1 \ldots M$ tasks. Each task has its training set $\mathcal{D}_m$ of $t = 1 \ldots T_m$ input-output pairs $(\mathbf{x}_{m,t}, y_{m,t})$ that are independent and identically distributed (*iid*) samples from an unknown probability density function $\mathcal{P}_m$ defined jointly over both inputs and outputs. The data for task m is therefore $\mathcal{D}_m = \{(\mathbf{x}_{m,1}, y_{m,1}), \ldots, (\mathbf{x}_{m,T_m}, y_{m,T_m})\}$. The inputs may be in a Euclidean vector space $\mathbf{x}_{m,t} \in \mathbb{R}^D$ or, more generally, $\mathbf{x}_{m,t} \in \mathcal{X}$ are objects that could be mapped to a Hilbert space via a kernel. In a regression setting we assume the outputs are scalars $y_{m,t} \in \mathbb{R}$ while in a classification setting we would assume binary[2] outputs $y_{m,t} \in \{\pm 1\}$.

There are many ways to tie together multiple inter-related tasks synergistically. In this section and in Section 4, it will be helpful to take a Bayesian perspective to the multitask problem although this perspective is not strictly necessary in subsequent sections. From a Bayesian point of view, several model parameters will be estimated and assumed to be random variables governed by a distribution and priors. Assume that there are task-specific model parameters Θ_m associated to each task or data set $\mathcal{D}_m$ for $m = 1 \ldots M$. The single-task or tabula rasa learning approach assumes that the models are independent given their respective data sets and, therefore, can be recovered independently. Such an assumption may be too simple in practice. The more general multitask learning assumption is that there exist dependencies between the tasks. In other words, the likelihood of the models given the data does not factorize,

$$p(\Theta_1, \ldots, \Theta_M | \mathcal{D}) \quad \neq \quad \prod_{m=1}^{M} p(\Theta_m | \mathcal{D}_m).$$

One specific way of coupling the various parameters $\Theta_1, \ldots, \Theta_M$ is to instead assume that there is another parameter $\mathbf{s}$ that is shared across tasks. For example, $\mathbf{s}$ could be a set of binary switches that eliminate all but a few features in the input space. The models then become independent only if the shared parameter[3] or *representation* is observed as follows:

$$p(\Theta_1, \ldots, \Theta_M | \mathbf{s}, \mathcal{D}) \quad = \quad \prod_{m=1}^{M} p(\Theta_m | \mathbf{s}, \mathcal{D}_m).$$

Note that, given data, the models are conditionally independent given the representation yet are dependent otherwise. This lack of factorization is on the *posterior* when data is observed, not on

2. In this article, only the binary classification case will be considered, however, the techniques herein extend easily to multiclass settings where $y_{m,t} \in \{1, \ldots, Y\}$ with $Y \in \mathbb{Z}$ and $Y \geq 3$. Alternatively, it is straightforward to use binary classification methods on multiclass problems by using Y one-versus-all binary classifiers, by using $Y(Y-1)/2$ one-versus-one binary classifiers, or by using error-correcting codes (Dietterich and Bakiri, 1995).

3. A more general approach is to assume a hierarchy of shared variables which couples the various learning tasks in more subtle ways (Heskes, 1998; Dudik et al., 2007). This hierarchical setting is out of the scope of this article but is of interest for future work.

the *prior*. We may still make the assumption that $p(\Theta)$ factorizes a priori. However, observing data with a latent shared parameter $\mathbf{s}$ induces dependencies across the multiple tasks. In terms of a directed acyclic graph where the joint probability density function factorizes as a product of nodes given their parents, the following dependency structure emerges in the (simplest) case of multitask learning with two models: $\Theta_1 \rightarrow \mathcal{D}_1 \leftarrow \mathbf{s} \rightarrow \mathcal{D}_2 \leftarrow \Theta_2$. Therefore, observing the data $\mathcal{D}_1$ and $\mathcal{D}_2$ couples the two models unless the shared representation $\mathbf{s}$ is also observed.

Thus, a natural way of exploring dependencies between tasks is to assume a shared representation variable $\mathbf{s}$ is implicated in the learning problem. We then have a total set of parameters $\Theta = \{\Theta_1, \ldots, \Theta_M, \mathbf{s}\}$ to *jointly* estimate from all the data sets. We explore the following scenarios:

- **Feature Selection:** Consider M individual models $\Theta_m = \{\theta_m, b_m\}$ which are linear classifiers where $\theta_m \in \mathbb{R}^D$ and $b_m \in \mathbb{R}$. The shared representation $\mathbf{s} \in \mathbb{B}^D$ is a binary feature selection vector that either keeps ($\mathbf{s}(d) = 1$) or eliminates ($\mathbf{s}(d) = 0$) each input vector dimension.
- **Kernel Selection:** Consider M individual models $\Theta_m = \{\theta_{m,1}, \ldots, \theta_{m,D}, b_m\}$ where each model Θ_m consists of D linear classifiers in D different Hilbert spaces and one scalar $b_m \in \mathbb{R}$. The shared configuration $\mathbf{s} \in \mathbb{B}^D$ is a binary feature selection vector that either keeps (when $\mathbf{s}(d) = 1$) or eliminates (when $\mathbf{s}(d) = 0$) the candidate Hilbert space from the classifiers.
- **Adaptive Pooling:** Consider $M+1$ different linear classification models where M tasks have to choose between using their own specialized classifier $\theta_1, \ldots, \theta_M$ or a communal classifier θ by estimating $\mathbf{s} \in \mathbb{B}^M$, a binary selection vector.
- **Graphical Model Structure:** Consider estimating from sample data a graphical model structure over D random variables by finding D classifiers that predict each variable from all others.

The following sections detail these multitask learning scenarios and show how we can learn discriminative classifiers (that predict outputs accurately and with large margin) from multiple tasks. To tackle this problem, we will apply the maximum entropy discrimination framework (Jaakkola et al., 1999) since it produces convex optimization problems where global optima can be reliably recovered. Furthermore, the framework produces large margin discrimination and thus inherits the performance benefits of support vector machines.

4. Bayes and Maximum Entropy

The standard Bayesian approach to inference begins with a prior $p(\Theta)$ over a model class Θ (which can be possibly uncountable or continuous). The prior is then refined given the data to obtain a posterior $p(\Theta|\mathcal{D})$ via Bayes' rule $p(\Theta|\mathcal{D}) \propto p(\mathcal{D}|\Theta)p(\Theta)$. Subsequently, the posterior is used to make predictions for new observations. The Bayesian prediction of a label for a new query input $\mathbf{x}$ for task m is as follows:

$$\hat{y} = \arg\max_y \int p(y|\mathbf{x}, \Theta_m) p(\Theta|\mathcal{D}) d\Theta. \tag{1}$$

In the above, a prediction $\hat{y}$ is obtained from the predictive distribution $p(y|\mathbf{x}, \Theta_m)$ by integrating over all models Θ while weighing each predictive distribution by the posterior $p(\Theta|\mathcal{D})$. This posterior, according the Bayes rule, is simply the product of the prior and the likelihood as follows:

$$p(\Theta|\mathcal{D}) \quad = \quad \frac{1}{Z} p(\Theta) \prod_{m=1}^{M} \prod_{t=1}^{T_m} p(y_{m,t}|\mathbf{x}_{m,t}, \Theta_m).$$

Previous approaches (Heskes, 2004) followed such a Bayesian treatment for multitask learning and obtained promising results. In this article, however, we will modify the standard Bayesian posterior to learn a more discriminative solution. Instead of using Bayes' rule to infer the posterior, we consider a posterior which produces predictions $\hat{y}$ with *large margin* as in the support vector machine (SVM) framework (Cortes and Vapnik, 1995). In other words, we will construct a discriminative posterior density which yields both accurate classification and large margins when used in Equation 1. Accurate classification on the observed data is obtained by forcing the marginal likelihood of the correct label $y_{m,t}$ to be larger than that of incorrect labels for each observation $t = 1, \ldots, T_m$ in all $m = 1, \ldots, M$ data sets:

$$\int p(y_{m,t}|\mathbf{x}_{m,t}, \Theta_m) p(\Theta|\mathcal{D}) d\Theta - \max_{y \neq y_{m,t}} \int p(y|\mathbf{x}_{m,t}, \Theta_m) p(\Theta|\mathcal{D}) d\Theta \quad \geq \quad 0.$$

This ensures that the posterior gives good predictions on average since the correct label $y_{m,t}$ has a higher probability than the wrong label (Crammer and Singer, 2001; Taskar et al., 2004). The above constraints require that the likelihood of the correct label remain larger than the likelihood of the incorrect label on average under the posterior over Θ. We consider one additional simplification for computational considerations. Instead of comparing *likelihoods*, we will require that the *log-likelihood* of the correct label is larger than the *log-likelihood* of the incorrect label on average under the posterior over Θ. Furthermore, to achieve large margin, we will force the posterior to not only make correct predictions but to also produce a score for the correct label that is at least a constant γ above the value obtained by incorrect labels:

$$\int \log p(y_{m,t}|\mathbf{x}_{m,t}, \Theta_m) p(\Theta|\mathcal{D}) d\Theta - \max_{y \neq y_{m,t}} \int \log p(y|\mathbf{x}_{m,t}, \Theta_m) p(\Theta|\mathcal{D}) d\Theta \quad \geq \quad \gamma.$$

In many parts of this article, without loss of generality, we will assume that $\gamma = 1$. These correct-classification constraints are applied to all training data $t = 1, \ldots, T_m$ for all tasks $m = 1 \ldots M$. Such classification or discrimination constraints were first introduced in the so-called maximum entropy discrimination (MED) framework (Jaakkola et al., 1999) and give rise to posterior distributions that mimic support vector machines and large-margin learning. The MED framework also conveniently leads to analytic expressions and closed-form solutions for all the necessary integrals. Instead of using Bayes rule to obtain the posterior, MED finds a posterior that is as close as possible to the prior in terms of Kullback-Leibler Divergence. In other words, it minimizes the relative entropy to the prior $\mathrm{KL}(p(\Theta|\mathcal{D}) \| p(\Theta))$ but still also *satisfies the above classification constraints*. This produces the following primal optimization problem:

$$O_{primal} \left\{ \begin{array}{l} \min_{P(\Theta|\mathcal{D})} \mathrm{KL}(p(\Theta|\mathcal{D}) \| p(\Theta)) \\ \text{s.t. } \int \log \left(\frac{p(y_{m,t}|\mathbf{x}_{m,t}, \Theta_m)}{p(y|\mathbf{x}_{m,t}, \Theta_m)} \right) p(\Theta|\mathcal{D}) d\Theta \geq \gamma \quad \forall y \neq y_{m,t}, m, t. \end{array} \right.$$

The solution is straightforward and gives the following posterior:

$$p(\Theta|\mathcal{D}) \quad = \quad \frac{1}{Z(\boldsymbol{\lambda})} p(\Theta) \prod_{m=1}^{M} \prod_{t=1}^{T_m} \prod_{y \neq y_{m,t}} \left(\frac{p(y_{m,t}|\mathbf{x}_{m,t}, \Theta_m)}{p(y|\mathbf{x}_{m,t}, \Theta_m)} \right)^{\lambda_{m,t}} \exp(-\gamma \lambda_{m,t}). \tag{2}$$

Here, $\boldsymbol{\lambda}$ is a collection (or a vector) of non-negative Lagrange multipliers $\{\lambda_{m,t}\}$ for $m = 1, \ldots, M$ and $t = 1, \ldots, T_m$ that are used to enforce the inequality constraints. The normalizer for the above

posterior is $Z(\boldsymbol{\lambda})$. Maximum entropy solves for the Lagrange multipliers by maximizing $J(\boldsymbol{\lambda}) = -\log Z(\boldsymbol{\lambda})$. This is simply the dual optimization

$$O_{dual}\left\{ \max_{\boldsymbol{\lambda}\geq\mathbf{0}} -\log \int p(\Theta)\prod_{m=1}^{M}\prod_{t=1}^{T_m}\prod_{y\neq y_{m,t}}\left(\frac{p(y_{m,t}|\mathbf{x}_{m,t},\Theta_m)}{p(y|\mathbf{x}_{m,t},\Theta_m)}\right)^{\lambda_{m,t}} \exp(-\gamma\lambda_{m,t})d\Theta.\right.$$

If all $\lambda_{m,t}$ are set to 1, the posterior resembles the standard Bayesian estimate. However, MED estimates different weights $\lambda_{m,t}$ for each datum (or classification constraint) in the posterior. This ensures that the classification constraints are achieved. Instead of treating all points equally, the MED solution explores weights on each datum to adjust the Bayesian solution such that it obtains better classification on the training data. The expected *log*-likelihood of the data under the MED posterior satisfies the classification constraints while staying close to the prior. Furthermore, MED uses the expected *log*-likelihood of a new query point to make predictions as follows:

$$\hat{y} = \arg\max_{y} \mathrm{E}_{p(\Theta|\mathcal{D})}[\log p(y|\mathbf{x},\Theta_m)] = \arg\max_{y} \int \log p(y|\mathbf{x},\Theta_m)p(\Theta|\mathcal{D})d\Theta.$$

This simple reformulation of the standard Bayesian posterior will give rise to large margin learning as explicated in the next section.

5. From Log-Linear Models to Support Vector Machines

We next make more specific assumptions on the form of the predictive distribution $p(y|\mathbf{x},\Theta_m)$. Assume that the predictive distribution is log-linear as follows:

$$p(y|\mathbf{x},\Theta_m) \quad \propto \quad \exp\left(\frac{y}{2}\langle \mathbf{x},\theta_m\rangle + b_m)\right).$$

This permits us to rewrite the above posterior $p(\Theta|\mathcal{D})$ more specifically as:

$$p(\Theta|\mathcal{D}) \quad = \quad \frac{1}{Z(\boldsymbol{\lambda})}p(\Theta)\prod_{m=1}^{M}\prod_{t=1}^{T_m}\exp(y_{m,t}(\langle \mathbf{x}_{m,t},\theta_m\rangle + b_m))^{\lambda_{m,t}}\exp(-\gamma\lambda_{m,t}).$$

We integrate the above over $\Theta = \{\Theta_1,\ldots,\Theta_M,\mathbf{s}\}$ to obtain the partition function $Z(\boldsymbol{\lambda})$. The objective function we need to maximize is the negative logarithm of the partition function:

$$J(\boldsymbol{\lambda}) \quad = \quad -\log\int p(\Theta)\exp\left(\sum_{m=1}^{M}\sum_{t=1}^{T_m}\lambda_{m,t}y_{m,t}(\langle \mathbf{x}_{m,t},\theta_m\rangle + b_m) - \gamma\lambda_{m,t}\right)d\Theta.$$

We will next *assume* that the prior over models factorizes as follows:

$$p(\Theta) \quad = \quad p(\mathbf{s})\prod_{m=1}^{M}p(\Theta_m) = p(\mathbf{s})\prod_{m=1}^{M}\mathcal{N}(\theta_m|\mathbf{0},I)\mathcal{N}(b_m|0,\sigma^2).$$

and assume that the priors over parameters are all white Gaussians with zero mean and identity covariance (take $\mathbf{0}$ to be the vector of all zeros and I to be the identity matrix). This factorization assumption on the prior will be kept throughout this article. Although the prior factorizes, this does not necessarily mean that the posterior will factorize too. The likelihood terms in Equation 2 may

couple the models in the posterior. However, in this first example we will not obtain any coupling per se. This is clear once we evaluate the integrals to obtain the objective function:

$$\begin{aligned} J(\boldsymbol{\lambda}) &= -\sum_{m=1}^{M} \log \int \exp(\langle \theta_m, \sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} \mathbf{x}_{m,t} \rangle) \mathcal{N}(\theta_m | \mathbf{0}, I) d\theta_m \\ &\quad -\sum_{m=1}^{M} \log \int \exp(b_m \sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t}) \mathcal{N}(b_m | 0, \sigma^2) db_m + \sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma \lambda_{m,t}. \end{aligned}$$

Simple algebra and completion of squares[4] yields the objective function $J(\boldsymbol{\lambda})$ which is maximized as follows

$$\max_{\boldsymbol{\lambda} \geq \mathbf{0}} \sum_{m=1}^{M} \left(\sum_{t=1}^{T_m} \gamma \lambda_{m,t} - \frac{1}{2} \sum_{t,\tau=1}^{T_m} \lambda_{m,t} \lambda_{m,\tau} y_{m,t} y_{m,\tau} \langle \mathbf{x}_{m,t}, \mathbf{x}_{m,\tau} \rangle - \frac{\sigma^2}{2} \left(\sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} \right)^2 \right).$$

The above dual optimization problem is simply a quadratic program and is straightforward to solve. If we further assume that $\sigma^2 \to \infty$, which corresponds to using a non-informative prior on the bias scalar terms b_m, the objective function above gives the constraints $\sum_t \lambda_{m,t} y_{m,t} = 0$ for all $m = 1 \ldots M$. We then get an objective function that is exactly the sum of the dual objective functions of M independent support vector machines (if $\gamma = 1$). Thus, our dual optimization is:

$$\max_{\boldsymbol{\lambda} \geq \mathbf{0}} \quad \sum_{m=1}^{M} \left(\sum_{t=1}^{T_m} \gamma \lambda_{m,t} - \frac{1}{2} \sum_{t,\tau=1}^{T_m} \lambda_{m,t} \lambda_{m,\tau} y_{m,t} y_{m,\tau} k(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau}) \right) \text{ s.t. } \sum_{t=1}^{T_m} y_{m,t} \lambda_{m,t} = 0 \ \forall m.$$

Here, we have also replaced all inner products of two inputs $\mathbf{x}$ and $\mathbf{x}$ of the form $\langle \mathbf{x}, \mathbf{x} \rangle$ with Mercer kernel evaluations $k(\mathbf{x}, \mathbf{x})$. This allows us to readily accommodate nonlinear classification. Finally, the prediction rule for a query input $\mathbf{x}$ given the current setting of the $\boldsymbol{\lambda}$ values for the m'th model involves integrating over the posterior which produces the following prediction:

$$\hat{y} = \arg\max_{y} \mathrm{E}_{p(\Theta|\mathcal{D})}[\log p(y|\mathbf{x}, \Theta_m)] = \text{sign} \left(\sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} k(\mathbf{x}, \mathbf{x}_{m,t}) + \hat{b}_m \right),$$

where the $\hat{b}_m$ scalars are given by the Karush Kuhn Tucker (KKT) conditions. Whenever a constraint or Lagrangian is active, the corresponding Lagrange multiplier must be strictly positive $\lambda_{m,t} > 0$ and we expect the inequalities in the primal problem to hold exactly. Therefore, we can obtain each $\hat{b}_m$ by solving

$$y_{m,t} = \sum_{\tau=1}^{T_m} \lambda_{m,\tau} y_{m,\tau} k(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau}) + \hat{b}_m$$

for any datum t which has a corresponding Lagrange multiplier (once the dual program halts) that satisfies $\lambda_{m,t} > 0$.

Clearly, because of additivity, updating $\lambda_{m,1}, \ldots, \lambda_{m,T_m}$ can be done independently of $\lambda_{n,1}, \ldots, \lambda_{n,T_n}$ for any $n \neq m$. In other words, we have tabula rasa independent learning of M independent SVMs on all the tasks. Even the scalar biases $\hat{b}_m$ are obtained independently via the KKT conditions. Therefore, to obtain multitask learning, we will need some shared representation $\mathbf{s}$ to couple the learning problems and give rise to a non-factorized posterior over models.

4. Recall that $\int \exp(\langle \theta, w \rangle) \mathcal{N}(\theta | \mathbf{0}, I) d\theta = \exp(\langle w, w \rangle / 2)$.

5.1 Non-Separable Case

For thoroughness, this section details the case where the classification problems are not separable; in other words, not all inequalities in the maximum entropy formulation can be achieved. In this case, we introduce non-negative slack variables $\xi = \{\xi_{m,t}\}$ on each constraint with a cost of C per unit of slack leading to the following primal optimization:

$$O_{primal} \left\{ \begin{array}{l} \min_{P(\Theta|\mathcal{D}),\xi} \mathrm{KL}(p(\Theta|\mathcal{D})\|p(\Theta)) + C\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{y\neq y_{m,t}}\xi_{m,t,y} \\ \text{s.t. } \int \log\left(\frac{p(y_{m,t}|\mathbf{x}_{m,t},\Theta_m)}{p(y|\mathbf{x}_{m,t},\Theta_m)}\right) p(\Theta|\mathcal{D})d\Theta \geq \gamma - \xi_{m,t} \text{ and } \xi_{m,t,y} \geq 0 \;\; \forall y \neq y_{m,t}, m, t. \end{array} \right.$$

The above produces the same type of solution as Equation 2 but has a slightly different dual optimization:

$$O_{dual} \left\{ \begin{array}{l} \max_{\lambda} \; \sum_{m=1}^{M}\left(\sum_{t=1}^{T_m}\gamma\lambda_{m,t} - \frac{1}{2}\sum_{t,\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}k(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\right) \\ \text{s.t. } 0 \leq \lambda_{m,t} \leq C \;\; \forall m,t \text{ and } \sum_{t=1}^{T_m} y_{m,t}\lambda_{m,t} = 0 \;\; \forall m \end{array} \right.$$

which merely bounds the Lagrange multipliers from above by C. Once again, MED mimics support vector machines (Cortes and Vapnik, 1995) in the non-separable case.

6. Feature Selection

We next explore feature selection and require $\mathbf{x} \in \mathbb{R}^D$ where $D \in \mathbb{Z}$. To couple the tasks, modify the predictive distribution for the label given the model such that it also depends on a *shared* variable $\mathbf{s}$ as follows:

$$p(y|\mathbf{x},\Theta_m,\mathbf{s}) \quad \propto \quad \exp\left(\frac{y}{2}\left(\sum_{d=1}^{D}\mathbf{s}(d)\mathbf{x}(d)\theta_m(d) + b_m\right)\right),$$

where $\mathbf{s}$ is a binary D-dimensional vector and the argument of a vector $\mathbf{s}(d)$ refers to its d'th entry. Thus, the shared representation consists of binary switches that delete or censor various entries of the $\mathbf{x}$ vector. If $\mathbf{s}(d) = 0$, then the $\mathbf{x}(d)$ entry is effectively set to zero. Meanwhile, if $\mathbf{s}(d) = 1$, the $\mathbf{x}(d)$ entry remains intact. In other words, the binary vector $\mathbf{s}$ performs a feature selection. In addition, assume the prior for $p(\mathbf{s})$ is a product of Bernoulli distributions for each element of $\mathbf{s}$,

$$p(\mathbf{s}) \quad = \quad \prod_{d=1}^{D}\rho^{\mathbf{s}(d)}(1-\rho)^{1-\mathbf{s}(d)},$$

where ρ is the a priori probability of keeping the features on. For example, setting $\rho = 1$ suggests that all features should be on and no feature selection is to be performed. Alternatively, we can reparametrize the prior as $\alpha = \frac{1-\rho}{\rho}$ where increasing α corresponds to sparser feature selection. A value of $\alpha = 0$ indicates no feature selection is being performed (no sparsity). Meanwhile a value of $\alpha \to \infty$ encourages the models to discard almost all features. If we perform feature selection and use a predictive distribution with shared $\mathbf{s}$, the $m = 1,\ldots,M$ tasks will become coupled and the posterior over models no longer factorizes. The MED solution is then

$$p(\Theta|\mathcal{D}) \quad = \quad \frac{1}{Z(\boldsymbol{\lambda})}p(\Theta)\prod_{m=1}^{M}\prod_{t=1}^{T_m}\exp\left(\lambda_{m,t}y_{m,t}\left(\sum_{d=1}^{D}\mathbf{s}(d)\mathbf{x}_{m,t}(d)\theta_m(d) + b_m\right) - \gamma\lambda_{m,t}\right).$$

We compute the corresponding partition function by integrating over all models $\Theta_1, \ldots, \Theta_m$ as well as summing over all binary settings of $\mathbf{s}$ which yields

$$\begin{aligned} Z(\boldsymbol{\lambda}) &= \int p(\Theta) \exp\left(\sum_{m=1}^{M} \sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} \left(\sum_{d=1}^{D} \mathbf{s}(d) \mathbf{x}_{m,t}(d) \theta_m(d) + b_m \right) - \gamma \lambda_{m,t} \right) d\Theta \\ &= \exp\left(\sum_m \frac{\sigma^2}{2} \left(\sum_t \lambda_{m,t} y_{m,t} \right)^2 - \sum_t \gamma \lambda_{m,t} \right) \prod_d \left(1 - \rho + \rho e^{\frac{1}{2} \sum_m \left(\sum_t \lambda_{m,t} y_{m,t} \mathbf{x}_{m,t}(d) \right)^2} \right). \end{aligned}$$

Taking $\sigma^2 \to \infty$ gives a new objective function $J(\boldsymbol{\lambda}) = -\log(Z(\boldsymbol{\lambda}))$ which is no longer a quadratic program yet is still a convex program as follows:

$$\begin{cases} \max_{\boldsymbol{\lambda}} \ \sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma \lambda_{m,t} - \sum_{d=1}^{D} \log\left(\alpha + e^{\frac{1}{2} \sum_{m=1}^{M} \left(\sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} \mathbf{x}_{m,t}(d) \right)^2} \right) + D \log(\alpha + 1) \\ \text{s.t. } 0 \leq \lambda_{m,t} \leq C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m} y_{m,t} \lambda_{m,t} = 0 \ \ \forall m. \end{cases}$$

Note the property that $J(\mathbf{0}) = 0$. Clearly, the objective function is no longer additive across $m = 1 \ldots M$ which means that learning is coupled across tasks. This is due to the non-linearity in the function $f(x) = \log(\alpha + \exp(-x))$ which involves a summation over $m = 1 \ldots M$. We will refer to this function as the log-sigmoid function. If we set $\alpha = 0$, the log-sigmoid becomes linear and we get back the independent optimization problems in Section 5. Therein, the tasks decouple completely (i.e., the objective function becomes additive over tasks $m = 1 \ldots M$). However, larger settings of α encourage some coupling between the SVMs (or large margin log-linear models) as they search for a joint feature selection.

Note the presence of logarithmic terms which prevent the direct application of quadratic programming to $J(\boldsymbol{\lambda})$. Fortunately, the log-sigmoid function $f(x) = \log(\alpha + \exp(-x))$ is known to be a convex function (more precisely, our objective involves a negated sum of such functions which is concave overall). Recently, new computational tools have been proposed for solving convex programs that involve such terms (Koh et al., 2007). In our implementation, we instead apply a bound on the log-sigmoid to reformulate the optimization as a sequential quadratic program. Optimization details are deferred to Section 10 but it will be assumed that a (nearly) optimal $\boldsymbol{\lambda}$ solution can be recovered.

Given the recovered $\boldsymbol{\lambda}$ setting, the prediction rule is straightforward to derive as follows:

$$\hat{y} = \arg\max_y \mathrm{E}_{p(\Theta|\mathcal{D})}[\log p(y|\mathbf{x}, \Theta_m, \mathbf{s})] = \mathrm{sign}\left(\sum_{d=1}^{D} \sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} \hat{\mathbf{s}}(d) \mathbf{x}(d) \mathbf{x}_{m,t}(d) + \hat{b}_m \right).$$

The $\hat{\mathbf{s}}(d)$ above are expected values of $\mathbf{s}(d)$ under the posterior and are given by:

$$\hat{\mathbf{s}}(d) = \frac{1}{1 + \alpha \exp\left(-\frac{1}{2} \sum_{m=1}^{M} \left(\sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} \mathbf{x}_{m,t}(d) \right)^2 \right)}.$$

In fact, $\hat{\mathbf{s}}(d)$ are scalars in $[1/(1+\alpha), 1]$ which give a soft feature selection as values close to the bottom of the range are candidates for removal after thresholding. The values of $\hat{\mathbf{s}}(d)$ multiplicatively scale the input domain features and are close to $1/(1+\alpha)$ for features that are not useful for prediction in the multiple tasks. Once again, the $\hat{b}_m$ scalars are given by the KKT conditions which require that the value inside the sign() function evaluation above exactly equals $y_{m,t}$ for the query $\mathbf{x} = \mathbf{x}_{m,t}$ whenever the corresponding Lagrange multiplier strictly satisfies $0 < \lambda_{m,t} < C$.

7. Kernel Selection

Feature selection and sparsity are not the only types of shared representation $\mathbf{s}$ one may consider. One crucial design issue of nonlinear SVMs is the choice of a kernel function. Also, kernels permit SVMs to handle non-vectorial inputs so we relax the assumption that the $\mathbf{x}$ inputs are Euclidean vectors and only require that they are objects from some sample space $\mathbf{x}_{m,t} \in \mathcal{X}$ for all $m = 1,\ldots,M$ and $t = 1,\ldots,T_m$. Typically, in kernel learning (Lanckriet et al., 2002), we are given a set of $d = 1,\ldots,D$ base Mercer kernels $k_1,\ldots,k_D$ where each kernel function $k_d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ accepts two inputs and produces a scalar. We wish to learn a conic combination of the kernels or a sparse selection using the non-negative scalar weights $w_1,\ldots,w_D$ as follows

$$K(\mathbf{x},\mathbf{x}) \quad = \quad \sum_{d=1}^{D} w_d k_d(\mathbf{x},\mathbf{x}).$$

Some base kernels may get a small weight and are thus not selected and others will be averaged with varying weights w_d to produce a potentially better final kernel K. Each base kernel $k_d(\mathbf{x},\mathbf{x})$ can be seen to correspond to a mapping ϕ_d which is applied to both inputs $\mathbf{x}$ and $\mathbf{x}$. The function ϕ_d maps an input $\mathbf{x} \in \mathcal{X}$ to some Hilbert space we denote Φ_d. The kernel is then the inner-product of $\phi_d(\mathbf{x})$ and $\phi_d(\mathbf{x})$ as follows:

$$k_d(\mathbf{x},\mathbf{x}) \quad = \quad \langle \phi_d(\mathbf{x}), \phi_d(\mathbf{x}) \rangle.$$

Kernel selection is equivalent to selecting some mappings and attenuating others. We thus need a shared representation vector $\mathbf{s}$ which is again binary and again D-dimensional to select which kernels will be used. However, now, we have a set of $M \times D$ linear models $\theta_{m,d} \in \Phi_d^*$ for each Hilbert space. A $\theta_{m,d}$ vector is available for each task $m = 1,\ldots,M$ and each mapping $d = 1,\ldots,D$. In other words, task m has the following modeling resources on its own: $\Theta_m = \{\theta_{m,1},\ldots,\theta_{m,D}, b_m\}$. The prior for the modeling resources for the m'th task is then chosen to be a product of independent white Gaussians on these D vector parameters. This leads to the following general prior for all model parameters:

$$p(\Theta) \quad = \quad \prod_{d=1}^{D} \rho^{\mathbf{s}(d)} (1-\rho)^{1-\mathbf{s}(d)} \prod_{m=1}^{M} \mathcal{N}(\theta_{m,d}|\mathbf{0}, I) \mathcal{N}(b_m|0,\sigma^2).$$

Once again, all tasks share and have to agree on the binary selector vector $\mathbf{s}$ which inherits the Bernoulli prior used in the previous section. Therefore, we have the following total set of parameters $\Theta = \{\Theta_1,\ldots,\Theta_M,\mathbf{s}\}$.

The predictive distribution for multitask kernel selection is then given by the following log-linear model (for the m'th task):

$$p(y|\mathbf{x},\Theta_m,\mathbf{s}) \quad \propto \quad \exp\left(\frac{y}{2}\left(\sum_{d=1}^{D} \mathbf{s}(d)\langle \theta_{m,d}, \phi_d(\mathbf{x})\rangle + b_m\right)\right).$$

We once again recover the MED posterior using Equation 2. The normalizer for the posterior $Z(\boldsymbol{\lambda})$ is then found by integrating over the parameters Θ. This multitask kernel selection objective function $J(\boldsymbol{\lambda})$ is the following convex program:

$$\begin{cases} \max_{\boldsymbol{\lambda}} \ \sum_{m=1}^{M}\sum_{t=1}^{T_m} \gamma\lambda_{m,t} + D\log(\alpha+1) \\ \qquad -\sum_{d=1}^{D} \log\left(\alpha + e^{\frac{1}{2}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m} \lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})}\right) \\ \text{s.t. } 0 \le \lambda_{m,t} \le C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m} y_{m,t}\lambda_{m,t} = 0 \ \ \forall m. \end{cases}$$

Similarly, given the $\boldsymbol{\lambda}$ setting, we obtain the following prediction rule:

$$\hat{y} \quad = \quad \text{sign}\left(\sum_{d=1}^{D}\sum_{t=1}^{T_m}\lambda_{m,t}y_{m,t}\hat{\mathbf{s}}(d)k_d(\mathbf{x},\mathbf{x}_{m,t})+\hat{b}_m\right)$$

where $\hat{\mathbf{s}}(d)$ are scalars that weight each kernel and are usually close to $1/(1+\alpha)$ for kernels that are not beneficial for our multiple classification tasks. The weights for each kernel are recovered as:

$$\hat{\mathbf{s}}(d) \quad = \quad \frac{1}{1+\alpha\exp\left(-\frac{1}{2}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\right)}.$$

The scalar biases $\hat{b}_m$ are once again recovered from the KKT conditions. From the prediction rule, it is clear that kernel learning is effectively creating a new kernel from the base kernels as follows:

$$K(\mathbf{x},\mathbf{x}) \quad = \quad \sum_{d=1}^{D}\hat{\mathbf{s}}(d)k_d(\mathbf{x},\mathbf{x}).$$

When $\alpha = 0$, there is no coupling of tasks or sparse selection of kernels. The solution simply corresponds to setting $\hat{\mathbf{s}}(d) = 1$ and forces the final kernel K to equal a simple sum of all base kernels for $d = 1,\ldots,D$. In general, however, a more appropriate final kernel could potentially be recovered if $\alpha > 0$. Given such an aggregate kernel $K(\mathbf{x},\mathbf{x})$, we can now write an SVM-like prediction rule for the m'th task:

$$\hat{y} \quad = \quad \text{sign}\left(\sum_{t=1}^{T_m}\lambda_{m,t}y_{m,t}K(\mathbf{x},\mathbf{x}_{m,t})+\hat{b}_m\right).$$

Another interesting fact is that feature selection is just an instance of kernel selection. If we choose the $d = 1,\ldots,D$ kernels as follows

$$k_d(\mathbf{x},\mathbf{x}) \quad = \quad \mathbf{x}(d)\mathbf{x}(d).$$

we are effectively replacing kernel evaluations in this section with the scalar product of the d'th dimension of the input that was needed for feature selection. Thus, the kernel selection problem in this section clearly subsumes the feature selection problem derived in Section 6.

7.1 Independent Kernel Selection

It is possible to break the above multitask framework by allowing each task to select a combination of kernels independently. This means that we introduce a separate $\mathbf{s}_m$ vector for each task $m = 1,\ldots,M$ instead of having a shared representation $\mathbf{s}$. The derivation is straightforward and produces the following convex program:

$$\begin{cases} \max_{\boldsymbol{\lambda}} \ \sum_{m=1}^{M}\sum_{t=1}^{T_m}\gamma\lambda_{m,t} + MD\log(\alpha+1) \\ \qquad -\sum_{m=1}^{M}\sum_{d=1}^{D}\log\left(\alpha + e^{\frac{1}{2}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})}\right) \\ \text{s.t. } 0 \le \lambda_{m,t} \le C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m} y_{m,t}\lambda_{m,t} = 0 \ \ \forall m \end{cases}$$

which is once again additive in $m = 1,\ldots,M$ indicating that the Lagrange multipliers for each task are estimated independently in a tabula rasa learning method. As usual, the prediction rule is

given by $\hat{y} = \arg\max_y \mathrm{E}_{p(\Theta|\mathcal{D})}[\log p(y|\mathbf{x},\Theta_m)]$ and the following formula emerges for the expected switches:

$$\hat{\mathbf{s}}_m(d) \quad = \quad \frac{1}{1+\alpha\exp\left(-\frac{1}{2}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\right)}.$$

The prediction function for each task then simply uses its own $\hat{\mathbf{s}}_m(d)$ weights to combine the base kernels. This approach resembles the multiple kernel learning method (Lanckriet et al., 2002) since each task performs its own kernel selection in isolation.

7.2 Metric Learning

It is known that a Mercer kernel $k(\mathbf{x},\mathbf{x})$ or affinity can be used to construct a distance metric $\Delta(\mathbf{x},\mathbf{x})$ that satisfies standard requirements such as the triangle inequality. Consider constructing a base distance metric $\Delta_d(\mathbf{x},\mathbf{x})$ from each base kernel $k_d(\mathbf{x},\mathbf{x})$ as follows:

$$\Delta_d(\mathbf{x},\mathbf{x}) \quad = \quad \sqrt{k_d(\mathbf{x},\mathbf{x}) - 2k_d(\mathbf{x},\mathbf{x}) + k_d(\mathbf{x},\mathbf{x})}.$$

Given this multitask kernel selection framework, it is possible to use the above formula to perform multitask metric learning. By applying the algorithm in Section 7, we obtain the kernel weights $\hat{\mathbf{s}}(1),\ldots,\hat{\mathbf{s}}(D)$. This permits us to learn an overall kernel as a conic combination of the set of base kernels. This solution can then be mapped into a learned distance metric as follows:

$$\Delta(\mathbf{x},\mathbf{x}) \quad = \quad \sqrt{\sum_{d=1}^{D}\hat{\mathbf{s}}(d)\left(\Delta_d(\mathbf{x},\mathbf{x})\right)^2}.$$

Thus, metric learning can be performed using the multitask kernel selection setup. Once a new kernel is learned, it is then possible to reconstruct the corresponding distance metric and apply any kernel or distance-based learning algorithm. For instance, kernel principal components analysis (Schölkopf et al., 1999) or any distance-based learning algorithm such as kernel nearest neighbors and kernel clustering can be used with such learned kernels and distance functions.

8. Shared Classifiers and Adaptive Pooling

Another interesting multitask learning approach involves shared classifiers or shared models. For example, if we have very few training examples for each task, we may consider *pooling* all tasks together and learning a single classifier for all. This may help initially yet some tasks with more training examples than others may want to specialize and form their own independent classifiers once we are confident these tasks have enough supporting data. Once again assume we have $m = 1,\ldots,M$ tasks. These tasks have to choose between using their own specialized classifier $\theta_1,\ldots,\theta_M$ or a communal classifier θ. To avoid a trivial solution, only some of the tasks are allowed to become specialized and use their own linear model. Consider a binary feature selection vector $\mathbf{s} \in \mathbb{B}^M$. For each task m, the element of the vector $\mathbf{s}(m) \in \mathbb{B}$ determines if the task will use its own specialized θ_m model (when $\mathbf{s}(m) = 1$) or use the communal θ model (when $\mathbf{s}(m) = 0$) for discrimination. This setup is clarified by the following log-linear predictive distribution for the m'th task:

$$p(y|m,\mathbf{x},\Theta,\mathbf{s}) \quad \propto \quad \exp\left(\frac{y}{2}\left(\mathbf{s}(m)\langle\theta_m,\phi_m(\mathbf{x})\rangle + \langle\theta,\phi(\mathbf{x})\rangle + b_m\right)\right).$$

The communal classifier is over a single Hilbert space mapping $\phi(\mathbf{x})$ while the specialized classifiers may operate over their own distinct Hilbert space mapping $\phi_m(\mathbf{x})$. Inner products in these Hilbert spaces are computed using kernels as usual $k(\mathbf{x},\mathbf{x}) = \langle \phi(\mathbf{x}), \phi(\mathbf{x}) \rangle$ and $k_m(\mathbf{x},\mathbf{x}) = \langle \phi_m(\mathbf{x}), \phi_m(\mathbf{x}) \rangle$. Another subtlety is that each task still has its own dedicated b_m constant scalar bias. The complete set of models is therefore $\Theta = \{\theta, \theta_1, \ldots, \theta_M, b_1, \ldots, b_M\}$. We can assume the priors on all models are white Gaussian distributions. We also continue to use Bernoulli priors for $\mathbf{s}(m)$ and zero-mean Gaussian priors for the biases. The normalizer for the posterior is recovered as:

$$\begin{aligned} Z(\boldsymbol{\lambda}) &= \int p(\Theta) e^{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} (\mathbf{s}(m) \langle \theta_m, \phi_m(\mathbf{x}_{m,t}) \rangle + \langle \theta, \phi(\mathbf{x}_{m,t}) \rangle + b_m) - \gamma \lambda_{m,t}} d\Theta \\ &= e^{\sum_m \frac{\sigma^2}{2} (\sum_t \lambda_{m,t} y_{m,t})^2} e^{-\sum_m \sum_t \gamma \lambda_{m,t}} e^{\frac{1}{2} \sum_m \sum_n \sum_t \sum_\tau \lambda_{m,t} \lambda_{n,\tau} y_{m,t} y_{n,\tau} k(\mathbf{x}_{m,t}, \mathbf{x}_{n,\tau})} \\ &\quad \times \sum_{\mathbf{s}(1)} \cdots \sum_{\mathbf{s}(M)} p(\mathbf{s}) \prod_m e^{\frac{1}{2} \mathbf{s}(m) \sum_t \sum_\tau \lambda_{m,t} \lambda_{m,\tau} y_{m,t} y_{m,\tau} k_m(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau})}. \end{aligned}$$

The final summations over the binary switch settings above distribute and become straightforward. We assume that $\sigma \to \infty$ and obtain the following objective function $J(\boldsymbol{\lambda})$

$$\begin{cases} \max_{\boldsymbol{\lambda}} \sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma \lambda_{m,t} - \frac{1}{2} \sum_{m=1}^{M} \sum_{n=1}^{M} \sum_{t=1}^{T_m} \sum_{\tau=1}^{T_n} \lambda_{m,t} \lambda_{n,\tau} y_{m,t} y_{n,\tau} k(\mathbf{x}_{m,t}, \mathbf{x}_{n,\tau}) \\ \quad - \sum_{m=1}^{M} \log \left(\alpha + e^{\frac{1}{2} \sum_{t=1}^{T_m} \sum_{\tau=1}^{T_m} \lambda_{m,t} \lambda_{m,\tau} y_{m,t} y_{m,\tau} k_m(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau})} \right) + M \log(\alpha + 1) \\ \text{s.t. } 0 \leq \lambda_{m,t} \leq C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m} y_{m,t} \lambda_{m,t} = 0 \ \ \forall m \end{cases}$$

which clearly shows that the tasks cannot be solved independently (since the quadratic term above sums over both m and n which couples all pairs of tasks). The solution of the above is once again a convex program. Given the optimal Lagrange multiplier solution, the prediction rule for an input $\mathbf{x}$ for the m'th task is given by:

$$\hat{y} = \text{sign} \left(\hat{\mathbf{s}}(m) \sum_{t=1}^{T_m} \lambda_{m,t} y_{m,t} k_m(\mathbf{x}, \mathbf{x}_{m,t}) + \sum_{n=1}^{M} \sum_{t=1}^{T_n} \lambda_{n,t} y_{n,t} k(\mathbf{x}, \mathbf{x}_{n,t}) + \hat{b}_m \right).$$

We recover the expected $\mathbf{s}(m)$ value which measures our confidence in using a specialized classifier for the m'th task as follows

$$\hat{\mathbf{s}}(m) = \frac{1}{1 + \alpha \exp \left(-\frac{1}{2} \sum_{t=1}^{T_m} \sum_{\tau=1}^{T_m} \lambda_{m,t} \lambda_{m,\tau} y_{m,t} y_{m,\tau} k_m(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau}) \right)}.$$

It is interesting to note that if α is infinity, then all the $\hat{\mathbf{s}}(m)$ values go to zero and the method performs complete pooling. Conversely, if $\alpha = 0$, then $\hat{\mathbf{s}}(m) = 1$ and each classifier mixes its specialized linear model equally with the communal model. It is natural to use a smaller scale for $k(\mathbf{x},\mathbf{x})$ than $k_m(\mathbf{x},\mathbf{x})$ such that the choice $\alpha = 0$ leads to a more specialized setting with M independent classifiers while larger α leads to a more communal setting with a single classifier. For instance, in the absence of any domain-specific knowledge, a good heuristic is to choose $k_m(\mathbf{x},\mathbf{x}) = \omega k(\mathbf{x},\mathbf{x})$ with $\omega = 10M$. Ultimately, the benefits of adaptive pooling will emerge if there is a natural trade-off between specialization and sharing at different rates for each of the M tasks as embodied by the non-uniform estimator of $\hat{\mathbf{s}}$ above.

9. Regression

It is easy to convert multitask feature selection, kernel selection and pooling problems to a regression setup where outputs are scalars $y_{m,t} \in \mathbb{R}$. While this article will only show experiments with classification problems, the multitask regression setting is briefly summarized here for completeness. The main decision in regression problems is what loss function to impose on output predictions. While many loss functions may be considered in regression problems, a popular one is the epsilon-tube loss.

In this type of regression, the goal is to predict the targets within $\pm\varepsilon$. Recall the close similarity between the dual learning problems for SVM classification and SVM regression (Schölkopf and Smola, 2001). The maximum entropy posterior can also be used to reproduce support vector machine regression (Jebara and Jaakkola, 2000; Jebara, 2003). Instead of following the MED derivations in detail, this subsection simply shows the resulting objective function which largely agrees with the standard quadratic program for (single-task) SVM regression with an ε-tube:

$$\max_{\boldsymbol{\lambda},\boldsymbol{\lambda}'} \sum_{t=1}^{T} y_t(\lambda_t - \lambda'_t) - \varepsilon \sum_{t=1}^{T}(\lambda_t + \lambda'_t) - \frac{1}{2}\sum_{t=1}^{T}\sum_{\tau=1}^{T}(\lambda_t - \lambda'_t)(\lambda_\tau - \lambda'_\tau)k(\mathbf{x}_t, \mathbf{x}_\tau)$$
$$\text{s.t. } 0 \leq \lambda_t, \lambda'_t \leq C, \text{ and } \sum_{t=1}^{T}\lambda_t = \sum_{t=1}^{T}\lambda'_t$$

which is solved over Lagrange multipliers $\boldsymbol{\lambda} = \{\lambda_t\}$ and $\boldsymbol{\lambda}' = \{\lambda'_t\}$ for all $t = 1 \ldots T$. SVM regression then applies the following prediction rule:

$$\hat{y} \quad = \quad \sum_{t=1}^{T}(\lambda_t - \lambda'_t)k(\mathbf{x}, \mathbf{x}_t) + \hat{b}.$$

It is straightforward to adapt this regression problem to multitask kernel selection (which once again subsumes feature selection if we select $k_d(\mathbf{x},\mathbf{x}) = \mathbf{x}(d)\mathbf{x}(d)$). MED yields the following multitask objective function which is a convex program:

$$\begin{cases} \max_{\boldsymbol{\lambda},\boldsymbol{\lambda}'} \sum_{m=1}^{M}\sum_{t=1}^{T_m} y_{m,t}(\lambda_{m,t} - \lambda_{m,t'}) - \varepsilon\sum_{m=1}^{M}\sum_{t=1}^{T_m}(\lambda_{m,t} + \lambda_{m,t'}) + D\log(\alpha+1) \\ \quad - \sum_{d=1}^{D}\log\left(\alpha + e^{\frac{1}{2}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}(\lambda_{m,t}-\lambda'_{m,t})(\lambda_{m,\tau}-\lambda'_{m,\tau})k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})}\right) \\ \text{s.t. } 0 \leq \lambda_{m,t}, \lambda'_{m,t} \leq C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m}\lambda_{m,t} - \lambda'_{m,t} = 0 \ \ \forall m. \end{cases}$$

The above is solved by adjusting the Lagrange multipliers $\boldsymbol{\lambda} = \{\lambda_{t,m}\}$ and $\boldsymbol{\lambda}' = \{\lambda'_{t,m}\}$ for all $t = 1 \ldots T_m$ and all $m = 1, \ldots, M$. The resulting prediction rule for a query datum $\mathbf{x}$ for the m'th regression task is then:

$$\hat{y} \quad = \quad \sum_{t=1}^{T_m}(\lambda_{m,t} - \lambda'_{m,t})K(\mathbf{x}, \mathbf{x}_{m,t}) + \hat{b}_m$$

with the kernel $K(\mathbf{x},\mathbf{x}) = \sum_{d=1}^{D}\hat{\mathbf{s}}(d)k_d(\mathbf{x},\mathbf{x})$ as a conic combination of base kernels with weights

$$\hat{\mathbf{s}}(d) \quad = \quad \frac{1}{1 + \alpha\exp(-\frac{1}{2}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}(\lambda_{m,t} - \lambda'_{m,t})(\lambda_{m,\tau} - \lambda'_{m,\tau})k_d(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau}))}.$$

Finally, the biases $\hat{b}_m$ for each task are obtained by solving for the KKT conditions at active Lagrange constraints. Appendix C discusses other choices for the MED loss function in regression settings and connections to previous sparse approaches (Ridge, Lasso and Elastic-Net regression).

10. Sequential Quadratic Programming

In all the optimization problems introduced so far, the optimization appears to be extremely similar to a quadratic program (QP) except for the presence of a handful of log-sigmoid functions. In fact, if the parameter α is set to zero, all the above optimization problems simplify into quadratic programs. It will be shown that the $\alpha > 0$ case can also be easily handled by quadratic programming as well. More precisely, it can be optimized using a sequential quadratic programming (SQP) method. This is a procedure which iteratively solves a QP for a number of iterations. In fact, if the QP is of a simple SVM-type form, much faster SVM solvers can be used instead of QP (Joachims, 2006; Shalev-Shwartz et al., 2007; Bottou and Bousquet, 2008; Shalev-Shwartz and Srebro, 2008). The next section explicates how all MED optimization problems encountered so far can be solved via SQP (or sequential SVM solutions) by bounding the log-sigmoid terms with quadratic functions.

For brevity, we focus on the multitask kernel selection problem which strictly subsumes multitask feature selection. Other learning problems in the previous sections can be implemented with sequential quadratic programming in a similar manner. Recall the kernel selection optimization:

$$\begin{cases} \max_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda}) = \sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma \lambda_{m,t} + D \log(\alpha + 1) \\ \qquad - \sum_{d=1}^{D} \log \left(\alpha + e^{\frac{1}{2} \sum_{m=1}^{M} \sum_{t=1}^{T_m} \sum_{\tau=1}^{T_m} \lambda_{m,t} \lambda_{m,\tau} y_{m,t} y_{m,\tau} k_d(\mathbf{x}_{m,t}, \mathbf{x}_{m,\tau})} \right) \\ \text{s.t. } 0 \leq \lambda_{m,t} \leq C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m} y_{m,t} \lambda_{m,t} = 0 \ \ \forall m. \end{cases}$$

This is a convex problem and generic methods exist for solving it including the ellipsoid method. The latter is a polynomial time algorithm requiring $O((\sum_{m=1}^{M} T_m)^3)$ time yet may still be impractically slow in practice due to large scaling constants (Boyd and Vanderberghe, 2004). Some related optimization methods involving logistic terms have been explored with the Lasso problem (Tibshirani, 1996). Logistic terms often emerge in algorithms that learn sparse (feature-selected) linear classifiers by maximizing the logistic likelihood while enforcing an ℓ_1 regularization on the linear model parameters. This is the approach followed by the ℓ_1 regularized sparse logistic regression technique (Koh et al., 2007). Interestingly, this recent work has developed fast interior-point optimization methods which may be eventually applicable to MED problems. Instead, we solve the MED problem by exploiting a convenient upper bound on logistic-quadratic functions that converts them into plain quadratic functions. In previous work, a looser version of the bound was proposed (Jebara and Jaakkola, 2000). This article refines the bound and provides a tight variational quadratic upper bound on a logistic function *of* a quadratic function. This conversion to quadratic functions permits us to use standard quadratic programming. In fact, the actual optimizations ultimately decouple into the solution of M separate support vector machines and prevent cubic growth in the number of tasks. Bounding is interleaved with the solution of support vector machines to iteratively maximize $J(\boldsymbol{\lambda})$. Due to the availability of fast SVM solvers, this optimization approach is potentially more promising than more generic convex programming tools (Koh et al., 2007). The necessary bound is derived in detail in Theorem 1 in the Appendix. The theorem states that $\log\left(\alpha + \exp\left(\frac{\mathbf{u}^\top \mathbf{u}}{2}\right)\right)$ is less than or equal to a convex quadratic function in $\mathbf{u}$ for all vectors $\mathbf{u}$ and achieves strict equality when $\mathbf{u} = \mathbf{v}$ for some vector $\mathbf{v}$. We will apply the above bound to each log-sigmoid term in the sum over $d = 1 \ldots D$ in $J(\boldsymbol{\lambda})$. We slightly abuse notation and interchangeably use $\boldsymbol{\lambda}_m$ to denote the vector of Lagrange multipliers $(\lambda_{m,1}, \ldots, \lambda_{m,T_m})^\top$ for each $m = 1 \ldots M$. Similarly, we will take $\boldsymbol{\lambda} \in \mathbb{R}^\Gamma$ where $\Gamma = \sum_{m=1}^{M} T_m$ to be a concatenation of all Lagrange multipliers. Consider the d'th log-sigmoid function in the sum $\sum_{d=1}^{D} \log(\ldots)$ in $J(\boldsymbol{\lambda})$. Denote the Hessian of the

quadratic term inside the d'th log-sigmoid as $H_d \in \mathbb{R}^{\Gamma\times\Gamma}$ which is given element-wise as follows:

$$H_d([m,t],[n,\tau]) \quad = \quad y_{m,t}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\delta_{m=n}.$$

Here we use $\delta_{m=n}$ as an indicator function that is 1 if $m=n$ and is zero otherwise. We also use the operator $[m,t]$ to compute the index value $[m,t]=(t+\sum_{n=1}^{m-1}T_n)$ to select the appropriate row and column entries of the matrix H_d. This allows us to write the dual objective function as

$$\begin{cases} \max_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda}) = D\log(\alpha+1) - \sum_{d=1}^{D}\log\left(\alpha+\exp\left(\frac{1}{2}\boldsymbol{\lambda}^\top H_d\boldsymbol{\lambda}\right)\right) + \gamma\boldsymbol{\lambda}^\top\mathbf{1} \\ \text{s.t. } 0\leq\lambda_{m,t}\leq C \ \ \forall m,t \text{ and } \sum_{t=1}^{T_m}y_{m,t}\lambda_{m,t}=0 \ \ \forall m. \end{cases}$$

Assume we have a current setting of the Lagrange multipliers $\tilde{\boldsymbol{\lambda}}$. We apply Theorem 1 in the Appendix after a simple change of variables, $\mathbf{u}=H_d^{1/2}\boldsymbol{\lambda}$ and $\mathbf{v}=H_d^{1/2}\tilde{\boldsymbol{\lambda}}$ which gives:

$$\begin{aligned}\log\left(\alpha+\exp\left(\frac{\boldsymbol{\lambda}^\top H_d\boldsymbol{\lambda}}{2}\right)\right) \quad \leq \quad & \log\left(\alpha+\exp\left(\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2}\right)\right) + \frac{\exp(\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})}{\alpha+\exp(\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})}\tilde{\boldsymbol{\lambda}}^\top H_d(\boldsymbol{\lambda}-\tilde{\boldsymbol{\lambda}}) \\ & +\frac{1}{2}(\boldsymbol{\lambda}-\tilde{\boldsymbol{\lambda}})^\top\left(\mathcal{G}_d H_d\tilde{\boldsymbol{\lambda}}\tilde{\boldsymbol{\lambda}}^\top H_d + H_d\right)(\boldsymbol{\lambda}-\tilde{\boldsymbol{\lambda}}).\end{aligned}$$

Such a bound is applied to each log-sigmoid term in $J(\boldsymbol{\lambda})$ individually for $d=1\ldots D$. The ratio terms in the bound are none other than the expected switch variables at the current setting of $\tilde{\boldsymbol{\lambda}}$:

$$\hat{\mathbf{s}}(d) \quad = \quad \frac{\exp(\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})}{\alpha+\exp(\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})} = \frac{1}{\alpha\exp(-\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})+1}.$$

Similarly, we obtain the following for $\mathcal{G}_d$ applying[5] the bound formula:

$$\mathcal{G}_d \quad = \quad \frac{\tanh(\frac{1}{2}\log(\alpha\exp(-\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})))}{2\log(\alpha\exp(-\frac{\tilde{\boldsymbol{\lambda}}^\top H_d\tilde{\boldsymbol{\lambda}}}{2})}.$$

Other convenient variables to define are the vectors $\hat{\mathbf{y}}_{m,t}\in\mathbb{R}^D$ for $m=1,\ldots,M$ and $t=1,\ldots,T_m$. These are the predicted label of the m'th SVM on the t'th datum using the d'th kernel at the current setting of $\tilde{\boldsymbol{\lambda}}$. They are given element-wise as follows:

$$\hat{\mathbf{y}}_{m,t}(d) \quad = \quad \sum_{\tau=1}^{T_m}\tilde{\boldsymbol{\lambda}}_{m,\tau}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau}).$$

Applying these substitutions and the bound on each log-sigmoid function produces the following variational lower bound on the objective function:

$$\begin{aligned} J(\boldsymbol{\lambda}) \quad \geq \quad & \text{constant} + \sum_{m=1}^{M}\sum_{t=1}^{T_m}\gamma\lambda_{m,t} - \sum_{m=1}^{M}\sum_{t=1}^{T_m}\lambda_{m,t}y_{m,t}\sum_{d=1}^{D}\hat{\mathbf{s}}(d)\hat{\mathbf{y}}_{m,t}(d) \\ & + \sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\tilde{\lambda}_{m,\tau}y_{m,t}y_{m,\tau}\sum_{d=1}^{D}\left(\mathcal{G}_d\hat{\mathbf{y}}_{m,t}(d)\hat{\mathbf{y}}_{m,\tau}(d)+k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\right). \\ & -\frac{1}{2}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}\sum_{d=1}^{D}\left(\mathcal{G}_d\hat{\mathbf{y}}_{m,t}(d)\hat{\mathbf{y}}_{m,\tau}(d)+k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\right). \end{aligned}$$

5. By continuity, take $\tanh(\frac{1}{2}\log(1))/(2\log(1))=1/4$ and also take $\lim_{z\to 0^+}\tanh(\frac{1}{2}\log(z))/(2\log(z))=0$.

Interestingly, given the current $\tilde{\boldsymbol{\lambda}}$ and the current $\hat{\mathbf{s}}(1),\ldots,\hat{\mathbf{s}}(D)$, the bound effectively *decouples* the learning problem across the M tasks. The objective function becomes quadratic and additive across $m = 1 \ldots M$. Therefore, we can solve each problem individually as a single support vector machine. This provides a simple iterative algorithm for multitask learning which builds on current SVM solvers. The steps[6] are summarized in Algorithm 1.

Algorithm 1 simply performs sequential quadratic programming by interleaving the bound computation with SVM programs. The SVMs are solved separately for $m = 1,\ldots,M$ tasks in Step 3b. If each SVM is solved using standard quadratic programming solvers, each requires $\mathrm{O}(T_m^3)$. However, by exploiting more recent *approximate* SVM solvers, the inner loop SVM problems can potentially complete in linear time or $\mathrm{O}(T_m)$ (Joachims, 2006; Shalev-Shwartz et al., 2007; Bottou and Bousquet, 2008; Shalev-Shwartz and Srebro, 2008). Admittedly, this is true only subject to certain reasonable assumptions (for instance, small approximation errors are allowed and explicit linear feature mappings are used rather then implicit nonlinear kernels). Therefore, under certain assumptions, step 3b in Algorithm 1 can potentially complete in $\mathrm{O}(\sum_{m=1}^{M} T_m)$ time.[7] Finally, it is also possible to use warm-starting and seed the SVM solver with a previous $\boldsymbol{\lambda}$ result to obtain further speedup. For instance, warm starting can be used from a previous iteration in Algorithm 1. Furthermore, we may warm start from a previous final solution of Algorithm 1 that converged for a smaller setting of C or α. This lets us explore the regularization path efficiently after initializing it at, for instance, the default setting of $\alpha = 0$ and $C = 1$ and increasing both parameters until error is minimized on a cross-validation set. Furthermore, we typically set $\gamma = 1$ to mimic the support vector machine case but that parameter may be adjusted as well (either manually or by cross-validation).

Algorithm 1 Multitask SVM Learning

0	Input data set $\mathcal{D}, C > 0, \alpha \geq 0, 0 < \varpi < 1$ and kernels k_d for $d = 1,\ldots,D$.
1	Initialize Lagrange multipliers to zero $\boldsymbol{\lambda} = \mathbf{0}$.
2	Store $\tilde{\boldsymbol{\lambda}} = \boldsymbol{\lambda}$.
3	For $m = 1,\ldots,M$ do:
3a	Set $g_d = \alpha \exp\left(-\frac{1}{2}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})\right)$ for all d. Set $\mathcal{G}_d = \frac{\tanh(\frac{1}{2}\log(g_d))}{2\log(g_d)}$ for all d. Set $\hat{\mathbf{s}}(d) = \frac{1}{1+g_d}$ for all d. Set $\hat{\mathbf{y}}_{m,t}(d) = \sum_{\tau=1}^{T_m}\lambda_{m,\tau}y_{m,\tau}k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau})$ for all t and d.
3b	Update each of the $\boldsymbol{\lambda}_m$ vectors with the SVM QP: $\max_{\boldsymbol{\lambda}_m} \sum_{t=1}^{T_m}\lambda_{m,t} - \sum_{t=1}^{T_m}\lambda_{m,t}y_{m,t}\sum_{d=1}^{D}\hat{\mathbf{s}}(d)\hat{\mathbf{y}}_{m,t}(d)$ $+\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\tilde{\lambda}_{m,\tau}y_{m,t}y_{m,\tau}\sum_{d=1}^{D}(\mathcal{G}_d\hat{\mathbf{y}}_{m,t}(d)\hat{\mathbf{y}}_{m,\tau}(d)+k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau}))$ $-\frac{1}{2}\sum_{t=1}^{T_m}\sum_{\tau=1}^{T_m}\lambda_{m,t}\lambda_{m,\tau}y_{m,t}y_{m,\tau}\sum_{d=1}^{D}(\mathcal{G}_d\hat{\mathbf{y}}_{m,t}(d)\hat{\mathbf{y}}_{m,\tau}(d)+k_d(\mathbf{x}_{m,t},\mathbf{x}_{m,\tau}))$ s.t. $0 \leq \lambda_{m,t} \leq C \ \ \forall t = 1,\ldots,T_m$ and $\sum_{t=1}^{T_m}y_{m,t}\lambda_{m,t} = 0$.
4	If $\|\boldsymbol{\lambda} - \tilde{\boldsymbol{\lambda}}\| > \varpi\|\boldsymbol{\lambda}\|$ go to 2.
5	Output: $\hat{\mathbf{s}}$ and $\boldsymbol{\lambda}$.

Next, we discuss the convergence of the above iterative algorithm. Clearly, since the algorithm maximizes a variational lower bound on the objective function, it must monotonically increase the objective. However, it is still possible that the algorithm can get stuck and produce negligible

6. Code available at `www.cs.columbia.edu/~jebara/code/multisparse/`.

7. Also, under mild *iid* assumptions, step 3a can be well approximated in $\mathrm{O}(\sum_{m=1}^{M} T_m)$ time using deviation bounds.

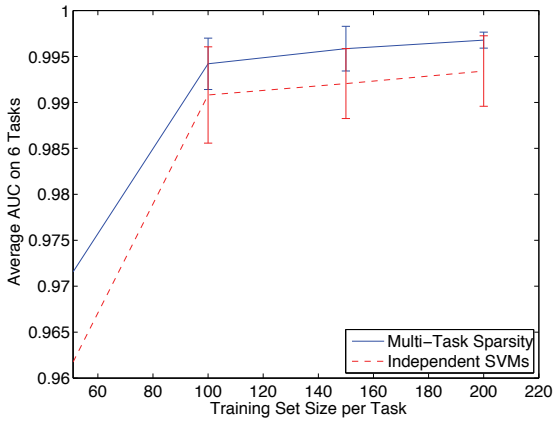


Figure 1: Feature selection on the UCI Dermatology data set. Multitask sparse feature selection and independent SVM classification are compared. Various data set sizes are shown ranging from 20 to 200 samples for each of the 6 tasks. The average area under the ROC curve on test data is shown for all tasks for 5 folds (along with the standard deviation). The values of C and α were obtained by cross-validation on held out data.

progress requiring an unbounded number of iterations. We will show that is not the case and, indeed, the sequential quadratic programming procedure in Algorithm 1 will only require a finite number of iterations (of step 3). The number of iterations is bounded by Theorem 2 which is proved in the Appendix. It guarantees that, for any $\alpha \geq 0, \varepsilon \in (0,1)$, Algorithm 1 finds a $\tilde{\boldsymbol{\lambda}}$ that satisfies $J(\tilde{\boldsymbol{\lambda}}) \geq (1-\varepsilon)J(\boldsymbol{\lambda}^*)$ (where $\boldsymbol{\lambda}^*$ is the constrained maximizer of $J(\boldsymbol{\lambda})$) in no more than

$$\left\lceil \frac{\log(1/\varepsilon)}{\log\left(\min\left(1+\frac{1}{\alpha}, 2\right)\right)} \right\rceil$$

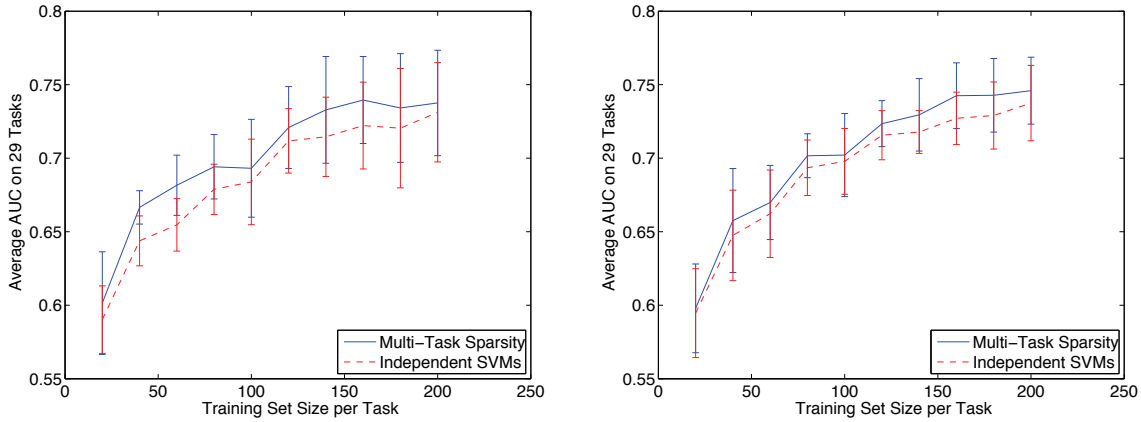
iterations. Here, each iteration involves (possibly warm-started) SVM programs and the expression $\lceil \ldots \rceil$ denotes the integer ceiling function.

Therefore, a constant number of iterations is needed that depends only on α. In summary, solving multitask feature or kernel selection is only a constant factor more computational effort than solving M independent support vector machines. A similar SQP or iterative SVM algorithm can be derived for the adaptive pooling setup described in Section 8.

11. Experiments

To evaluate the multitask learning framework, we considered UCI data[8] as well as the Land Mine data set[9] which was developed and investigated in previous work (Xue et al., 2007). The classification accuracy of standard support vector machines learned independently is compared to the accuracy of the multitask kernel selection procedure described in Section 6 and Section 7. In all experiments, we explore multiple values of the regularizer C for the SVM and multiple values of

8. Data available at `http://archive.ics.uci.edu/ml/`.
9. Data available at `http://www.cs.columbia.edu/~jebara/code/multisparse/LandmineData.mat`.



(a) Feature and RBF kernel selection

(b) Feature, polynomial and RBF kernel selection

Figure 2: Feature selection and kernel selection on the Landmine data set. In (a), feature selection is combined with RBF kernel selection. In (b), feature selection is combined with both polynomial and RBF kernel selection. Multitask sparse kernel selection and independent SVM classification are compared. Various data set sizes are shown ranging from 20 to 200 samples for each of the 29 tasks. The average area under the ROC curve on test data is shown for all tasks for 5 folds (along with the standard deviation). The values of C and α were obtained by cross-validation on held out data.

C and α (or, equivalently, ρ) for the multitask learner. The values of C and α are determined by cross-validation on held out data and then tested on an unseen test set.

The UCI dermatology data set consists of 6 classes which can be converted into binary classification tasks to be predicted from an input space of 34-dimensional features. A total of 366 instances are available. Both the independent SVMs and the multitask feature selection approach were evaluated by training on various numbers of examples (from 20 to 200) for each task, and the remaining examples (with labels kept unobserved) are split in half for cross-validation and testing. The feature selection method chooses a sparse subset of the 34 features that are consistently good at predicting the label for the 6 different tasks (or classes). All evaluations were done using the average area under the Receiver Operating Characteristic (ROC) curve for the 6 tasks. This score is the MAUC since it involves the mean of M tasks' Area Under the Curve (AUC) scores. Cross-validation was used to select a value of C for the independent SVMs and values of α and C for the multitask feature selection SVMs. Figure 1 shows the MAUC performance of the independent SVMs versus the multitask SVMs with averages and standard deviations across 5 folds. There is a clear and statistically significant advantage (under a paired t-test) for multitask learning over independent SVM classification.

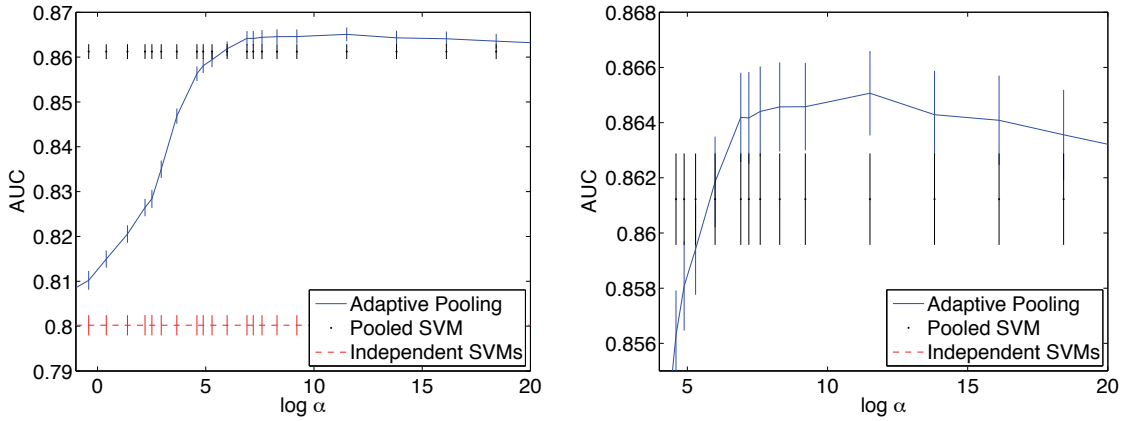
The Landmine data set consists of 29 binary classification tasks involving an input space of $D = 9$ dimensional features. The number of samples for each task varies from 445 to 690. Both independent SVM learning and the multitask kernel selection approach were evaluated by training on various numbers of examples $(20, 40, \ldots, 200)$ from each task. The remaining examples were split in half for cross-validation and for testing. We perform feature selection by building a kernel for each feature that is simply the product of a single scalar dimension for a pair of data points. This

produces 9 kernels. In addition, 9 radial basis function (RBF) kernels were computed with different settings of the bandwidth parameter. The kernel selection method was then used to choose a sparse subset of these $D = 18$ total kernels. All evaluations were done using the average area under the ROC curve for the 29 tasks. Cross-validation was used to select a value of C for the independent SVM approach and to select values for C and α for the multitask kernel selection SVM. Figure 2(a) shows the performance of the independent SVMs versus the multitask SVMs as an average and standard deviation of MAUC across 5 folds. Tabula rasa learning obtains lower accuracy in general while multitask learning improved accuracy at all sizes of the training data set with statistical significance (a paired t-test produced a p-value below 0.05) on most training set sizes.

Another experiment exploring kernel selection was considered using all the previous kernels as well as linear, quadratic, cubic and quartic kernels for a total of $D = 22$ kernels. Figure 2(b) summarizes the results which again demonstrate an advantage for the multitask setup. These results compare favorably with previous experiments on this data set (Xue et al., 2007).

In all experiments, solving the more elaborate objective function in the MED convex program required only a constant factor more time than solving each task separately with independent SVMs. We verified that the number of iterations of the SVMs only increased as a function of α and required 2 to 40 iterations of Step 3 in Algorithm 1 as α was swept across the range of interest. Since the SVMs were warm-started at their previous solutions, sweeping across a range of α values in the multitask sparsity approach (after starting from an initial SVM solution) never required more than 50 times the run time of the initial SVM solution. Thus, empirically, the multitask sparsity framework, while sweeping over the full regularization path over α, incurs a constant factor (under 50) increase in the computational effort over independent SVM learning. These runtime results agree with Theorem 2.

In another experiment with adaptive pooling, the Heart data set from the UCI repository was used. All features were normalized to within the $[0,1]$ box and a polynomial kernel of degree three was used throughout. The Heart data set was changed into a multitask data set by dividing the data into ten different tasks based on the age of the patient. This division was done by splitting the data along the age variable by forming 10 intervals with equal number of examples in each interval. For each task, the examples were divided into train/test/validation sets with equal number of examples in each. A scaling factor of $\frac{1}{10}$ was applied to the communal kernel and a scaling factor of $\frac{9}{10}M$ was applied to the specialized (task-specific) kernels. Independent learning and full pooling results were obtained by finding the SVM solutions on each data set in isolation and then by finding an SVM on the pooled data from all tasks. The parameter C was chosen based on performance on a validation set. For adaptive pooling, α values were also explored from 0 to e^9. The C value which resulted in the highest AUC on the validation data was used to pick the AUC for each α value. The experiment was repeated 100 times to get the test AUC over different random splits of the data. An advantage for adaptive pooling was evident when $\alpha = 1e5$ and was statistically significant at better than the 5% p-value threshold (using a paired t-test). Figure 3 shows the average test AUC results across 100 folds using independent SVMs, pooling and adaptive pooling for various values of α (after cross-validation only over the value of C for all methods). The Figure reveals an advantage for adaptive pooling compared to full pooling and independent learning.



(a) Average AUC (b) Average AUC zoomed in

Figure 3: Adaptive pooling experiments on the Heart data set. The area under the curve (AUC) for adaptive pooling, pooling and independent SVMs is shown in (a). In (b), a zoomed in version of the plot is shown to focus on the setting with highest average AUC. The value of the regularization parameter C was found using cross-validation for all three methods.

12. Graphical Model Structure Estimation

The multitask sparse discrimination framework is a general tool for large margin classification since most elements of $\hat{\mathbf{s}}$ become vanishingly small (at appropriate settings of ρ or α). This motivates extending the framework to other sparse inference problems including the estimation of graphical model structure which has been explored as an ℓ_1 sparse regression with asymptotic guarantees (Wainwright et al., 2007). The ℓ_1 approach infers a graphical model by learning functions that reconstruct some dimensions given others under sparsity constraints. Assume that we are given T binary vectors $\mathbf{x}_1, \ldots, \mathbf{x}_T$ where $\mathbf{x}_t \in \mathbb{B}^D$ are sampled *iid* from an unknown distribution

$$p(\mathbf{x}) \quad \propto \quad \exp\left(\sum_{m=1}^{D} \eta(m)\mathbf{x}(m) + \sum_{m=1}^{D}\sum_{n=1}^{D} E(m,n)\theta(m,n)\mathbf{x}(m)\mathbf{x}(n) \right).$$

This Ising model is specified by an undirected graph $G = (V, E)$ with D vertices V and edges E, where, without loss of generality, we may assume that $E \in \mathbb{B}^{D\times D}$ is also a binary symmetric adjacency matrix with zero on its diagonal, $\theta \in \mathbb{R}^{D\times D}$ is a symmetric real matrix with zero on its diagonal and $\eta \in \mathbb{R}^D$ is a real vector. The goal of graphical model structure estimation is to recover an estimate $\hat{E}_T$ of the binary matrix E solely from the observations $\mathbf{x}_1, \ldots, \mathbf{x}_T$.

In previous work (Wainwright et al., 2007), a method was provided that achieves $\Pr[\hat{E}_T = E] \to 1$ as $T \to \infty$ by solving independent sparse regression problems as follows,

$$\hat{\theta}_m = \arg\min_{\theta\in\mathbb{R}^D} \nu \sum_{d\neq m} |\theta(d)| + \sum_{t=1}^{T} \log(1 + e^{\sum_{d\neq m}\theta(d)\mathbf{x}_t(d)+\theta(m)}) - \mathbf{x}_t(m)\Big(\sum_{d\neq m}\theta(d)\mathbf{x}_t(d) + \theta(m)\Big)$$

for $m = 1, \ldots, D$. These tasks reconstruct each dimension from all other dimensions. In other words, the m'task is given $\{\mathbf{x}(1), \ldots, \mathbf{x}(D)\} \setminus \mathbf{x}(m)$ and predicts $\mathbf{x}(m)$. The ℓ_1 sparsity constraint, for

appropriate settings of the parameter ν, makes the problem non trivial since only some inputs can be used in the reconstruction. To recover a single consistent set of edges $\hat{E}_T$, the nonzero components of $\hat{\theta}_m$ estimated for various tasks are combined using either an AND or an OR rule. In the AND case, $\hat{E}(m,n)$ is set to 1 if both $\hat{\theta}_m(n)$ is nonzero and $\hat{\theta}_n(m)$ is nonzero. In the OR case, $\hat{E}(m,n) = 1$ if either of the terms is nonzero.

The multitask MED approach can potentially circumvent this ad hoc AND/OR step by forcing all sparse predictors to agree on a single undirected edge connectivity matrix E from the outset. The MED approach considers $m = 1,\ldots,D$ tasks where the m'th task is given $\mathbf{x}_t$ and must predict $y_{m,t} = 2\mathbf{x}_t(m) - 1 \in \pm 1$ as a classification output. We assume the following predictive distribution:

$$p(y|m,\mathbf{x},\theta,b,\mathbf{s}) \quad \propto \quad \exp\left(\frac{y}{2}\sum_{d\neq m}\mathbf{s}(m,d)\mathbf{x}(d)\theta(m,d) + \mathbf{b}(m)\right).$$

The MED model Θ contains a matrix $\theta \in \mathbb{R}^{D\times D}$ with its diagonal forced to zero. In addition, it contains a binary matrix $\mathbf{s} \in \mathbb{B}^{D\times D}$ (again with its diagonal forced to zero) and finally a scalar vector $\mathbf{b} \in \mathbb{R}^D$. The standard Gaussian priors are applied to the model parameters in $P(\Theta)$ except for the $\mathbf{s}$ variable which obtains a Bernoulli prior over its binary entries and (for sufficiently large α) to encourage its sparsity. In addition, we a priori enforce the symmetry $\mathbf{s}(m,d) = \mathbf{s}(d,m)$. This ensures that, if input $\mathbf{x}(d)$ is used for the prediction of $\mathbf{x}(m)$, $\mathbf{x}(m)$ can also be used for predicting $\mathbf{x}(d)$. However, symmetry is not enforced on the θ parameters which permits us to learn different linear relationships once a consistent dependency structure is determined. Thus, consistency of the edges used by the sparse prediction is enforced up-front in a multitask setting instead of resorting to a post-processing (i.e., the AND or OR steps) as in the previous approach which independently learns D regression functions.

The MED framework computes the partition function by integrating the following:

$$\begin{aligned} Z(\boldsymbol{\lambda}) &= \int p(\Theta)\exp\left(\sum_{m=1}^{D}\sum_{t=1}^{T}\lambda_{m,t}y_{m,t}\left(\sum_{d\neq m}\mathbf{s}(m,d)\mathbf{x}_t(d)\theta(m,d) + \mathbf{b}(m)\right) - \lambda_{m,t}\right) d\Theta \\ &= e^{\sum_m \frac{\sigma^2}{2}\left(\sum_t \lambda_{m,t}y_{m,t}\right)^2 - \sum_t \lambda_{m,t}} \sum_{\mathbf{s}} p(\mathbf{s}) e^{\frac{1}{2}\sum_{m=1}^{D}\sum_{d=m+1}^{D}\mathbf{s}(m,d)\left(\left(\sum_t \lambda_{m,t}y_{m,t}\mathbf{x}_t(d)\right)^2 + \left(\sum_t \lambda_{d,t}y_{d,t}\mathbf{x}_t(m)\right)^2\right)}. \end{aligned}$$

Taking $\sigma \to \infty$ and $J(\boldsymbol{\lambda}) = -\log(Z(\boldsymbol{\lambda}))$ produces (up to an additive constant) the dual program:

$$\begin{cases} \max_{\boldsymbol{\lambda}} \sum_{m=1}^{D}\sum_{t=1}^{T}\lambda_{m,t} - \sum_{m=1}^{D}\sum_{d=m+1}^{D}\log\left(\alpha + e^{\frac{1}{2}\left(\left(\sum_t \lambda_{m,t}y_{m,t}\mathbf{x}_t(d)\right)^2 + \left(\sum_t \lambda_{d,t}y_{d,t}\mathbf{x}_t(m)\right)^2\right)}\right) \\ \text{s.t. } 0 \leq \lambda_{m,t} \leq C \ \ \forall m,t \text{ and } \sum_{t=1}^{T} y_{m,t}\lambda_{m,t} = 0 \ \ \forall m. \end{cases}$$

The objective function can be written as

$$\max_{\boldsymbol{\lambda}\in\Lambda} \boldsymbol{\lambda}^\top \mathbf{1} - \sum_{m=1}^{D}\sum_{d=m+1}^{D}\log\left(\alpha + e^{\frac{1}{2}\boldsymbol{\lambda}^\top H_{m,d}\boldsymbol{\lambda}}\right)$$

where the $H_{m,d} \in \mathbb{R}^{DT\times DT}$ matrices for $d > m \in \{1,\ldots,D\}$ are defined element-wise as

$$H_{m,d}([n,t],[o,\tau]) \quad = \quad y_{m,t}y_{m,\tau}\mathbf{x}_t(d)\mathbf{x}_\tau(d)\delta_{m=n=o} + y_{d,t}y_{d,\tau}\mathbf{x}_t(m)\mathbf{x}_\tau(m)\delta_{d=n=o}.$$

It is easy to maximize the objective using sequential quadratic programming which gives an estimate for $\boldsymbol{\lambda}$. The prediction rule is then $\hat{y} = \arg\max_y \mathrm{E}_{p(\Theta|\mathcal{D})}[\log p(y|m,\mathbf{x},\Theta,\mathbf{b},\mathbf{s})]$ which involves the sparse variable $\mathbf{s}$. These switch configurations essentially identify the network structure and are obtained from expected $\mathbf{s}(m,d)$ values under the posterior $p(\Theta|\mathcal{D})$ as follows:

$$\hat{\mathbf{s}}(m,d) \quad = \quad \frac{1}{1+\alpha \exp\left(-\frac{1}{2}\left(\left(\sum_t \lambda_{m,t} y_{m,t} \mathbf{x}_t(d)\right)^2 + \left(\sum_t \lambda_{d,t} y_{d,t} \mathbf{x}_t(m)\right)^2\right)\right)}.$$

For large α, many entries of $\hat{\mathbf{s}}$ are driven towards small values as MED resembles an ℓ_1 regularizer. MED produces sparsity although only in a probabilistic sense since coefficients do not strictly go to zero but typically shrink to small values. The matrix $\hat{\mathbf{s}}$ represents MED's estimate of the unknown adjacency matrix E in the original graphical model.

To test the accuracy of the method, the scalar values of $\hat{\mathbf{s}}$ are used as scalar classification predictions for the presence or absence of an edge. Given the true graph, these predictions are straightforward to evaluate using the AUC. Experimental results with synthetic data are obtained by generating random graphs and obtaining samples from them according to the Ising model above (Wainwright et al., 2007). In Figure 4, the mean area under the curve (MAUC) is reported for the MED technique as well as the independent ℓ_1 regularized regressions with an AND and an OR step. Multiple settings of the regularization parameters are shown in the plot as the value of the regularization ν is explored in the original method (for both the AND and OR setting) and the values of C and α are explored in the proposed method. Since the ℓ_1 regularization method (Wainwright et al., 2007) is asymptotically correct, the experiments here focus on the small sample regime. From ten random graphical models over 5 nodes, 60 samples were drawn using Monte Carlo methods and the average area under the curve for the various methods was reported. To fairly compare results using an AUC measure, we did not only use the support found by the ℓ_1 regularized method but also considered all possible thresholds on the ℓ_1 solution. More specifically, the min or the max operators were first used to symmetrize the absolute value of the regression weights recovered by ℓ_1 regularization. These non-negative scalars were then used in the graphical model to allow all operating points on the receiver operator characteristic to be explored. This can only improve the performance of the ℓ_1 regularization method in terms of AUC (a binary estimate of edges followed by an AND or an OR step can only obtain lower AUC). Despite this, the proposed method [10] performs significantly better possibly due to the explicit symmetry in the edge estimation. These preliminary experiments motivate large scale future empirical work.

13. Discussion

A multitask learning framework was developed for support vector machines and large-margin linear classifiers. Each task-specific classifier is estimated to solve its own problem yet all tasks have to share a common representation $\mathbf{s}$. This common representation included sparse feature selection and conic kernel combination. This common representation couples tasks to go beyond standard tabula rasa learning. To compute the coupled linear models, we applied the maximum entropy discrimination framework which produces support vector machines that share a common sparse representation. The framework combines classification problems non-trivially in a convex dual-space optimization. We presented a simple sequential quadratic programming approach for solving the

10. Code available at `www.cs.columbia.edu/~jebara/code/multisparse/`.

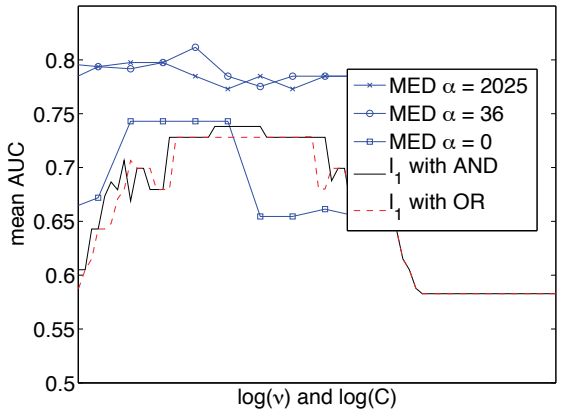


Figure 4: Graphical model structure estimation from data sampled from Ising models. The average area under the curve is shown for ten random models. The proposed method is evaluated across various values of C and α and compared to ℓ_1 regularized logistic regression method across various values of ν with both AND and OR symmetrization.

dual optimization for both multitask feature selection and multitask kernel selection problems. We interleave bound computations with standard SVM updates (either using quadratic programming or, preferably, nearly linear-time modern SVM solvers). In addition, the extensions to adaptive pooling, sparse regression and graphical model reconstruction were illustrated. The MED multitask framework potentially allows flexible exploration of sparsity structure over different groups of variables and is reminiscent of Group Lasso methods (Yuan and Lin, 2006; Bach, 2008). Experiments on real world data sets show that MED multitask learning is advantageous over single-task or tabula rasa learning.

In future work, it would be interesting to investigate theoretical generalization guarantees for multitask sparse MED. This may involve exploiting PAC-Bayesian model selection methods or online mistake bound methods (McAllester, 1999; Langford and Shawe-Taylor, 2002; Long and Wu, 2004) which have already given generalization arguments for the single-task MED approach. Since generalization guarantees in multitask settings have already been provided for other algorithms (Ando and Zhang, 2005; Maurer, 2006, 2009), this may be a fruitful line of work. Finally, it may be useful to explore methods for automatically estimating the hyper-parameters in the MED framework such as α which, as in classical Bayesian approaches, might be handled via optimization or integration rather than cumbersome cross-validation.

Acknowledgments

The author is grateful to the anonymous referees, P. Shivaswamy, T. Jaakkola, M. Meila and D. Rosenberg for valuable comments. The author also acknowledges P. Shivaswamy for help with experiments involving adaptive pooling and P. Ravikumar and M. Wainwright for sharing code to perform ℓ_1-regularized graphical model structure estimation. This work was funded in part

by the National Science Foundation (Grant IIS-0347499), the Office of Naval Research (Grant N000140710507) and the Department of Homeland Security (Grant N66001-09-C-0080).

Appendix A. Bounding the Logistic-Quadratic Function

Theorem 1 *For all* $\mathbf{u} \in \mathbb{R}^D$, $\log\left(\alpha + \exp\left(\frac{\mathbf{u}^\top \mathbf{u}}{2}\right)\right)$ *is bounded above by*

$$\log\left(\alpha + \exp\left(\frac{\mathbf{v}^\top \mathbf{v}}{2}\right)\right) + \frac{\mathbf{v}^\top(\mathbf{u}-\mathbf{v})}{1+\alpha\exp(-\frac{\mathbf{v}^\top \mathbf{v}}{2})} + \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top\left(I + \mathcal{G}\mathbf{v}\mathbf{v}^\top\right)(\mathbf{u}-\mathbf{v})$$

for the scalar term $\mathcal{G} = \frac{1}{2}\tanh(\frac{1}{2}\log(\alpha\exp(-\mathbf{v}^\top\mathbf{v}/2)))/\log(\alpha\exp(-\mathbf{v}^\top\mathbf{v}/2))$. *The bound holds for any* $\alpha \geq 0, \mathbf{v} \in \mathbb{R}^D$ *and strict equality is achieved when* $\mathbf{u} = \mathbf{v}$.

Proof The proof proceeds by first making the bound achieve (tangential) equality at $\mathbf{u} = \mathbf{v}$. It then applies a previously known bound on the logistic function using convexity arguments. The logistic-quadratic function $g(\mathbf{u})$ and the general quadratic function $q(\mathbf{u})$ are defined as

$$\begin{aligned} g(\mathbf{u}) &= \log\left(\alpha + \exp\left(\frac{\mathbf{u}^\top \mathbf{u}}{2}\right)\right), \\ q(\mathbf{u}) &= c + \mathbf{b}^\top(\mathbf{u}-\mathbf{v}) + \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v}). \end{aligned}$$

The quadratic function $q(\mathbf{u})$ is parametrized by a scalar $c \geq 0$, a vector $\mathbf{b} \in \mathbb{R}^D$ and a positive semi-definite matrix $A \in \mathbb{R}^{D\times D}$. These parameters must be selected to ensure $q(\mathbf{u}) \geq g(\mathbf{u})$ for all $\mathbf{u} \in \mathbb{R}^D$. Furthermore, the theorem requires that $g(\mathbf{v}) = q(\mathbf{v})$. This determines the additive constant $c = \log\left(\alpha + \exp\left(\frac{\mathbf{v}^\top \mathbf{v}}{2}\right)\right)$. Since equality is achieved at $\mathbf{u} = \mathbf{v}$, the gradients must be equal there as well, in other words $\left.\frac{\partial g(\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{u}=\mathbf{v}} = \left.\frac{\partial q(\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{u}=\mathbf{v}}$. This determines that $\mathbf{b} = \exp\left(\frac{\mathbf{v}^\top \mathbf{v}}{2}\right)/(\alpha + \exp\left(\frac{\mathbf{v}^\top \mathbf{v}}{2}\right))\mathbf{v}$. Otherwise, the functions cross at $\mathbf{u} = \mathbf{v}$ which violates the bound. Inserting these values for $\mathbf{b}$ and c into the quadratic form for $q(\mathbf{u})$ reveals that A must be chosen such that $\frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v})$ is greater than or equal to

$$\log\left(\frac{\alpha + \exp(\mathbf{u}^\top\mathbf{u}/2)}{\alpha + \exp(\mathbf{v}^\top\mathbf{v}/2)}\right) - \frac{\exp(\mathbf{v}^\top\mathbf{v}/2)}{\alpha + \exp(\mathbf{v}^\top\mathbf{v}/2)}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}).$$

Consider the choice for A suggested by the theorem to prove that it satisfies this requirement

$$A = I + \frac{\tanh(\frac{1}{2}\log(\alpha\exp(-\mathbf{v}^\top\mathbf{v}/2)))}{2\log(\alpha\exp(-\mathbf{v}^\top\mathbf{v}/2))}\mathbf{v}\mathbf{v}^\top.$$

Multiply A appropriately to obtain the desired expression

$$\frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v}) = \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top\left(I + \frac{\tanh(\frac{1}{2}\log(\varphi))}{2\log(\varphi)}\mathbf{v}\mathbf{v}^\top\right)(\mathbf{u}-\mathbf{v}),$$

where, for brevity, we define the scalar $\varphi = \alpha\exp(-\mathbf{v}^\top\mathbf{v}/2)$. Rewrite the right hand side as

$$\begin{aligned} \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v}) &= \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top\left(I + \frac{\tanh(\frac{1}{2}\log(\varphi))}{2\log(\varphi)}\mathbf{v}\mathbf{v}^\top\right)(\mathbf{u}-\mathbf{v}) - \frac{\tanh(\frac{1}{2}\log\varphi)}{2}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}) \\ &\quad + \frac{\varphi-1}{2}\frac{1}{\varphi+1}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}) \end{aligned}$$

while noting that $\tanh(\frac{1}{2}\log\varphi) = (\varphi-1)/(\varphi+1)$. The right hand side further simplifies into

$$\frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v}) \;=\; \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top(\mathbf{u}-\mathbf{v}) + \frac{\tanh(\frac{1}{2}\log\varphi)}{4\log\varphi}(\chi^2-(\log\varphi)^2) + \frac{\varphi-1}{2}z$$

where, for brevity, we have defined the following

$$\begin{aligned} z &= \frac{1}{\varphi+1}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}), \\ \chi &= \mathbf{v}^\top(\mathbf{u}-\mathbf{v}) - \log\varphi \;=\; (\varphi+1)z - \log\varphi. \end{aligned}$$

Recall the following inequality (Jaakkola and Jordan, 2000) which holds for any choice of $\xi \in \mathbb{R}$:

$$\log\left(\exp\left(-\frac{\xi}{2}\right)+\exp\left(\frac{\xi}{2}\right)\right) + \frac{\tanh(\frac{\xi}{2})}{4\xi}(\chi^2-\xi^2) \;\geq\; \log\left(\exp\left(-\frac{\chi}{2}\right)+\exp\left(\frac{\chi}{2}\right)\right).$$

Choose $\xi = \log\varphi$ (or, equivalently, $\xi = -\log\varphi$) and rewrite the bound as

$$\frac{\tanh(\frac{1}{2}\log\varphi)}{4\log\varphi}(\chi^2-(\log\varphi)^2) \;\geq\; \log\left(\exp\left(-\frac{\chi}{2}\right)+\exp\left(\frac{\chi}{2}\right)\right) - \log(\varphi^{\frac{1}{2}}+\varphi^{-\frac{1}{2}}).$$

Applying this bound in the formula involving the A matrix and rearranging yields

$$\begin{aligned} \frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v}) &\geq \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\exp\left(-\frac{\chi}{2}\right)+\exp\left(\frac{\chi}{2}\right)\right) - \log(\varphi^{\frac{1}{2}}+\varphi^{-\frac{1}{2}}) + \frac{\varphi-1}{2}z \\ &= \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\exp(-z+\log\varphi)+\exp(\varphi z)\right) - \log(\varphi+1) \\ &= \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\frac{\varphi}{\varphi+1}\exp(-z)+\frac{1}{\varphi+1}\exp(\varphi z)\right) \\ &= \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\frac{\varphi}{\varphi+1}\exp\left(-\frac{\mathbf{v}^\top(\mathbf{u}-\mathbf{v})}{\varphi+1}\right)+\frac{1}{\varphi+1}\exp\left(\frac{\varphi\mathbf{v}^\top(\mathbf{u}-\mathbf{v})}{\varphi+1}\right)\right) \\ &= \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\frac{\varphi+\exp(\mathbf{v}^\top(\mathbf{u}-\mathbf{v}))}{\varphi+1}\right) - \frac{\mathbf{v}^\top(\mathbf{u}-\mathbf{v})}{\varphi+1} \\ &= \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\frac{\alpha+\exp(-\mathbf{v}^\top\mathbf{v}/2+\mathbf{v}^\top\mathbf{u})}{\alpha+\exp(\mathbf{v}^\top\mathbf{v}/2)}\right) - \frac{\exp(\mathbf{v}^\top\mathbf{v}/2)}{\alpha+\exp(\mathbf{v}^\top\mathbf{v}/2)}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}) \\ &\geq \frac{\|\mathbf{u}-\mathbf{v}\|^2}{2} + \log\left(\frac{\alpha\exp(-\frac{1}{2}\|\mathbf{u}-\mathbf{v}\|^2)+\exp(-\mathbf{v}^\top\mathbf{v}/2+\mathbf{v}^\top\mathbf{u})}{\alpha+\exp(\mathbf{v}^\top\mathbf{v}/2)}\right) \\ &\quad - \frac{\exp(\mathbf{v}^\top\mathbf{v}/2)}{\alpha+\exp(\mathbf{v}^\top\mathbf{v}/2)}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}). \end{aligned}$$

In the last line, we use the fact that $1 \geq \exp(-\frac{1}{2}\|\mathbf{u}-\mathbf{v}\|^2)$. Absorbing the $\frac{1}{2}\|\mathbf{u}-\mathbf{v}\|^2$ term into the logarithm multiplicatively gives the desired inequality

$$\frac{1}{2}(\mathbf{u}-\mathbf{v})^\top A(\mathbf{u}-\mathbf{v}) \;\geq\; \log\left(\frac{\alpha+\exp(\mathbf{u}^\top\mathbf{u}/2)}{\alpha+\exp(\mathbf{v}^\top\mathbf{v}/2)}\right) - \frac{\exp(\mathbf{v}^\top\mathbf{v}/2)}{\alpha+\exp(\mathbf{v}^\top\mathbf{v}/2)}\mathbf{v}^\top(\mathbf{u}-\mathbf{v}).$$

■

Appendix B. Convergence of Sequential Quadratic Programming

Theorem 2 *Algorithm 1 finds a* $\tilde{\boldsymbol{\lambda}} \in \Lambda$ *achieving* $J(\tilde{\boldsymbol{\lambda}}) \geq (1-\varepsilon)\max_{\boldsymbol{\lambda}\in\Lambda} J(\boldsymbol{\lambda})$ *where*

$$J(\boldsymbol{\lambda}) = D\log(\alpha+1) - \sum_{d=1}^{D} \log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}^\top H_d \boldsymbol{\lambda}\right)\right) + \boldsymbol{\lambda}^\top \mathbf{1}$$

$$\text{s.t. } \boldsymbol{\lambda} \in \Lambda = \left\{ \begin{array}{ll} 0 \leq \lambda_{m,t} \leq C, & t = 1,\ldots,T_m,\ m = 1,\ldots,M \\ \sum_{t=1}^{T_m} y_{m,t}\lambda_{m,t} = 0, & m = 1,\ldots,M \end{array} \right.$$

in at most $\left\lceil \frac{\log(1/\varepsilon)}{\log\left(\min\left(1+\frac{1}{\alpha},2\right)\right)} \right\rceil$ *iterations for any* $\alpha \geq 0$ *and* $\varepsilon \in (0,1)$.

Proof Sequential quadratic programing is used to approximate $\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}\in\Lambda} J(\boldsymbol{\lambda})$. Given a current setting $\boldsymbol{\lambda}_i$ at iteration i, Theorem 1 obtains a variational quadratic bound on $J(\boldsymbol{\lambda})$ as:

$$\begin{aligned} L_i(\boldsymbol{\lambda}) &= D\log(\alpha+1) - \sum_{d=1}^{D} \log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)\right) - \sum_{d=1}^{D} \frac{\exp(\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})}{\alpha + \exp(\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})} \boldsymbol{\lambda}_i{}^\top H_d (\boldsymbol{\lambda} - \boldsymbol{\lambda}_i) \\ &\quad -\frac{1}{2}(\boldsymbol{\lambda}-\boldsymbol{\lambda}_i)^\top \left(\sum_{d=1}^{D} \frac{\tanh(\frac{1}{2}\log(\alpha\exp(-\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})))}{2\log(\alpha\exp(-\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2}))} H_d \boldsymbol{\lambda}_i \boldsymbol{\lambda}_i{}^\top H_d + H_d \right)(\boldsymbol{\lambda}-\boldsymbol{\lambda}_i) + \boldsymbol{\lambda}^\top \mathbf{1}. \end{aligned}$$

The bound satisfies $L_i(\boldsymbol{\lambda}) \leq J(\boldsymbol{\lambda})$ for all $\boldsymbol{\lambda}$ and equality is achieved when $\boldsymbol{\lambda} = \boldsymbol{\lambda}_i$. Next, we will find an upper bound $J(\boldsymbol{\lambda}) \leq U_i(\boldsymbol{\lambda})$. We first form a component of $U(\boldsymbol{\lambda})$ called $U^d(\boldsymbol{\lambda})$ that upper bounds the following component of the objective function

$$J^d(\boldsymbol{\lambda}) = -\log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}^\top H_d \boldsymbol{\lambda}\right)\right).$$

Apply Jensen's inequality for any choice of the scalar variational parameter $\zeta_d \in [0,1]$ to get

$$J^d(\boldsymbol{\lambda}) \leq -\zeta_d \log\left(\frac{\alpha}{\zeta_d}\right) - (1-\zeta_d)\log\left(\frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}^\top H_d \boldsymbol{\lambda}\right)}{(1-\zeta_d)}\right).$$

Setting the variational parameter as $\zeta_d = \alpha\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)\right)^{-1}$ produces the bound $J^d(\boldsymbol{\lambda}) \leq$

$$-\log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)\right) + \frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)}{\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)} \frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i - \frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)}{\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)} \frac{1}{2}\boldsymbol{\lambda}^\top H_d \boldsymbol{\lambda}.$$

Repeating the above for $d = 1,\ldots,D$ terms produces the overall variational upper bound

$$\begin{aligned} U_i(\boldsymbol{\lambda}) &= D\log(\alpha+1) - \sum_{d=1}^{D} \log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)\right) + \sum_{d=1}^{D} \frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)}{\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)} \frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i \\ &\quad - \sum_{d=1}^{D} \frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)}{\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)} \frac{1}{2}\boldsymbol{\lambda}^\top H_d \boldsymbol{\lambda} + \boldsymbol{\lambda}^\top \mathbf{1}. \end{aligned}$$

Clearly, $J(\boldsymbol{\lambda}) \leq U_i(\boldsymbol{\lambda})$ and equality is achieved when $\boldsymbol{\lambda} = \boldsymbol{\lambda}_i$. Thus, we have an upper quadratic bound and a lower quadratic bound which sandwich the objective as $L_i(\boldsymbol{\lambda}) \leq J(\boldsymbol{\lambda}) \leq U_i(\boldsymbol{\lambda})$. Both

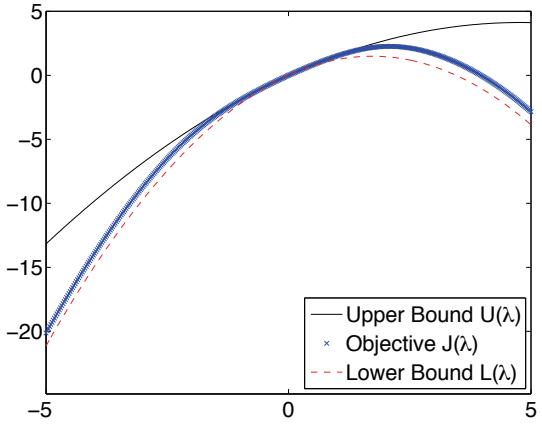


Figure 5: Upper and lower quadratic bounds on the objective function.

bounds are tight at $\boldsymbol{\lambda}_i$, in other words, $L_i(\boldsymbol{\lambda}_i) = J(\boldsymbol{\lambda}_i) = U_i(\boldsymbol{\lambda}_i)$. Figure 5 depicts the bounds. The algorithm initializes $\boldsymbol{\lambda}_0 = \mathbf{0}$ and updates via $\boldsymbol{\lambda}_{i+1} = \arg\max_{\boldsymbol{\lambda}\in\Lambda} L_i(\boldsymbol{\lambda})$ for each iteration i. Apply Lemma 3 which provides a value of $\kappa = \max(\alpha+1, 2)$ such that the following holds

$$\sup_{\boldsymbol{\lambda}\in\Lambda} L_i(\boldsymbol{\lambda}) - L_i(\boldsymbol{\lambda}_i) \quad \geq \quad \frac{1}{\kappa} \sup_{\boldsymbol{\lambda}\in\Lambda} \left(U_i(\boldsymbol{\lambda}) - U_i(\boldsymbol{\lambda}_i)\right).$$

Since $L_i(\boldsymbol{\lambda}_i) = J(\boldsymbol{\lambda}_i) = U_i(\boldsymbol{\lambda}_i), J(\boldsymbol{\lambda}_{i+1}) \geq \sup_{\boldsymbol{\lambda}\in\Lambda} L_i(\boldsymbol{\lambda})$ and $\sup_{\boldsymbol{\lambda}\in\Lambda} U_i(\boldsymbol{\lambda}) \geq J(\boldsymbol{\lambda}^*)$, we have

$$J(\boldsymbol{\lambda}_{i+1}) - J(\boldsymbol{\lambda}_i) \quad \geq \quad \frac{1}{\kappa} \left(J(\boldsymbol{\lambda}^*) - J(\boldsymbol{\lambda}_i)\right).$$

Rearrange the inequality as follows

$$J(\boldsymbol{\lambda}_{i+1}) - J(\boldsymbol{\lambda}^*) \quad \geq \quad \left(1 - \frac{1}{\kappa}\right) \left(J(\boldsymbol{\lambda}_i) - J(\boldsymbol{\lambda}^*)\right).$$

Iterate the above inequality starting at $i = 0$ to obtain

$$J(\boldsymbol{\lambda}_i) - J(\boldsymbol{\lambda}^*) \quad \geq \quad \left(1 - \frac{1}{\kappa}\right)^i \left(J(\boldsymbol{\lambda}_0) - J(\boldsymbol{\lambda}^*)\right).$$

Since the initialization used was $J(\boldsymbol{\lambda}_0) = J(\mathbf{0}) = 0$, the above simplifies as

$$J(\boldsymbol{\lambda}_i) \quad \geq \quad \left(1 - \left(1 - \frac{1}{\kappa}\right)^i\right) J(\boldsymbol{\lambda}^*).$$

Therefore, a solution that is within a relative multiplicative factor of ε implies that

$$\begin{aligned} \varepsilon &= \left(1 - \frac{1}{\kappa}\right)^i = \left(1 - \frac{1}{\max(\alpha+1, 2)}\right)^i \\ \log(1/\varepsilon) &= i \log\left(\min\left(1 + \frac{1}{\alpha}, 2\right)\right). \end{aligned}$$

Therefore, the number of iterations i required is at most $\left\lceil \frac{\log(1/\varepsilon)}{\log\left(\min\left(1+\frac{1}{\alpha}, 2\right)\right)} \right\rceil$. ■

Lemma 3 *The functions*

$$\begin{aligned}
L_i(\boldsymbol{\lambda}) &= D\log(\alpha+1) - \sum_{d=1}^{D} \log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)\right) - \sum_{d=1}^{D} \frac{\exp(\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})}{\alpha + \exp(\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})} \boldsymbol{\lambda}_i{}^\top H_d (\boldsymbol{\lambda} - \boldsymbol{\lambda}_i) \\
&\quad -\frac{1}{2}(\boldsymbol{\lambda}-\boldsymbol{\lambda}_i)^\top \left(\sum_{d=1}^{D} \mathcal{G}_d H_d \boldsymbol{\lambda}_i \boldsymbol{\lambda}_i{}^\top H_d + H_d\right)(\boldsymbol{\lambda}-\boldsymbol{\lambda}_i) + \boldsymbol{\lambda}^\top \mathbf{1}, \\
U_i(\boldsymbol{\lambda}) &= D\log(\alpha+1) - \sum_{d=1}^{D} \log\left(\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)\right) + \sum_{d=1}^{D} \frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)}{\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)} \frac{1}{2} \boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i \\
&\quad - \sum_{d=1}^{D} \frac{\exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)}{\alpha + \exp\left(\frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i\right)} \frac{1}{2} \boldsymbol{\lambda}^\top H_d \boldsymbol{\lambda} + \boldsymbol{\lambda}^\top \mathbf{1}
\end{aligned}$$

for $\mathcal{G}_d = \frac{1}{2}\tanh(\frac{1}{2}\log(\alpha\exp(-\frac{\tilde{\boldsymbol{\lambda}}^\top H_d \tilde{\boldsymbol{\lambda}}}{2})))/\log(\alpha\exp(-\frac{\tilde{\boldsymbol{\lambda}}^\top H_d \tilde{\boldsymbol{\lambda}}}{2})$ *and* $H_d \succeq \mathbf{0}$ *for* $d = 1, \ldots, D$ *satisfy*

$$\sup_{\boldsymbol{\lambda}\in\Lambda} \left(L_i(\boldsymbol{\lambda}) - L_i(\boldsymbol{\lambda}_i)\right) \geq \frac{1}{\max(\alpha+1, 2)} \sup_{\boldsymbol{\lambda}\in\Lambda} \left(U_i(\boldsymbol{\lambda}) - U_i(\boldsymbol{\lambda}_i)\right)$$

where

$$\Lambda = \begin{cases} 0 \leq \lambda_{m,t} \leq C, & t = 1, \ldots, T_m,\ m = 1, \ldots, M \\ \sum_{t=1}^{T_m} y_{m,t} \lambda_{m,t} = 0, & m = 1, \ldots, M. \end{cases}$$

Proof Rewrite the functions as follows

$$\begin{aligned}
L_i(\boldsymbol{\lambda}) - L_i(\boldsymbol{\lambda}_i) &= -\frac{1}{2}(\boldsymbol{\lambda}-\boldsymbol{\lambda}_i)^\top \Phi (\boldsymbol{\lambda}-\boldsymbol{\lambda}_i) - (\boldsymbol{\lambda}-\boldsymbol{\lambda}_i)^\top \boldsymbol{\mu} \\
U_i(\boldsymbol{\lambda}) - U_i(\boldsymbol{\lambda}_i) &= -\frac{1}{2}(\boldsymbol{\lambda}-\boldsymbol{\lambda}_i)^\top \Psi (\boldsymbol{\lambda}-\boldsymbol{\lambda}_i) - (\boldsymbol{\lambda}-\boldsymbol{\lambda}_i)^\top \boldsymbol{\mu}
\end{aligned}$$

where

$$\begin{aligned}
\Phi &= \sum_{d=1}^{D} \left(\mathcal{G}_d H_d \boldsymbol{\lambda}_i \boldsymbol{\lambda}_i{}^\top + I\right) H_d \\
\Psi &= \sum_{d=1}^{D} \frac{\exp(\frac{1}{2}\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i)}{\alpha + \exp(\frac{1}{2}\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i)} H_d \\
\boldsymbol{\mu} &= \sum_{d=1}^{D} \frac{\exp(\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})}{\alpha + \exp(\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})} \boldsymbol{\lambda}_i{}^\top H_d - \mathbf{1}.
\end{aligned}$$

Since $\mathrm{tr}(A) I \succeq A$ for matrices $A \succeq \mathbf{0}$, the following holds in the Loewner ordering sense

$$\Phi \preceq \sum_{d=1}^{D} \left(\mathcal{G}_d \boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i + 1\right) H_d.$$

Rewrite this bound as $\Phi \preceq \sum_d \phi_d H_d$ and rewrite $\Psi = \sum_d \psi_d H_d$. Consider the ratio

$$\frac{\phi_d}{\psi_d} = \frac{\alpha + \exp(\frac{1}{2}\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i)}{\exp(\frac{1}{2}\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i)} \left(\frac{\tanh(\frac{1}{2}\log(\alpha\exp(-\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2})))}{2\log(\alpha\exp(-\frac{\boldsymbol{\lambda}_i{}^\top H_d \boldsymbol{\lambda}_i}{2}))} \boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i + 1\right).$$

Define $\zeta_d = \frac{1}{2}\boldsymbol{\lambda}_i^\top H_d \boldsymbol{\lambda}_i$ and rewrite the ratio as

$$\frac{\phi_d}{\psi_d} \;=\; \frac{\alpha + \exp(\zeta_d)}{\exp(\zeta_d)} \left(\frac{\tanh(\frac{1}{2}\log(\alpha\exp(-\zeta_d)))}{\log(\alpha\exp(-\zeta_d)} \zeta_d + 1 \right).$$

It is easy to verify that this ratio is maximized when $\zeta_d \to \infty$ if $\alpha \leq 1$ and when $\zeta_d = 0$ when $\alpha > 1$. This reveals that the ratio is bounded as $\frac{\phi_d}{\psi_d} \leq \kappa$ where $\kappa = \max(\alpha+1, 2)$. Therefore, we can rewrite

$$\Phi \;\preceq\; \sum_{d=1}^{D} \phi_d H_d \;\preceq\; \sum_{d=1}^{D} \kappa\psi_d H_d \;=\; \kappa\Psi.$$

Recall the primal maximization problems of interest: $\mathbf{P}_L = \sup_{\boldsymbol{\lambda}\in\Lambda} L_i(\boldsymbol{\lambda}) - L_i(\boldsymbol{\lambda}_i)$ and $\mathbf{P}_U = \sup_{\boldsymbol{\lambda}\in\Lambda} U_i(\boldsymbol{\lambda}) - U_i(\boldsymbol{\lambda}_i)$. The constraints $\boldsymbol{\lambda} \in \Lambda$ can be summarized by linear inequalities $A\boldsymbol{\lambda} \leq \mathbf{b}$ for some A and $\mathbf{b}$. Apply the change of variables $\mathbf{z} = \boldsymbol{\lambda} - \boldsymbol{\lambda}_i$. The constraint $A(\mathbf{z} + \boldsymbol{\lambda}_i) \leq \mathbf{b}$ simplifies into $A\mathbf{z} \leq \tilde{\mathbf{b}}$ where $\tilde{\mathbf{b}} = \mathbf{b} - A\boldsymbol{\lambda}_i$. Since $\boldsymbol{\lambda}_i \in \Lambda$ is a feasible solution (which is true by construction), it is easy to show that $\tilde{\mathbf{b}} \geq \mathbf{0}$. We obtain the following equivalent primal optimization problems

$$\mathbf{P}_L = \sup_{A\mathbf{z}\leq\tilde{\mathbf{b}}} -\frac{1}{2}\mathbf{z}^\top\Phi\mathbf{z} - \mathbf{z}^\top\boldsymbol{\mu}, \quad \mathbf{P}_Z = \sup_{A\mathbf{z}\leq\tilde{\mathbf{b}}} -\frac{\kappa}{2}\mathbf{z}^\top\Psi\mathbf{z} - \mathbf{z}^\top\boldsymbol{\mu}, \quad \mathbf{P}_U = \sup_{A\mathbf{z}\leq\tilde{\mathbf{b}}} -\frac{1}{2}\mathbf{z}^\top\Psi\mathbf{z} - \mathbf{z}^\top\boldsymbol{\mu}.$$

The respective dual problems to the above are

$$\begin{aligned}
\mathbf{D}_L &= \inf_{\mathbf{y}\geq\mathbf{0}} \frac{1}{2}\mathbf{y}^\top A\Phi^{-1}A^\top\mathbf{y} + \mathbf{y}^\top A\Phi^{-1}\boldsymbol{\mu} + \mathbf{y}^\top\tilde{\mathbf{b}} + \frac{1}{2}\boldsymbol{\mu}^\top\Phi^{-1}\boldsymbol{\mu} \\
\mathbf{D}_Z &= \inf_{\mathbf{y}\geq\mathbf{0}} \frac{1}{\kappa}\frac{1}{2}\mathbf{y}^\top A\Psi^{-1}A^\top\mathbf{y} + \frac{1}{\kappa}\mathbf{y}^\top A\Psi^{-1}\boldsymbol{\mu} + \mathbf{y}^\top\tilde{\mathbf{b}} + \frac{1}{\kappa}\frac{1}{2}\boldsymbol{\mu}^\top\Psi^{-1}\boldsymbol{\mu} \\
\mathbf{D}_U &= \inf_{\mathbf{y}\geq\mathbf{0}} \frac{1}{2}\mathbf{y}^\top A\Psi^{-1}A^\top\mathbf{y} + \mathbf{y}^\top A\Psi^{-1}\boldsymbol{\mu} + \mathbf{y}^\top\tilde{\mathbf{b}} + \frac{1}{2}\boldsymbol{\mu}^\top\Psi^{-1}\boldsymbol{\mu}.
\end{aligned}$$

Due to strong duality, $\mathbf{P}_L = \mathbf{D}_L$, $\mathbf{P}_Z = \mathbf{D}_Z$ and $\mathbf{P}_U = \mathbf{D}_U$. Apply the bound $\Phi \preceq \kappa\Psi$ as follows

$$\begin{aligned}
\mathbf{P}_L &= \sup_{A\mathbf{z}\geq\tilde{\mathbf{b}}} -\frac{1}{2}\mathbf{z}^\top\Phi\mathbf{z} - \mathbf{z}^\top\boldsymbol{\mu} \\
&\geq \sup_{A\mathbf{z}\geq\tilde{\mathbf{b}}} -\frac{\kappa}{2}\mathbf{z}^\top\Psi\mathbf{z} - \mathbf{z}^\top\boldsymbol{\mu} = \mathbf{P}_Z = \mathbf{D}_Z \\
&= \inf_{\mathbf{y}\geq\mathbf{0}} \frac{1}{\kappa}\frac{1}{2}\mathbf{y}^\top A\Psi^{-1}A^\top\mathbf{y} + \frac{1}{\kappa}\mathbf{y}^\top A\Psi^{-1}\boldsymbol{\mu} + \mathbf{y}^\top\tilde{\mathbf{b}} + \frac{1}{\kappa}\frac{1}{2}\boldsymbol{\mu}^\top\Psi^{-1}\boldsymbol{\mu} \\
&= \inf_{\mathbf{y}\geq\mathbf{0}} \frac{1}{\kappa}\frac{1}{2}\mathbf{y}^\top A\Psi^{-1}A^\top\mathbf{y} + \frac{1}{\kappa}\mathbf{y}^\top A\Psi^{-1}\boldsymbol{\mu} + \frac{1}{\kappa}\mathbf{y}^\top\tilde{\mathbf{b}} + \frac{\kappa-1}{\kappa}\mathbf{y}^\top\tilde{\mathbf{b}} + \frac{1}{\kappa}\frac{1}{2}\boldsymbol{\mu}^\top\Psi^{-1}\boldsymbol{\mu} \\
&\geq \inf_{\mathbf{y}\geq\mathbf{0}} \frac{1}{\kappa}\frac{1}{2}\mathbf{y}^\top A\Psi^{-1}A^\top\mathbf{y} + \frac{1}{\kappa}\mathbf{y}^\top A\Psi^{-1}\boldsymbol{\mu} + \frac{1}{\kappa}\mathbf{y}^\top\tilde{\mathbf{b}} + \frac{1}{\kappa}\frac{1}{2}\boldsymbol{\mu}^\top\Psi^{-1}\boldsymbol{\mu} = \frac{1}{\kappa}\mathbf{D}_U = \frac{1}{\kappa}\mathbf{P}_U.
\end{aligned}$$

In the last line, we have dropped the term $\frac{\kappa-1}{\kappa}\mathbf{y}^\top\tilde{\mathbf{b}}$ since it is positive (recall that $\mathbf{y} \geq \mathbf{0}$ and $\tilde{\mathbf{b}} \geq \mathbf{0}$). Thus, $\mathbf{P}_L \geq \frac{1}{\kappa}\mathbf{P}_U$ which yields the desired inequality

$$\sup_{\boldsymbol{\lambda}\in\Lambda} L_i(\boldsymbol{\lambda}) - L_i(\boldsymbol{\lambda}_i) \;\geq\; \frac{1}{\max(\alpha+1,2)} \sup_{\boldsymbol{\lambda}\in\Lambda} U_i(\boldsymbol{\lambda}) - U_i(\boldsymbol{\lambda}_i).$$

■

Appendix C. Relation to Other Sparse Regression Methods

This section considers MED regression with a squared error loss. This will show a connection between the MED regression framework and standard regression methods such as least squares or Ridge regression, ℓ_1 regularized regression methods such as the Lasso (Tibshirani, 1996) and intermediates such as the Elastic Net (Zou and Hastie, 2005). In particular, the regularizer introduced by feature selection and kernel selection in the MED framework will be shown to resemble the Elastic Net and the Lasso and Ridge regression for appropriate choices of α. In this article, we define the ℓ_1 norm of a vector $\mathbf{w} \in \mathbb{R}^d$ as $\|\mathbf{w}\|_1 = \sum_{d=1}^{D} |\mathbf{w}(d)|$, and the ℓ_2 norm as $\|\mathbf{w}\|_2^2 = \sum_{d=1}^{D} |\mathbf{w}(d)|^2$.

Consider the ℓ_2-regularized least squares problem with input-output pairs $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}$ where $\mathbf{x}_t \in \mathbb{R}^D$ and $y_t \in \mathbb{R}$. The squared error in predicting y_t from $\mathbf{x}_t$ is minimized while also minimizing the ℓ_2 norm of the classifier. Equivalently, this can be posed as the minimization of the ℓ_2 norm of the classifier subject to a hard constraint on the total squared error obtained on the training data. We wish to estimate a regression function of the form $\hat{y} = \mathbf{w}^\top \mathbf{x} + b$ whose parameters $\mathbf{w} \in \mathbb{R}^D$ and $b \in \mathbb{R}$ are given by the following constrained minimization problem:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \text{ s.t. } \sum_{t=1}^{T} \|\mathbf{w}^\top \mathbf{x}_t + b - y_t\|^2 \leq \Upsilon$$

for some $\Upsilon \in \mathbb{R}^+$. The dual problem for the above can be obtained by noting that the solution must be of the form $\mathbf{w}^* = \sum_{t=1}^{T} \lambda_t \mathbf{x}_t$ by standard reproducing kernel Hilbert space arguments (Schölkopf and Smola, 2001). We can rewrite the optimization problem as follows:

$$\min_{\boldsymbol{\lambda}} \frac{1}{2} \sum_{t,\tau} \lambda_t \lambda_\tau \mathbf{x}_t^\top \mathbf{x}_\tau \text{ s.t. } \sum_{t=1}^{T} \left\| \sum_{\tau=1}^{T} \lambda_\tau \mathbf{x}_t^\top \mathbf{x}_\tau + \frac{1}{T} \sum_{u=1}^{T} y_u - \frac{1}{T} \sum_{u=1}^{T} \sum_{\tau=1}^{T} \lambda_\tau \mathbf{x}_u^\top \mathbf{x}_\tau - y_t \right\|^2 \leq \Upsilon \tag{3}$$

after minimization over b has been performed. Recall that the prediction function can also be written in terms of $\lambda_1, \ldots, \lambda_T$ as a function of a query datum $\mathbf{x}$ as follows:

$$\hat{y} \quad = \quad \sum_{t=1}^{T} \lambda_t \mathbf{x}_t^\top \mathbf{x} + \frac{1}{T} \sum_{t=1}^{T} y_t - \frac{1}{T} \sum_{t=1}^{T} \sum_{\tau=1}^{T} \lambda_\tau \mathbf{x}_t^\top \mathbf{x}_\tau.$$

It is possible to now consider the same manipulation that MED with feature selection produces by integrating over switches with a Bernoulli prior. This yields the following feature selection convex program that is a simple variant of least squares regression:

$$\begin{cases} \min_{\boldsymbol{\lambda}} \sum_{d=1}^{D} \log(\alpha + \exp(\frac{1}{2} \sum_t \sum_\tau \lambda_t \lambda_\tau \mathbf{x}_t(d) \mathbf{x}_\tau(d))) - D \log(\alpha + 1) \\ \text{s.t. } \sum_{t=1}^{T} \left\| \sum_{\tau=1}^{T} \lambda_\tau \mathbf{x}_t^\top \mathbf{x}_\tau + \frac{1}{T} \sum_{u=1}^{T} y_t - \frac{1}{T} \sum_{u=1}^{T} \sum_{\tau=1}^{T} \lambda_\tau \mathbf{x}_u^\top \mathbf{x}_\tau - y_t \right\|^2 \leq \Upsilon. \end{cases}$$

Similarly, the prediction rule is as follows in the feature selection variant:

$$\hat{y} \quad = \quad \sum_{t=1}^{T} \lambda_t \sum_{d=1}^{D} \hat{\mathbf{s}}(d) \mathbf{x}_t(d) \mathbf{x}(d) + \frac{1}{T} \sum_{t=1}^{T} y_t - \frac{1}{T} \sum_{t=1}^{T} \sum_{\tau=1}^{T} \lambda_\tau \sum_{d=1}^{D} \hat{\mathbf{s}}(d) \mathbf{x}_t(d) \mathbf{x}_\tau(d) \tag{4}$$

where $\hat{\mathbf{s}}(d)$ is given by:

$$\hat{\mathbf{s}}(d) \quad = \quad \frac{1}{1 + \alpha \exp(-\frac{1}{2} \sum_{t=1}^{T} \sum_{\tau=1}^{T} \lambda_t \lambda_\tau \mathbf{x}_t(d) \mathbf{x}_\tau(d))}.$$

If we let $\alpha = 0$, it is straightforward to see that we recover the standard least squares setup in Equation 3. However, this dual problem in the MED formulation is clearly inducing a different regularization on the classifier. We next investigate what primal regularizer corresponds to this change and write it in terms of the original classification parameters $\mathbf{w}$. This will show a connection to the ℓ_1 regularization popularized by the Lasso method.

First, note that the prediction $\hat{y}$ in Equation 4 can be written in terms of a primal parameter $\mathbf{w}$ as $\hat{y} = \mathbf{w}^\top \mathbf{x} + b$ if we define the parameter element-wise as:

$$\mathbf{w}(d) \quad = \quad \frac{\sum_t \lambda_t \mathbf{x}_t(d)}{1 + \alpha \exp(-\frac{1}{2}\sum_{t=1}^T \sum_{\tau=1}^T \lambda_t \lambda_\tau \mathbf{x}_t(d)\mathbf{x}_\tau(d))}.$$

Instead of an ℓ_2 norm, the MED program corresponds to minimizing the following regularizer:

$$\ell_{MED} \quad = \quad \sum_{d=1}^D \log\left(\frac{\alpha}{\alpha+1} + \frac{1}{\alpha+1}\exp(\frac{1}{2}\sum_t \sum_\tau \lambda_t \lambda_\tau \mathbf{x}_t(d)\mathbf{x}_\tau(d))\right).$$

The above can be written in terms of $\mathbf{w}$ as follows:

$$\ell_{MED}(\mathbf{w}) \quad = \quad \sum_{d=1}^D h(\mathbf{w}_d)$$

where the function $h()$ is defined implicitly by the following equation

$$\mathbf{w}(d)^2 \quad = \quad \frac{2\log(\alpha) + 2\log(\exp(h) - 1)}{(1 - 1/(\exp(h) - 1))^2}.$$

Near the origin, this function behaves like an ℓ_1 norm and, further away, behaves like an ℓ_2 norm. In Figure 6(a) we plot the function $h(\mathbf{w}(d))$ for various values of $\mathbf{w}(d)$ scaled appropriately so that $h(1) = 1$. For small α, the induced penalty on the regression parameters resembles an ℓ_2 norm. As α increases, a behavior resembling an ℓ_1 norm emerges. In intermediate settings, the MED regularizer interpolates between these two behaviors in a manner reminiscent of the so-called Elastic Net (Zou and Hastie, 2005) which uses a conic combination of ℓ_1 and ℓ_2 regularization. In addition, two-dimensional contour plots are shown comparing ℓ_{MED} to ℓ_1 and ℓ_2 regularization in Figure 6(b). While ℓ_{MED} is not identical to the Elastic Net regularization, the similarity warrants further exploration and may be useful in group Lasso and multitask settings (Turlach et al., 2005).

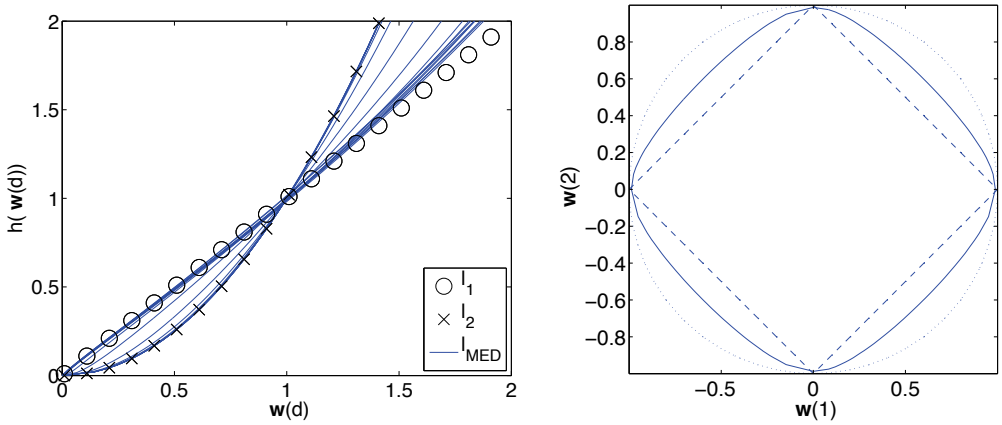
References

R.K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, December 2008.

F. Bach, G.R.G. Lanckriet, and M.I. Jordan. Multiple kernel learning, conic duality and the SMO algorithm. In *Proceedings of the International Conference on Machine Learning*, 2004.

F.R. Bach. Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.



(a) One dimensional plot (b) Two dimensional contour plot

Figure 6: Various penalties on the classifier weights. In (a), the penalties are shown as one dimensional functions of the form $h(\mathbf{w}(d))$ as α varies from $\alpha = 0$ (which mimics an ℓ_2 norm) to large α large (which resembles an ℓ_1 norm). In (b), a two-dimensional contour plot is provided showing the shape of the ℓ_1 penalty as a dashed line, the shape of the ℓ_2 penalty as a dotted line and the shape of the ℓ_{MED} penalty with $\alpha = 2$ as a solid line.

J. Baxter. Learning internal representations. In *Proceedings of the International ACM Workshop on Computational Learning Theory*, 1995.

J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12: 149–198, 2000.

S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *Proceedings of the Conference on Learning Theory and Kernel Machines*, 2003.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, 2008.

S. Boyd and L. Vanderberghe. *Convex Optimization*. Cambridge University Press, 2004.

R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On kernel target alignment. In *Advances in Neural Information Processing Systems*, volume 14, 2001.

O. Dekel, Y. Singer, and P. Long. Online multitask learning. In *Proceedings of the Conference on Learning Theory*, 2006.

T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

M. Dudik, M.E. Blei, and R.E. Schapire. Hierarchical maximum entropy density estimation. In *Proceedings of the International Conference on Machine learning*, 2007.

T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernels. *Journal of Machine Learning Research*, 6:615–637, 2005.

T. Heskes. Solving a huge number of similar tasks: A combination of multi-task learning and a hierarchical Bayesian approach. In *Proceedings of the International Conference on Machine Learning*, 1998.

T. Heskes. Empirical Bayes for learning to learn. In *Proceedings of the International Conference on Machine Learning*, 2004.

T. Jaakkola and M. Jordan. Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10:25–37, 2000.

T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *Advances in Neural Information Processing Systems*, 1999.

T. Jebara. *Machine Learning: Discriminative and Generative*. Kluwer Academic, 2003.

T. Jebara. Multi-task feature and kernel selection for SVMs. In *Proceedings of the International Conference on Machine Learning*, 2004.

T. Jebara and T. Jaakkola. Feature selection and dualities in maximum entropy discrimination. In *Uncertainty in Artifical Intelligence*, 2000.

T. Joachims. Training linear SVMs in linear time. In *Proceedings of the ACM International Conference On Knowledge Discovery and Data Mining*, pages 217–226, 2006.

K. Koh, S.J. Kim, and S. Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 8:1519–1555, July 2007.

G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan. Learning the kernel matrix with semi-definite programming. In *Proceedings of the International Conference on Machine Learning*, 2002.

J. Langford and J. Shawe-Taylor. PAC-Bayes and margins. In *Advances in Neural Information Processing Systems*, pages 423–430, 2002.

P. Long and X. Wu. Mistake bounds for maximum entropy discrimination. In *Advances in Neural Information Processing Systems*, 2004.

A. Maurer. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:117–139, December 2006.

A. Maurer. Transfer bounds for linear feature learning. *Machine Learning*, 75(3):327–350, June 2009.

D. McAllester. PAC-Bayesian model averaging. In *Proceedings of the Workshop on Computational Learning Theory*, 1999.

G. Obozinski, B. Taskar, and M.I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252, April 2010.

B. Schölkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, 2001.

B. Schölkopf, A. J. Smola, and K.R. Muller. *Advances in Kernel Methods - Support Vector Learning*, chapter Kernel principal component analysis, pages 327–352. MIT Press, 1999.

S. Shalev-Shwartz and N. Srebro. SVM optimization: Inverse dependence on training set size. In *Proceedings of the International Conference on Machine Learning*, 2008.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *Proceedings of the International Conference on Machine Learning*, 2007.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*, 2004.

S. Thrun. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems*, 1995.

S. Thrun and L.Y. Pratt. *Learning to Learn*. Kluwer Academic, 1997.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

J.A. Tropp. Just relax: Convex programming methods for identifying sparse signals. *IEEE Transactions on Information Theory*, 51(3):1030–1051, 2006.

B.A. Turlach, W.N. Venables, and S.J. Wright. Simultaneous variable selection. *Technometrics*, pages 349–363, 2005.

M. Wainwright, P. Ravikumar, and J. Lafferty. High-dimensional graphical model selection using l1-regularized logistic regression. In *Advances in Neural Information Processing Systems*, 2007.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems*, 2000.

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society B*, 68:49–67, 2006.

J. Zhu, E.P. Xing, and B. Zhang. Laplace maximum margin Markov networks. In *Proceedings of the International Conference on Machine Learning*, 2008.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67(2):301–320, 2005.

Journal of Machine Learning Research 12 (2011) 111-139 Submitted 11/09; Revised 10/10; Published 1/11

Bayesian Generalized Kernel Mixed Models

Zhihua Zhang ZHZHANG@ZJU.EDU.CN
Guang Dai GUANG.GDAI@GMAIL.COM
College of Computer Science and Technology
Zhejiang University
Hangzhou, Zhejiang 310027, China

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Editor: Neil Lawrence

Abstract

We propose a fully Bayesian methodology for generalized kernel mixed models (GKMMs), which are extensions of generalized linear mixed models in the feature space induced by a reproducing kernel. We place a mixture of a point-mass distribution and Silverman's g-prior on the regression vector of a generalized kernel model (GKM). This mixture prior allows a fraction of the components of the regression vector to be zero. Thus, it serves for sparse modeling and is useful for Bayesian computation. In particular, we exploit data augmentation methodology to develop a Markov chain Monte Carlo (MCMC) algorithm in which the reversible jump method is used for model selection and a Bayesian model averaging method is used for posterior prediction. When the feature basis expansion in the reproducing kernel Hilbert space is treated as a stochastic process, this approach can be related to the Karhunen-Loève expansion of a Gaussian process (GP). Thus, our sparse modeling framework leads to a flexible approximation method for GPs.

Keywords: reproducing kernel Hilbert spaces, generalized kernel models, Silverman's g-prior, Bayesian model averaging, Gaussian processes

1. Introduction

Supervised learning based on reproducing kernel Hilbert spaces (RKHSs) has become increasingly popular since the support vector machine (SVM) (Vapnik, 1998) and its variants such as penalized kernel logistic regression models (Zhu and Hastie, 2005) have been proposed. Sparseness has also emerged as a significant theme generally associated with RKHS methods. The SVM naturally embodies sparseness due to its use of the hinge loss function. Penalized kernel logistic regression models, however, are not naturally sparse. Thus, Zhu and Hastie (2005) proposed a methodology that they refer to as the *import vector machine* (IVM), where a fraction of the training data—called *import vectors* by analogy to the support vectors of the SVM—are used to index kernel basis functions.

Kernel supervised learning methods can be unified using the tools of regularization theory (Hastie et al., 2001). The regularization term is usually defined as the L_1 or L_2 norm of the vector of regression coefficients. From a Bayesian standpoint, this term is obtained from assigning a Gaussian or Laplacian prior to the regression vector. Moreover, using logarithmic scoring rules (Bernardo and

©2011 Zhihua Zhang, Guang Dai and Michael I. Jordan.

Smith, 1994), a loss function can often be viewed as the negative conditional log-likelihood. This perspective leads to interpreting regularization methods in terms of maximum *a posteriori* (MAP) estimation, and has motivated recent Bayesian interpretations of kernel methods (Tipping, 2001; Sollich, 2001; Mallick et al., 2005; Chakraborty et al., 2005; Zhang and Jordan, 2006; Pillai et al., 2007; Liang et al., 2009; MacLehose and Dunson, 2009).

Although the use of either the hinge loss function or L_1 regularization is an effective tool for achieving sparsity in the frequentist paradigm (Vapnik, 1998; Tibshirani, 1996), in the Bayesian setting the corresponding prior yields posteriors that cannot be computed in closed form. In the Bayesian methods of Mallick et al. (2005), for example, since conjugate priors for the regression vector do not exist, a sampling methodology based on data augmentation was employed to update the regression vector. In the Bayesian lasso (Park and Casella, 2008) or the Bayesian elastic net (Li and Lin, 2010), Gibbs sampling was used, based on assumptions of normality and independence. Given that an appeal to sampling methods must be made, it is not clear that mimicking frequentist methods is the best way to achieve sparsity within the Bayesian paradigm. Indeed, explicit support-vector selection or variable selection is not straightforward for these existing Bayesian approaches, and sparsity is often enforced in an ad hoc manner via Bayesian credible intervals (Park and Casella, 2008; Li and Lin, 2010).

In this paper we propose *generalized kernel models* (GKMs) as a framework in which sparsity can be given an explicit treatment and in which a fully Bayesian methodology can be carried out. The GKM is derived from generalized linear models (GLMs) (McCullagh and Nelder, 1989) in the RKHS. We define *active vectors* to be those input vectors that are indexed by the nonzero components of the regression vector in GKMs.[1] We assign to the regression vector a mixture of the point-mass distribution and a prior which we refer to as the *Silverman g-prior* (Silverman, 1985). Our use of this prior is based on three facts. First, the Silverman g-prior can induce an empirical RKHS norm on the training data (see Section 2.2). Second, posterior consistency results are available for Bayesian estimation procedures based on the Silverman g-prior (Zhang et al., 2008). Third, the mixture of the point-mass prior and the Silverman g-prior allows a fraction of regression coefficients in question to be zero and thus provides an explicit Bayesian approach to the selection of active vectors.

It is worth noting that the Silverman g-prior is related to the Zellner g-prior (Zellner, 1986), which has been widely applied to Bayesian variable selection and Bayesian model selection (Smith and Kohn, 1996; George and McCulloch, 1997; Kohn et al., 2001; Nott and Green, 2004; Sha et al., 2004) because of its computational tractability in evaluating marginal likelihoods.

We develop Bayesian approaches to parameter estimation, model selection and response prediction for the GKM. In particular, motivated by the use of the data augmentation methodology in Bayesian GLMs (Albert and Chib, 1993; Holmes and Held, 2006), we exploit this methodology to devise an MCMC algorithm for our Bayesian GKMs. The algorithm uses a reversible jump procedure (Green, 1995) for the automatic selection of active vectors and a Bayesian model averaging method (Raftery et al., 1997) for the posterior prediction of future observations. We show that our algorithm is amenable to low-rank matrix update techniques (see Section 3.2) that make it computationally feasible even for large data sets.

Another development in Bayesian kernel methods is based on Gaussian processes (GPs), which provide a general approach to assigning prior distributions to functions for nonparametric modeling.

1. Our "active vectors" are the analogs of import vectors for the IVM and support vectors for the SVM.

In geostatistics, GPs have been seen numerous applications to spatial statistical analysis under the name of "kriging." Diggle et al. (1998) broadened the scope of kriging by exploiting the combination of kriging and GLMs. In the machine learning community, ideas related to kriging and its extensions have been widely exploited in Bayesian treatments of classification and regression problems (Williams and Barber, 1998; Neal, 1999; Rasmussen and Williams, 2006). In these problems the data in question are not necessarily spatial. A major concern with GPs is the computational burden for large data sets. Thus, sparse approximations, such as the "subset of regressors," the Nyström method, the informative vector machine, the "subset of data" and the "data squashing" technique, are generally used to mitigate the computational burden (Williams and Seeger, 2001; Smola and Bartlett, 2001; Lawrence et al., 2003; Snelson, 2007).

Building on existing connections between kernel methods and GP-based models (see, e.g., Pillai et al., 2007), we use the Karhunen-Loève expansion of the Gaussian process to explore relationships between our Bayesian GKMs and GP-based classification. In particular, we show that our reversible jump method can be used to implement a "subset of regressors" approximation method for GP-based classification.

The rest of this paper is organized as follows. Section 2 presents a Bayesian framework for kernel supervised learning. Sections 3 and 4 present the MCMC algorithm for fully Bayesian GKMs and sparse GP classifiers, respectively. The experimental analysis is then presented in Section 5. Two extensions and some conclusions are given in Sections 6 and 7, respectively.

2. A Bayesian Approach for Kernel Supervised Learning

We start with a supervised learning problem over a set of training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^p$ is an input vector and y_i is a univariate continuous output for the regression problem or binary output for the classification problem. Our current concern is to learn a predictive function $f(\mathbf{x})$ from the training data.

Suppose $f = u + h \in (\{1\} + \mathcal{H}_K)$ where $\mathcal{H}_K$ is an RKHS. Estimating $f(\mathbf{x})$ from data is formulated as a regularization problem of the form

$$\min_{f \in \mathcal{H}_K} \left\{ \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \frac{g}{2} \|h\|^2_{\mathcal{H}_K} \right\}, \tag{1}$$

where $L(y, f(\mathbf{x}))$ is a loss function, $\|h\|^2_{\mathcal{H}_K}$ is the RKHS norm and $g > 0$ is the regularization parameter. By the representer theorem (Wahba, 1990), the solution for (1) is of the form

$$f(\mathbf{x}) = u + \sum_{j=1}^{n} \beta_j K(\mathbf{x}, \mathbf{x}_j), \tag{2}$$

where u is called an offset term, $K(\cdot,\cdot)$ is the kernel function and the β_j are referred to as regression coefficients. Noticing that $\|h\|^2_{\mathcal{H}_K} = \sum_{i,j=1}^n K(\mathbf{x}_i, \mathbf{x}_j)\beta_i\beta_j$ and substituting (2) into (1) we obtain the minimization problem with respect to (w.r.t.) the u and β_j as

$$\min_{u,\beta} \left\{ \frac{1}{n} \sum_{i=1}^{n} L(y_i, u + \mathbf{k}_i'\beta) + \frac{g}{2} \beta' \mathbf{K} \beta \right\}, \tag{3}$$

where $\beta = (\beta_1, \ldots, \beta_n)'$ is an $n \times 1$ regression vector and $\mathbf{K} = [\mathbf{k}_1, \ldots, \mathbf{k}_n]$ is the $n \times n$ kernel matrix with $\mathbf{k}_i = (K(\mathbf{x}_i, \mathbf{x}_1), \ldots, K(\mathbf{x}_i, \mathbf{x}_n))'$. Since $\mathbf{K}$ is symmetric and positive semidefinite, the term $\beta' \mathbf{K} \beta$ is in fact an empirical RKHS norm w.r.t. the training data.

The predictive function $f(\mathbf{x})$ in (2) is based on a basis expansion of kernel functions. We now show that the predictive function can also be expressed by a basis expansion of feature functions. Given a Mercer reproducing kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, there exists a corresponding mapping (say ψ) from the input space $\mathcal{X}$ to a feature space (say $\mathcal{F} \subset \mathbb{R}^r$). That is, we have a vector-valued function $\psi(\mathbf{x}) = (\psi_1(\mathbf{x}), \ldots, \psi_r(\mathbf{x}))'$, which is called the *feature vector* of $\mathbf{x}$, such that $K(\mathbf{x}_i, \mathbf{x}_j) = \psi(\mathbf{x}_i)'\psi(\mathbf{x}_j)$. By the *Mercer-Hilbert-Schmidt Theorem* (Wahba, 1990), we know that there exists an orthogonal sequence of continuous eigenfunctions $\{\phi_j\}$ in the square integrable Hilbert functional space $L_2(\mathcal{X})$ and eigenvalues $l_1 \geq l_2 \geq \ldots \geq 0$. Furthermore, we have a definition of the feature functions $\psi : \mathcal{X} \to L_2(\mathcal{X})$ as $\psi(\mathbf{x}) = \left\{\sqrt{l_j}\phi_j(\mathbf{x})\right\}_{j=1}^{r}$. That is, $\psi_j(\mathbf{x}) = \sqrt{l_j}\phi_j(\mathbf{x})$. Thus the $\psi_j(\mathbf{x})$ constitute a set of basis functions of $L_2(\mathcal{X})$. Consequently, they can be used to express the predictive function as follows:

$$f(\mathbf{x}) = u + \sum_{k=1}^{r} b_k \psi_k(\mathbf{x}) = u + \psi(\mathbf{x})'\mathbf{b}, \tag{4}$$

where $\mathbf{b} = (b_1, \ldots, b_r)'$. There are possibly infinitely many basis functions in (4) because r is possibly infinite. In the case that r is infinite, one may use a finite-dimensional approximation to $f(\mathbf{x})$ by keeping the first n $\psi_j(\mathbf{x})$'s and setting the remaining b_j, $j > n$ to zero (Zhang et al., 2007). Now letting $\mathbf{b} = \Psi'\beta$, we re-derive (2) from (4) due to $\mathbf{K} = \Psi\Psi'$ where $\Psi = [\psi(\mathbf{x}_1), \ldots, \psi(\mathbf{x}_n)]'$.

2.1 Generalized Kernel Models

Using the logarithmic scoring rule (Bernardo and Smith, 1994), the loss $L(y, f(\mathbf{x}))$ can be viewed as a negative conditional log-likelihood. This motivates us to construct the following model

$$y \sim p(y|\mu) \quad \text{with} \quad \mu = \tau(u + \mathbf{k}'\beta), \tag{5}$$

where $\tau(\cdot)$ is a known link function and $\mathbf{k} = (K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_n))'$. This model can be obtained from the model

$$y \sim p(y|\mu) \quad \text{with} \quad \mu = \tau(u + \psi(\mathbf{x})'\mathbf{b}) \tag{6}$$

by using the transformation $\mathbf{b} = \Psi'\beta$. Since the model in (6) is a GLM in the feature space, we call model (5) the *generalized kernel model* (GKM).

GKMs provide a unifying framework for kernel-based regression and classification. With different $p(y|\mu)$ and τ, we have different kernel models. In the regression problem, $p(y|\mu)$ is usually normal and τ is the identity function.

In this paper we are mainly concerned with the classification problem where y is encoded as a binary value, that is, $y \in \{0, 1\}$. We thus model $p(y|\mu)$ as Bernoulli distribution:

$$p(y|\mu) = \mu^y(1-\mu)^{1-y} = [\tau(u + \mathbf{k}'\beta)]^y[1 - \tau(u + \mathbf{k}'\beta)]^{1-y}.$$

Typically, τ is either the logistic link $\tau(z) = \frac{\exp(z)}{1+\exp(z)}$ or the probit link $\tau(z) = \Phi(z)$, the cumulative distribution function of a standard normal variable. The probit link is widely used in Bayesian GLMs due to its tractability in calculating the marginal likelihood. In our fully Bayesian GKMs in Section 3, we will use this link.

2.2 Silverman's g-prior

Assume that the b_k are independent Gaussian variables with $E(b_k) = 0$ and $E(b_k^2) = g^{-1}$, that is, $\mathbf{b} \sim N_r(\mathbf{0},\ g^{-1}\mathbf{I}_r)$. Here and later, we denote by $\mathbf{I}_m$ the $m \times m$ identity matrix, by $\mathbf{1}_m$ the $m \times 1$ vector

of ones, and by $\mathbf{0}$ the zero vector or matrix with appropriate size. Because of $\mathbf{b} = \Psi'\beta$, we have $\beta = \mathbf{K}^{-1}\Psi\mathbf{b}$. As a result, the prior for β is $\beta \sim N_n\left(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}\right)$ due to $\mathbf{K}^{-1}\Psi\Psi'\mathbf{K}^{-1} = \mathbf{K}^{-1}$. It is possible that the kernel matrix $\mathbf{K}$ is singular. For such a $\mathbf{K}$, we use its Moore-Penrose inverse $\mathbf{K}^{+}$ instead and still have $\mathbf{K}^{+}\mathbf{K}\mathbf{K}^{+} = \mathbf{K}^{+}$. The prior distribution for β becomes a singular normal distribution (Mardia et al., 1979). In either case, we use $\mathbf{K}^{-1}$ for notational simplicity.

The prior $N_n\left(\mathbf{0},\, \mathbf{K}^{-1}\right)$ for β was first proposed by Silverman (1985) in his Bayesian formulation of spline smoothing. Thus, Zhang et al. (2008) referred to the prior $\beta \sim N_n\left(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}\right)$ as the *Silverman g-prior* because it is related to the Zellner g-prior (Zellner, 1986). Since the prior density of β is proportional to $\exp(-g\beta'\mathbf{K}\beta/2)$, the Silverman g-prior is design-dependent. Moreover, the regularization term $g\beta'\mathbf{K}\beta/2$ in (3) is readily derived from this prior.

When $\mathbf{K}$ is singular, by analogy to the *generalized singular g-prior* (*gsg*-prior) (West, 2003) we call $N_n\left(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}\right)$ a *generalized Silverman g-prior*. It is worth pointing out that Green (1985) argued that the definition of Silverman's prior is implicit. We have presented an explicit derivation of this prior. Like the Zellner g-prior (Zellner, 1986; Liang et al., 2008), the Silverman g-prior has only a single shared global scaling parameter g. Thus, the prior induces a global shrinkage rule.

2.3 Sparse Models

Recall that the number of active vectors is equal to the number of nonzero components of β. That is, if $\beta_j = 0$, the jth input vector is excluded from the basis expansion in (2), otherwise the jth input vector is an active vector. We are thus interested in a prior for β which allows some components of β to be zero. In particular, we assign a point-mass mixture prior to β built on the Silverman g-prior.

We introduce an indicator binary vector $\gamma = (\gamma_1, \ldots, \gamma_n)'$ such that $\gamma_j = 1$ if $\mathbf{x}_j$ is an active vector and $\gamma_j = 0$ if it is not. Let $n_\gamma = \sum_{j=1}^{n} \gamma_j$ be the number of active vectors, and let $\mathbf{K}_\gamma$ be the $n \times n_\gamma$ submatrix of $\mathbf{K}$ consisting of those columns of $\mathbf{K}$ for which $\gamma_j = 1$. We further let $\mathbf{K}_{\gamma\gamma}$ be the $n_\gamma \times n_\gamma$ submatrix of $\mathbf{K}_\gamma$ consisting of those rows of $\mathbf{K}_\gamma$ for which $\gamma_j = 1$, and β_γ and $\mathbf{k}_\gamma$ be the corresponding $n_\gamma \times 1$ subvectors of β and $\mathbf{k}$. Based on GKMs in (5) and the Silverman g-prior, we thus obtain the following sparse model

$$y \sim p(y|\tau(f(\mathbf{x}))) \quad \text{with} \quad f(\mathbf{x}) = u + \mathbf{k}_\gamma'\beta_\gamma \text{ and } \beta_\gamma \sim N_{n_\gamma}(\mathbf{0},\, g^{-1}\mathbf{K}_{\gamma\gamma}^{-1}). \tag{7}$$

In the existing literature for Bayesian sparse classification and regression (Tipping, 2001; Figueiredo, 2003; Park and Casella, 2008; Hans, 2009; Li and Lin, 2010; Carvalho et al., 2010), a typical choice of the prior on β is the class of multivariate scale mixtures of normals. The resulting shrinkage rule is derived by mixing over a set of local scaling parameters. This differ from our global shrinkage rule. See Carvalho et al. (2010) for further discussion of sparsity priors.

3. Methodology

In this section we present a fully Bayesian GKM (FBGKM) based on (7). Since $p(y|\tau(f(\mathbf{x})))$ is non-normal for the classification problem, conjugate priors for β usually do not exist. In order to facilitate the implementation of Bayesian inference in this setting, we make use of the data augmentation methodology which has been used by Albert and Chib (1993) for Bayesian GLMs and by Mallick et al. (2005) for their Bayesian SVMs. The basic idea is to introduce auxiliary variables linking y and the model parameters. We apply this methodology to our FBGKM.

3.1 Hierarchical Models

Let $\mathbf{s} = (s_1, \ldots, s_n)'$ be a vector of auxiliary variables corresponding to the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$. We in particular define

$$\mathbf{s} = u\mathbf{1}_n + \mathbf{K}_\gamma \beta_\gamma + \varepsilon \text{ with } \varepsilon \sim N_n(\mathbf{0}, \sigma^2 \mathbf{I}_n).$$

Since τ is defined as the probit link in our FBGKM, we have $\sigma^2 = 1$ and

$$y_i = \begin{cases} 1 & \text{if } s_i > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Given $\mathbf{s}$, $\mathbf{y} = (y_1, \ldots, y_n)'$ is independent of u, β and γ. Consequently, we can assign conjugate priors for these parameters and perform an efficient Bayesian inference.

Firstly, we assume $u \sim N(0, \eta^{-1})$ and $g \sim Ga(a_g/2, b_g/2)$ where $Ga(a,b)$ represents a gamma distribution. Let $\tilde{\beta}_\gamma = (u, \beta_\gamma')'$. We thus have

$$\widetilde{\beta}_\gamma \sim N_{n_\gamma+1}(\mathbf{0}, \Sigma_\gamma^{-1}) \text{ with } \Sigma_\gamma = \begin{bmatrix} \eta & \mathbf{0} \\ \mathbf{0} & g\mathbf{K}_{\gamma\gamma} \end{bmatrix}.$$

By integrating out $\widetilde{\beta}_\gamma$, the marginal distribution of $\mathbf{s}$ conditional on γ is normal, namely,

$$p(\mathbf{s}|\gamma) = N_n(\mathbf{0}, \mathbf{Q}_\gamma) \tag{8}$$

with $\mathbf{Q}_\gamma = \mathbf{I}_n + \widetilde{\mathbf{K}}_\gamma \Sigma_\gamma^{-1} \widetilde{\mathbf{K}}_\gamma'$ where $\widetilde{\mathbf{K}}_\gamma = [\mathbf{1}_n, \mathbf{K}_\gamma]$ $(n \times (n_\gamma + 1))$. Bayes theorem yields the following distribution of $\widetilde{\beta}_\gamma$ conditional on $\mathbf{s}$ and γ:

$$[\widetilde{\beta}_\gamma | \mathbf{s}, \gamma] \sim N_{n_\gamma+1}(\Upsilon_\gamma^{-1} \widetilde{\mathbf{K}}_\gamma' \mathbf{s}, \Upsilon_\gamma^{-1}), \tag{9}$$

where $\Upsilon_\gamma = \widetilde{\mathbf{K}}_\gamma' \widetilde{\mathbf{K}}_\gamma + \Sigma_\gamma$.

Secondly, the kernel function K is assumed to be indexed by hyperparameters θ (see, e.g., Mallick et al., 2005). For example, the Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\theta^2)$ is a function of the width parameter θ. For simplicity, the dependence of K on θ will be left implicit henceforth. If θ is p-dimensional, we take a uniform prior for each element of θ on $[a_{\theta_j}, b_{\theta_j}]$. Namely,

$$\theta \sim \prod_{j=1}^p U(a_{\theta_j}, b_{\theta_j}).$$

Thirdly, as in Kohn et al. (2001) and Nott and Green (2004), we assign an independent Bernoulli prior to each component of γ, namely,

$$p(\gamma|\alpha) = \prod_{j=1}^n \alpha^{\gamma_j}(1-\alpha)^{1-\gamma_j} = \alpha^{n_\gamma}(1-\alpha)^{n-n_\gamma},$$

where $\alpha \in (0,1)$. It is natural to place a Beta prior on α, $\alpha \sim B(a_\alpha, b_\alpha)$. Marginalizing out α results in the following prior on γ:

$$p(\gamma) = \frac{Be(n_\gamma + a_\alpha, n - n_\gamma + b_\alpha)}{Be(a_\alpha, b_\alpha)}, \tag{10}$$

where $Be(\cdot,\cdot)$ is the Beta function. Kohn et al. (2001) proposed a method of selecting the hyperparameters a_α and b_α by controlling the value of n_γ. In the following experiments, we use the uninformative fixed specification $a_\alpha = 1$ and $b_\alpha = 1$.

Finally, we assume that η follows $Ga(a_\eta/2,\ b_\eta/2)$ and we shall keep the hyperparameters a_η, b_η, a_g and b_g fixed in this paper. In summary, we form a hierarchical model in which the joint density of all variables mentioned takes the form

$$p(\mathbf{y},\mathbf{s},\gamma,u,\beta,\theta,\eta,g) = p(\eta)p(g)p(\gamma)p(\theta)p(u|\eta)p(\beta|g,\gamma,\theta)p(\mathbf{s}|u,\beta,\theta,\gamma)p(\mathbf{y}|\mathbf{s}).$$

The corresponding directed acyclic graph is shown in Figure 1.

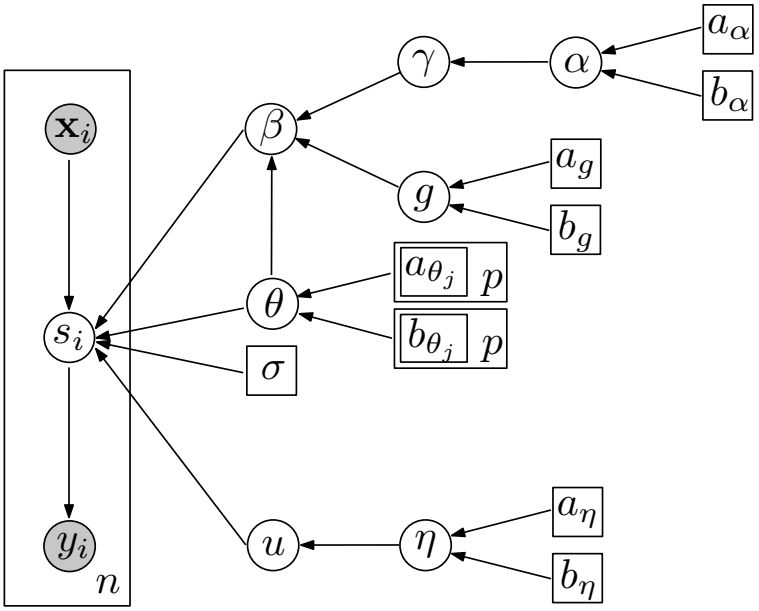


Figure 1: A graphical representation for the hierarchical model.

3.2 Inference

Our goal is to generate realizations of parameters from the conditional joint density $p(\mathbf{s},u,\beta,\gamma,g|\mathbf{y})$ via an MCMC algorithm. In order to speed up mixing of the MCMC, we use marginal posterior distributions whenever possible. Our MCMC algorithm consists of the following steps.

Start Give a_η, b_η, a_g and b_g, and initialize $\mathbf{s}$, γ, g, η, u and β_γ.

Step (a) Impute each s_i from $p(s_i|y_i,u,\beta_\gamma)$.

Step (b) Update η, g, $\widetilde{\beta}_\gamma$ and θ according to $p(\eta|u)$, $p(g|\beta_\gamma)$, $p(\widetilde{\beta}_\gamma|\mathbf{s},\gamma,\eta,g)$ and $p(\theta|\mathbf{s},\gamma)$, respectively.

Step (c) Update γ from $p(\gamma|\mathbf{s})$.

Step (a) is to draw $\mathbf{s}$ from $p(\mathbf{s}|\mathbf{y}, u, \beta_\gamma)$. We perform this step by using a technique which was proposed by Holmes and Held (2006) for the conventional probit regression. In particular, $\mathbf{s}$ is updated from its marginal distribution having integrated over $\tilde{\beta}_\gamma$; that is, s_i is generated from $p(s_i|\mathbf{s}_{-i}, y_i, \gamma)$ where $\mathbf{s}_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)'$. The details of this procedure is given in Appendix A. Please also refer to Holmes and Held (2006).

We now consider the updates of $\widetilde{\beta}_\gamma$, η and g. Given $\mathbf{s}$, these parameters are independent of $\mathbf{y}$, so their updates are based on $p(\widetilde{\beta}_\gamma, \eta, g|\mathbf{s}, \gamma)$. Hence, we update $\widetilde{\beta}_\gamma$ from $[\widetilde{\beta}_\gamma|\mathbf{s}, \gamma, \eta, g] \sim N_{n_\gamma+1}(\Upsilon_\gamma^{-1}\widetilde{\mathbf{K}}_\gamma'\mathbf{s},\ \Upsilon_\gamma^{-1})$. Since g is only dependent on β_γ and the prior is conjugate, we use the Gibbs sampler to update g from its conditional distribution, which is given by

$$[g|\beta_\gamma] \sim Ga\Big(\frac{a_g + n_\gamma}{2}, \frac{b_g + \beta_\gamma'\mathbf{K}_{\gamma\gamma}\beta_\gamma}{2}\Big).$$

The update of η is obtained from its conditional distribution as

$$[\eta|u] \sim Ga\Big(\frac{a_\eta + 1}{2}, \frac{b_\eta + u^2}{2}\Big).$$

In order to update θ, we need to use an MH sampler. We write the marginal conditional distribution of θ as

$$p(\theta|\mathbf{s}, \gamma) \propto p(\mathbf{s}|\gamma, \eta, g, \theta)p(\theta),$$

where $p(\mathbf{s}|\gamma, \eta, g, \theta)$ is given by (8). In the following experiments (see Section 5.3), the proposal distribution is specified as a Gaussian distribution with the current value of θ as mean and 0.2 as variance. Let θ^* denote the proposed move from the current θ. Then this move is accepted with probability

$$\min\Big\{1,\ \frac{p(\mathbf{s}|\gamma, \eta, g, \theta^*)}{p(\mathbf{s}|\gamma, \eta, g, \theta)}\Big\}.$$

This acceptance probability involves the calculations of the inverses and determinants of both $\mathbf{Q}_\gamma$ and $\mathbf{Q}_\gamma^*$, where $\mathbf{Q}_\gamma^*$ is obtained from $\mathbf{Q}_\gamma$ with θ^* replacing θ. To reduce computational costs, we employ the formulas in (11) which is given below for computing these inverses and determinants. Our Bayesian estimation method for the kernel parameter θ is more efficient than that given in BSVM and CSVM (Mallick et al., 2005), in which computing the inverses and determinants of two consecutive full kernel matrices $\mathbf{K}$ and $\mathbf{K}^*$ is required at each sweep of MCMC sampling.

Step (c) is used for the automatic choice of active vectors. To implement this step, we borrow a method devised by Nott and Green (2004). This method was derived from the reversible jump methodology of Green (1995). Specifically, we generate a proposal γ^* from the current value of γ by one of three possible moves:

Birth move randomly choose a 0 in γ and change it to 1;

Death move randomly choose a 1 in γ and change it to 0;

Swap move randomly choose a 0 and a 1 in γ and switch them.

The acceptance probability for each move is

$$\min\{1, \text{likelihood ratio} \times \text{prior ratio} \times \text{proposal ratio}\}.$$

Letting $k = n_\gamma$, we denote the probabilities of birth, death and swap by b_k, d_k and $1-b_k-d_k$, respectively. For birth, death and swap moves, the acceptance probabilities are

$$\min\left\{1, \frac{p(\mathbf{s}|\gamma^*)p(\gamma^*)d_{k+1}(n-k)}{p(\mathbf{s}|\gamma)p(\gamma)b_k(k+1)}\right\},$$

$$\min\left\{1, \frac{p(\mathbf{s}|\gamma^*)p(\gamma^*)b_{k-1}k}{p(\mathbf{s}|\gamma)p(\gamma)d_k(n-k+1)}\right\},$$

$$\min\left\{1, \frac{p(\mathbf{s}|\gamma^*)p(\gamma^*)}{p(\mathbf{s}|\gamma)p(\gamma)}\right\},$$

where $p(\mathbf{s}|\gamma)$ and $p(\gamma)$ are given in (8) and (10). In our experiments we set $b_0 = 1$ and $d_0 = 0$, $b_k = d_k = 0.3$ for $1 \le k \le k_{\max}-1$, and $d_k = 1$ and $b_k = 0$ for $k_{\max} \le k \le n$. Here, $k_{\max}$ is a specified maximum number of active vectors such that $k_{\max} \le n$.

An alternative to this approach is the stochastic search method of George and McCulloch (1997). This method also employs birth, death and swap moves; it differs from the reversible jump procedure because it does not incorporate the probabilities of birth, death and swap into its acceptance probabilities.

Recall that the main computational burden of our MCMC algorithm comes from the calculations of the determinant and inverse of $\mathbf{Q}_\gamma$ ($\mathbf{Q}_{\gamma^*}$) during the MCMC sweeps. It is worth noting that when n is relatively large, we can reduce the computational burden by giving $k_{\max}$ a value far less than n, that is, $k_{\max} \ll n$, and then computing:

$$\mathbf{Q}_\gamma^{-1} = \mathbf{I}_n - \widetilde{\mathbf{K}}_\gamma \Upsilon_\gamma^{-1} \widetilde{\mathbf{K}}_\gamma' \quad \text{and} \quad |\mathbf{Q}_\gamma| = |\Upsilon_\gamma||\Sigma_\gamma|^{-1} = \eta^{-1} g^{-n_\gamma} |\mathbf{K}_{\gamma\gamma}|^{-1} |\Upsilon_\gamma|. \tag{11}$$

For example, for both the USPS and NewsGroups data sets used in our experiments, we set $k_{\max} = 200 \ll n$. In this setting, we always have $n_\gamma \le k_{\max} \ll n$. Since Υ_γ and $\mathbf{K}_{\gamma\gamma}$ are $(n_\gamma+1)\times(n_\gamma+1)$ and $n_\gamma \times n_\gamma$, these formulas for $\mathbf{Q}_\gamma^{-1}$ and $|\mathbf{Q}_\gamma|$ are feasible computationally. This is an advantage over the stochastic search method of George and McCulloch (1997). Finally, in the reversible jump method, the matrices obtained before and after each move only change a column and a row. Thus, it is possible to exploit rank-one matrix update techniques to make the method still more efficient.

3.3 Prediction

Given a new input vector $\mathbf{x}_*$, we need to predict its label y_*. The posterior predictive distribution of y_* is

$$p(y_*|\mathbf{x}_*, \mathbf{y}) = \int p(y_*|\mathbf{x}_*, \tilde{\beta}_\gamma, \mathbf{y}) p(\tilde{\beta}_\gamma|\mathbf{y}) d\tilde{\beta}_\gamma.$$

We know that this integral cannot be computed in closed form. Moreover, it is intractable to select the model which is parameterized by β_γ for prediction. An intuitive approach is to choose a model with a value of γ having the highest posterior probability among those γ that appear during the MCMC sweeps. However, this is expensive in terms of memory because γ takes 2^n possible distinct values. To deal with this problem, we use a Bayesian model averaging method (Raftery et al., 1997) for posterior prediction.

The Bayesian model averaging method is based on the MCMC sampling process. Specifically, we have

$$p(y_* = 1|\mathbf{x}_*, \mathbf{y}) \approx \frac{1}{T} \sum_{t=1}^{T} p\left(y_* = 1 \middle| \mathbf{y}, \mathbf{x}_*, u^{(t)}, \beta_\gamma^{(t)}\right).$$

Here $(\cdot)^{(t)}$ is the tth MCMC realization of $(\cdot)$, which is taken at every Mth sweep after the burn-in of the MCMC algorithm. In the following experiments, we run the MCMC algorithm for $10,000$ sweeps, discard the first $5,000$ as the burn-in, and retain every 5th (i.e., $M = 5$) realization of parameters after the burn-in for inference and prediction. This implies that the Bayesian model averaging method uses $1,000$ ($T = (10,000-5,000)/5$) active sets for prediction.

We should point out that our Bayesian model does not treat the training and test as two separate procedures. In fact, our reversible jump MCMC algorithm deals with parameter estimation, model selection and posterior prediction jointly in a single paradigm. Moreover, the reversible jump method is a sequential approach for model selection and posterior prediction. This implies that after the burn-in the selection of active vectors and the prediction of responses are simultaneously implemented. Thus, the MCMC algorithm does not require extra computational complexity for the prediction of responses.

4. Sparse Gaussian Processes for Classification

In nonparametric Bayesian methods for regression and classification, $f(\mathbf{x})$ is directly regarded as a stochastic function; in particular, $f(\mathbf{x})$ is often modeling as a Gaussian process. There has been much discussion of the relationships between RKHS-based methods and GP-based methods (see, e.g., Rasmussen and Williams, 2006; Pillai et al., 2007). In this section we further investigate this relationship and then propose an effective and efficient GP-based classification method.

4.1 Gaussian Process Priors

The following proposition summarizes the connection between the Gaussian process and the feature basis expansion $\sum_{k=1}^{r} b_k \psi_k(\mathbf{x})$ given in (4).

Proposition 1 *Given a Gaussian process $\zeta(\mathbf{x})$ over $\mathcal{X}$, with zero mean and covariance function $g^{-1}K(\cdot,\cdot)$, where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a Mercer reproducing kernel, there exists a vector-valued function $\psi(\mathbf{x}) = (\psi_1(\mathbf{x}), \ldots, \psi_r(\mathbf{x}))'$ from $\mathcal{X}$ to $\mathbb{R}^r$ (r is possibly infinite) such that $K(\mathbf{x}_i, \mathbf{x}_j) = \psi(\mathbf{x}_i)'\psi(\mathbf{x}_j)$ for $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$ and*

$$\zeta(\mathbf{x}) = \sum_{k=1}^{r} b_k \psi_k(\mathbf{x}) \;\; \textit{with} \;\; b_k \overset{i.i.d}{\sim} N(0,\, g^{-1}). \tag{12}$$

Conversely, given a function $\zeta : \mathcal{X} \to \mathbb{R}$ in (12), then ζ is a Gaussian process with zero mean and covariance function $g^{-1}K(\mathbf{x}_i, \mathbf{x}_j)$ where $K(\mathbf{x}_i, \mathbf{x}_j) = \psi(\mathbf{x}_i)'\psi(\mathbf{x}_j)$.

If the feature expansion in (12) is regarded as a stochastic process, it is known as the Karhunen-Loève expansion. Proposition 1 provides a direct connection between GKMs and GP classifiers (GPCs) (Neal, 1999; Girolami and Rogers, 2006), and between GKMs and model-based geostatistics (Diggle et al., 1998). We see that $\mathbf{b} = (b_1, \ldots, b_r)'$ behaves as a regression vector in GKMs, whereas it plays the role of a latent vector in GPCs. Consequently, the feature function $\psi(\mathbf{x})$ defines the fixed-effect part of GKMs and the random-effect part of GPCs. In parallel with the fact that GKMs are GLMs in the feature space induced by the reproducing kernel K, we see that GPCs are generalized linear mixed models (Harville, 1977) in the feature space.

As discussed in Section 2, the Karhunen-Loève expansion can also be approximated by a finite-dimensional expansion over the training data set; that is,

$$\zeta(\mathbf{x}) = \sum_{i=1}^{n} \beta_i K(\mathbf{x}, \mathbf{x}_i) \text{ with } \beta = (\beta_1, \ldots, \beta_n)' \sim N_n(\mathbf{0},\ g^{-1}\mathbf{K}^{-1}).$$

Let $\zeta = (\zeta(\mathbf{x}_1), \ldots, \zeta(\mathbf{x}_n))'$ be the vector of n realizations of ζ over the training data. We then have $\zeta = \mathbf{K}\beta \sim N_n(\mathbf{0}, g^{-1}\mathbf{K})$. In our sparse treatment, some of the β_i are set to zero and the subvector β_γ of the nonzero elements is modeled as $N_{n_\gamma}(\mathbf{0}, g^{-1}\mathbf{K}_{\gamma\gamma}^{-1})$. In this case, $\zeta = \mathbf{K}_\gamma\beta_\gamma$ follows a singular normal distribution, that is, $\zeta \sim N_n(\mathbf{0}, g^{-1}\mathbf{K}_\gamma\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma')$. This sparse technique is called the "subset of regressors" (Rasmussen and Williams, 2006). In the following sections we investigate this sparsity-inducing approach to GP-based classification as an alternative to the FBGKM introduced in Section 3.

4.2 The MCMC Algorithm

By analogy with on the hierarchical model for our FBGKM in Section 3.1, we model the auxiliary variable s_i as

$$s_i = s(\mathbf{x}_i) = u + \zeta(\mathbf{x}_i) + \varepsilon_i \text{ with } \varepsilon_i \sim N(0,\ 1)$$

and keep other settings unchanged. Here $\zeta(\mathbf{x})$ is the Gaussian process with $\mathrm{E}(\zeta(\mathbf{x})) = 0$ and $\mathrm{Cov}(\zeta(\mathbf{x}_i), \zeta(\mathbf{x}_j)) = g^{-1}K(\mathbf{x}_i, \mathbf{x}_j)$. Applying $\zeta(\mathbf{x})$ to the training data, we have

$$\mathbf{s} = u\mathbf{1}_n + \zeta + \varepsilon \text{ with } \varepsilon \sim N_n(\mathbf{0},\ \mathbf{I}_n).$$

The inverses of $n \times n$ matrices are also required during Bayesian inference and prediction for GPCs. In order to reduce the computational costs, we use $\mathbf{K}_\gamma\beta_\gamma$ with $\beta_\gamma \sim N_{n_\gamma}(\mathbf{0}, g^{-1}\mathbf{K}_{\gamma\gamma}^{-1})$ to approximate ζ as in Section 4.1. This yields a *sparse GPC* (SGPC) model. The MCMC algorithm for SGPC is immediately obtained from that for FBGKM by simply removing the update of β_γ in Section 3.2, because β_γ is now the latent vector and it is not used for prediction. In particular, GPCs use the expectation of y_* w.r.t. $p(y_*|\mathbf{x}_*, \mathbf{y})$ as the predictor. We thus need to insert a step, which is to sample $s_* = s(\mathbf{x}_*)$ from $p(s_*|\mathbf{s}, \gamma, u, \mathbf{y})$, into the MCMC algorithm for prediction. This step is only necessary at every Mth sweep after the burn-in of the MCMC algorithm (see Diggle et al., 1998).

Now the marginal distribution of $\mathbf{s}$ is $[\mathbf{s}|u, \gamma] \sim N_n(u\mathbf{1}_n,\ \mathbf{M}_\gamma)$, where $\mathbf{M}_\gamma = \mathbf{I}_n + g^{-1}\mathbf{K}_\gamma\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma'$. Since s_* is conditionally independent of $\mathbf{y}$, given $\mathbf{s}$, we have

$$p(s_*|\mathbf{s}, \gamma, u) = N\big(u + g^{-1}\mathbf{k}_\gamma(\mathbf{x}_*)'\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma'\mathbf{M}_\gamma^{-1}(\mathbf{s} - u\mathbf{1}_n),\ v\big),$$

where $v = g^{-1}\mathbf{k}_\gamma(\mathbf{x}_*)'\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{k}_\gamma(\mathbf{x}_*) + 1 - g^{-2}\mathbf{k}_\gamma(\mathbf{x}_*)'\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma'\mathbf{M}_\gamma^{-1}\mathbf{K}_\gamma\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{k}_\gamma(\mathbf{x}_*)$ and $\mathbf{k}_\gamma(\mathbf{x}_*)$ is the subvector of $(K(\mathbf{x}_*, \mathbf{x}_1), \ldots, K(\mathbf{x}_*, \mathbf{x}_n))'$ corresponding to $\gamma_i = 1$. Since we have $\mathsf{E}(y_*|s_*) = \tau(s_*)$ and the used probit link τ is a monotonically increasing function on $(-\infty,\ \infty)$, we allocate $y_* = 1$ if $s_* > 0$ and $y_* = 0$ otherwise.

Let $I = \{\mathbf{x}_i : \gamma_i = 1, i = 1, \ldots, n\}$ be the set of active vectors. If the kernel function is stationary, then v is near zero and the posterior predictive mean of s_* reverts to u when $\mathbf{x}_*$ is far from points in the set I. Thus the sparse technique will give poor predictions, especially underestimates of the predictive variance. However, this problem is mitigated in our SGPC methodology since it uses Bayesian model averaging for prediction. That is, the prediction is based on the average over T active sets. In the following experiments the average is taken on $1,000$ active sets.

It is again worth noting that the reversible jump MCMC algorithm devised in this paper deals with parameter estimation and posterior prediction jointly in a single paradigm. Moreover, the reversible jump methodology is a sequential approach to model selection and posterior prediction. The main computational burden of the MCMC algorithm comes from the sampling procedure for parameter estimation, and the MCMC algorithm does not require extra computational complexity for the selection of active vectors and the prediction of responses.

The MCMC algorithms used in Neal (1999) and Diggle et al. (1998) is less efficient than ours because they do not use data augmentation or exploit sparsity. Girolami and Rogers (2006) proposed a Bayesian multinomial probit regression model, using variational methods for inference. For additional discussion of sparse approaches to GPs, interested readers should refer to Quiñonero-Candela and Rasmussen (2005), Rasmussen and Williams (2006) and Snelson (2007) and references therein.

5. Experimental Evaluations

In this section we conduct several experiments to evaluate the performance of our proposed Bayesian classification methods: FBGKM and SGPC. We compare the methods with various closely related Bayesian and non-Bayesian classification methods, including the Bayesian SVM (BSVM) (Mallick et al., 2005), the complete SVM (CSVM) (Mallick et al., 2005), sparse Gaussian processes (SGP+FIC) (Snelson and Ghahramani, 2006), and the conventional IVM and SVM.

We also implement our Bayesian GKM without Step (c) of the MCMC algorithm in Section 3.2. That is, we implement an MCMC algorithm that consists of Steps (a)-(b) by fixing $n_\gamma = n$. We denote the resulting model by BGKM to distinguish it from FBGKM. We could also implement a full (non-sparse) GPC, but since such a full GPC would have almost the same computational complexity as the GBKM, we do not implement the non-sparse GPC. All experiments have been implemented in Matlab on a Pentium 4 with a 2.80GHz CPU and 2.00GB of RAM.

5.1 Setup

We perform the experiments on several benchmark data sets: `BCI`, `g241d`, `Digit1`, $\texttt{COIL}_2$, `USPS` digits {(`0 vs. 1`), (`0 vs. 9`)}, `Letters` {(`A vs. B`), (`A vs. C`)}, `NewsGroups corpora`, $\texttt{Adult}_1$, $\texttt{Adult}_2$, `Mushrooms`, `Splice`, `Astroparticle`, `Ringnorm`, `Thyroid`, `Twonorm`, and `Waveform`. We first present a brief review of these data sets.

The `BCI` data set contains data obtained from project in brain-computer interfaces in which a single subject performs 400 trials in which he imagines movements with either the left or right hand. The `g241d` data set is an artificial data set which is generated by two unit-variance isotropic Gaussians with potentially misleading cluster structure. The `Digit1` data set is generated by applying a sequence of transformations to digit images, leading to a low-dimensional manifold geometrical structure embedded into a high-dimensional space. The $\texttt{COIL}_2$ data set is derived from the Columbia object image library (COIL-100) under a sequence of transformations, for example, rescaling, adding noise, and masking dimensions. Note that the `BCI`, `g241d`, `Digit1` and $\texttt{COIL}_2$ data sets are available at `http://www.kyb.tuebingen.mpg.de/ssl-book/`.

The USPS database is a handwritten digits data set which contains the digits from 0 to 9 automatically scanned from envelopes by the U.S. Postal Service. In our experiments, two digit pairs {(`0 vs. 1`), (`0 vs. 9`)} data sets are randomly constituted from the USPS database, and the dimensionality of each digit image and the number of digits in each digit class of each data set are 256 and 1000, respectively. The `Letters` data set consists of images of the 26 capital letters from

"A" to "Z," and two letter pairs {(A vs. B), (A vs. C)} are randomly constituted from "A," "B" and "C" with 789, 766, and 736 cases, respectively.

The 20 NewsGroups data set is organized into 20 different newsgroups, each corresponding to a different topic, and we randomly select the alt.atheism and comp.graphics topics for the binary classification problem. The total vocabulary size is 1390. Based on the information gain, 893 features are employed.

The Adult data set is originally extracted from the 1994 Census database with 14 features, of which six features are continuous and eight are categorical. Further, the Adult data set is processed with dimensionality of 123, that is, each continuous feature is discretized into a binary feature and each categorical feature with q categories is converted to q binary features. Here, the $\texttt{Adult}_1$ and $\texttt{Adult}_2$ data sets are constituted according to different training and test sizes.

The Mushrooms data set is originally drawn from the Audubon Society Field Guide to North American Mushrooms with 22 features. Similar to the Adult data set, the Mushrooms data set is processed into the binary feature representations, leading to 123 dimensions for each instance. The Splice data set is based on the biological process whereby intronic DNA is removed during protein translation. The Astrop (Astroparticle) data set is obtained from Jan Conrad of Uppsala University, Sweden. The Adult, Mushrooms, Splice, and Astrop data sets are available at http://www.csie.ntu.edu.tw/~cjlin.

The Ringnorm data set is artificially generated from two multivariate Gaussian distributions for the binary classification problem. That is, the instances within each class are obtained from a 20-variate Gaussian distribution. The Thyroid is collected from several databases of thyroid disease records. We use this data set to conduct a binary classification experiment in which the class euthyroidism is considered as the normal class and the classes hypothyroidism and hyperthyroidism are considered as an abnormal class.

The Twonorm data set is also an artificial 20-dimensional two-class classification example, which consists of 7400 instances. The Waveform data set is generated from a combination of 2 of 3 "base" waves in a 21-dimensional space. The Ringnorm, Thyroid, Twonorm, and Waveform data sets are widely used for the classification benchmarking, and they are available at http://ida.first.gmd.de/~raetsch/data/benchmarks.htm.

Table 1 gives a summary of these data sets. In our experiments, each data set is randomly partitioned into two disjoint subsets as the training and test. Twenty random partitions are generated for each data set. Based on these partitions, several evaluation criteria, including the average classification error rate, standard deviation and average computational time, are reported.

All of the methods that we implement are based on a Gaussian RBF kernel with a single width parameter; that is, $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\theta^2\right)$. In Section 5.3 we present experiments in which this hyperparameter is estimated from data based on the ideas discussed in Section 3.2. In the remaining sections, however, we use a simpler procedure in which the value of θ is set to the mean Euclidean distance between training data points. We found this setting to be effective empirically in our applications. The gain in computational complexity is significant, particularly for the full GP methods, BSVM and CSVM, whose calculations involve two full kernel matrices. In particular, for each new value of θ, it is necessary to recalculate the kernel matrix $\mathbf{K}$ for each sweep of the MCMC algorithms.

In addition, we set the hyperparameters in both FBGKM and SGPC as follows: $a_\eta = 1$, $b_\eta = 0.1$, $a_g = 4$ and $b_g = 0.1$. For all of the Bayesian classification methods, we run each MCMC algorithm for $10,000$ sweeps, discard the first $5,000$ as the burn-in, and retain every 5th realization

Data Set	n	m	p	$k_{\max}$
BCI	300	100	117	100
g241d	300	1200	241	200
Digit1	300	1200	241	200
COIL_2	300	1200	241	200
USPS (0 vs.1)	500	1500	256	200
USPS (0 vs.9)	500	1500	256	200
Letters (A vs.B)	300	1255	16	100
Letters (A vs.C)	300	1225	16	100
NewsGroups	500	1485	893	200
Splice	2000	1175	60	200
Astrop(article)	4000	3089	4	200
Mushrooms	4000	4124	112	200
Adult_1	6000	10000	123	200
Adult_2	20000	12500	123	200
Ringnorm	400	7000	20	400
Thyroid	140	75	5	140
Twonorm	400	7000	20	400
Waveform	400	4600	21	400

Table 1: Summary of the Benchmark Data Sets: n—the size of the training data set; m—the size of the test data set; p—the dimension of the input vector; $k_{\max}$—the maximum number of active vectors.

of parameters after the burn-in for inference and prediction. These settings are empirically validated to be sufficient for these methods to achieve convergence. Recall that the test is implemented after the burn-in of the MCMC sampling. This implies that the Bayesian model averaging component of our Bayesian methods uses $1,000$ ($T = (10,000-5,000)/5$) active sets for test.

5.2 Evaluation 1

In the first evaluation, we compare BGKM, FBGKM and SGPC with BSVM and CSVM, because they are the two existing Bayesian kernel methods most closely related to our Bayesian classification methods.

We conduct this evaluation on the first nine data sets in Table 1, randomly partitioning the data into disjoint training and test data sets according to the corresponding settings of n and m. All the inputs are normalized to have zero mean and unit variance. Tables 2 and 3 report the performance of the five Bayesian methods on the nine different data sets in terms of the average classification error rate (%), the standard deviation and the corresponding average computational time (s).

From Tables 2 and 3, we can see that our FBGKM, SGPC and BGKM methods based on the Silverman g-prior achieve slightly lower classification error rates than the BSVM and CSVM methods on the whole. Moreover, our methods have roughly similar classification error rates on the nine data sets. In addition, FBGKM and SGPC are more efficient computationally than BGKM the other methods; this is due to their exploitation of sparsity.

Data Set	BSVM		CSVM		BGKM		SGPC		FBGKM	
	err	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)
BCI	28.15	($\pm$2.15)	29.40	($\pm$2.58)	29.35	($\pm$2.82)	27.10	($\pm$1.85)	29.83	($\pm$2.36)
g241d	17.15	($\pm$1.68)	17.63	($\pm$1.15)	16.37	($\pm$1.11)	16.55	($\pm$1.22)	16.30	($\pm$0.89)
Digit1	4.86	($\pm$0.74)	4.88	($\pm$0.75)	4.87	($\pm$0.65)	5.51	($\pm$0.66)	4.85	($\pm$0.67)
COIL$_2$	9.71	($\pm$0.81)	9.86	($\pm$0.71)	9.16	($\pm$0.99)	9.83	($\pm$0.97)	9.797	($\pm$0.32)
USPS(0 vs. 1)	0.40	($\pm$0.30)	0.35	($\pm$0.11)	0.28	($\pm$0.05)	0.31	($\pm$0.14)	0.28	($\pm$0.06)
USPS(0 vs. 9)	1.36	($\pm$0.36)	1.40	($\pm$0.29)	1.36	($\pm$0.28)	1.21	($\pm$0.19)	1.37	($\pm$0.24)
Letters(A vs. B)	0.92	($\pm$0.59)	0.95	($\pm$0.45)	0.75	($\pm$0.24)	0.53	($\pm$0.19)	0.77	($\pm$0.24)
Letters(A vs. C)	0.83	($\pm$0.15)	0.93	($\pm$0.27)	0.87	($\pm$0.15)	0.65	($\pm$0.20)	0.84	($\pm$0.15)
NewsGroups	5.62	($\pm$0.80)	5.08	($\pm$0.33)	4.92	($\pm$0.28)	4.66	($\pm$0.38)	4.83	($\pm$0.25)

Table 2: Experimental results for the five methods on different data sets: *err*$-$ the test error rates (%); *std*$-$ the corresponding standard deviation.

Data Set	BSVM	CSVM	BGKM	SGPC	FBGKM
BCI	2.615×10^3	2.596×10^3	1.063×10^3	0.756×10^3	0.688×10^3
g241d	4.339×10^3	4.365×10^3	1.819×10^3	1.227×10^3	1.451×10^3
Digit1	5.248×10^3	5.210×10^3	2.459×10^3	1.738×10^3	2.011×10^3
COIL$_2$	4.988×10^3	4.996×10^3	2.454×10^3	1.357×10^3	1.502×10^3
USPS(0 vs. 1)	2.133×10^4	2.047×10^4	6.013×10^3	2.464×10^3	2.700×10^3
USPS(0 vs. 9)	2.239×10^4	2.230×10^4	6.479×10^3	2.868×10^3	2.974×10^3
Letters(A vs. B)	2.009×10^3	2.007×10^3	0.914×10^3	0.568×10^3	0.593×10^3
Letters(A vs. C)	2.026×10^3	2.042×10^3	0.896×10^3	0.604×10^3	0.596×10^3
NewsGroups	2.286×10^4	2.291×10^4	6.270×10^3	2.675×10^3	2.910×10^3

Table 3: The computational times (*s*) for the five methods on different data sets.

In the following experiments, we attempt to analyze the performance of the methods with respect to different values of the training size n and the maximum number k_{max} of active vectors. For the sake of simplicity, we only report results on the NewsGroups data set.

Tables 4 and 5 show the experimental results when changing the training size n and fixing the maximum number of active vectors to $k_{max} = 200$. As can be seen, all the five methods obtain a lower classification error rate and have greater computational costs as the training size n increases. Furthermore, FBGKM, SGPC and BGKM slightly outperform BSVM and CSVM in both classification error rate and computational cost. The FBGKM and SGPC methods are relatively more efficient for the data sets of large training size n.

Table 6 shows the experimental results for our FBGKM and SGPC methods with respect to different values of the maximum number k_{max} of active vectors and for a fixed training size of $n = 800$. The performance of these two methods is roughly similar for each setting k_{max}; that is, they are insensitive to k_{max}. However, their computational costs tend to slightly increase as the maximum number k_{max} of active vectors increases.

Additionally, in order to study the MCMC mixing performance of our FBGKM and SGPC methods we report the numbers of active vectors over different data sets. In particular, Figure 2

Training size n	BSVM	CSVM	BGKM	SGPC	FBGKM
	err $(\pm std)$	err $(\pm std)$	err $(\pm std)$	err $(\pm std)$	err $(\pm std)$
n=300	5.99 (±1.44)	5.84 (±0.80)	5.37 (±0.52)	5.34 (±0.40)	5.08 (±0.49)
n=400	5.65 (±0.98)	5.83 (±0.93)	5.10 (±0.35)	5.03 (±0.55)	5.05 (±0.39)
n=500	5.62 (±0.80)	5.08 (±0.33)	4.92 (±0.28)	4.66 (±0.38)	4.83 (±0.25)
n=600	5.77 (±0.61)	5.13 (±0.20)	4.92 (±0.43)	4.35 (±0.47)	4.74 (±0.28)
n=700	5.63 (±0.82)	4.82 (±0.21)	4.44 (±0.36)	4.12 (±0.22)	4.61 (±0.52)
n=800	5.14 (±0.59)	5.10 (±0.16)	4.49 (±0.47)	4.13 (±0.51)	4.56 (±0.34)

Table 4: Experimental results for the five methods corresponding to different training sizes n on the `NewsGroups` data set with $k_{max} = 200$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

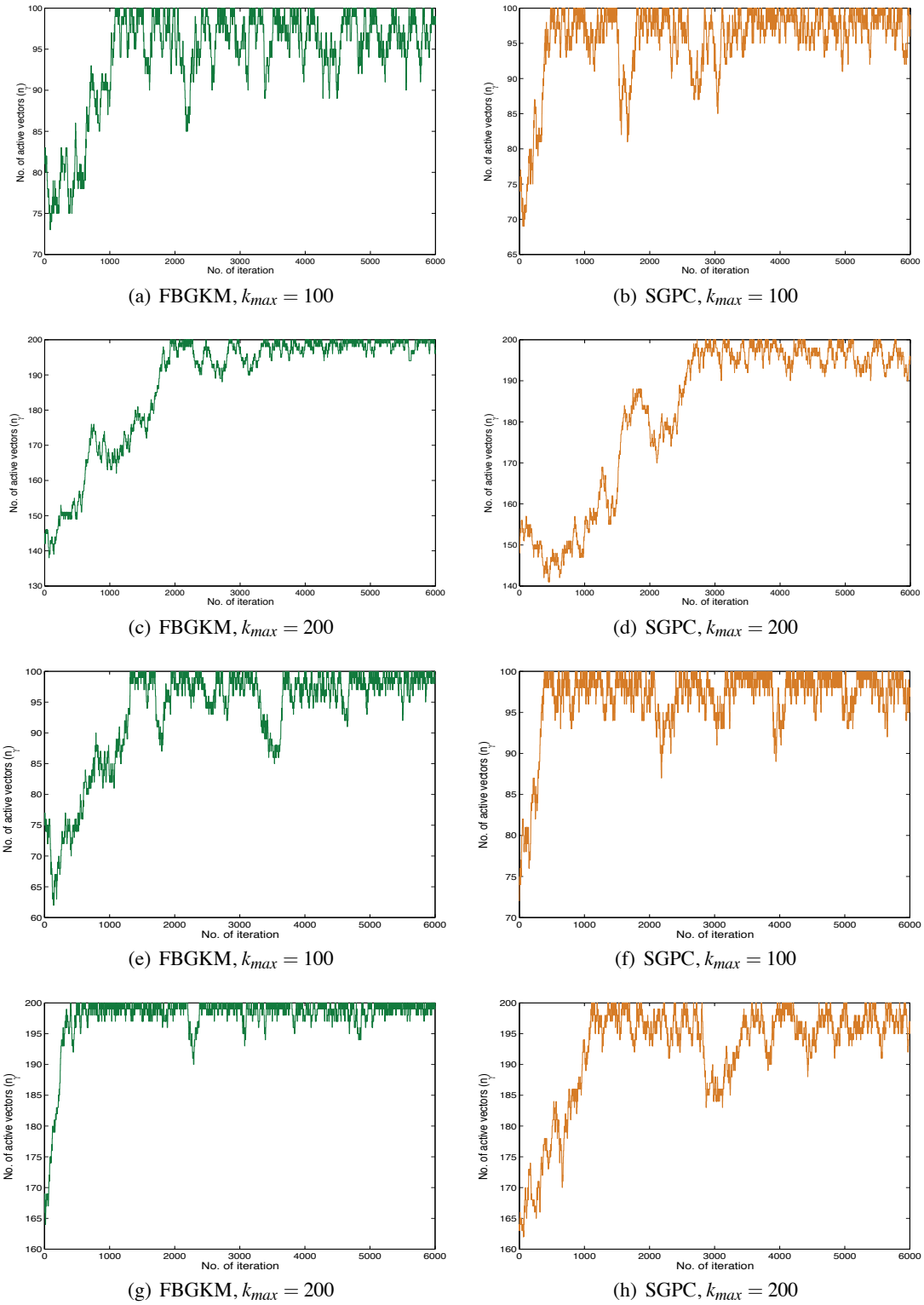
Training size n	BSVM	CSVM	BGKM	SGPC	FBGKM
n=300	5.949×10^3	5.830×10^3	2.467×10^3	1.862×10^3	2.085×10^3
n=400	1.173×10^4	1.171×10^4	4.674×10^3	2.555×10^3	2.804×10^3
n=500	2.286×10^4	2.291×10^4	6.270×10^3	2.675×10^3	2.910×10^3
n=600	3.458×10^4	3.461×10^4	8.340×10^3	2.748×10^3	2.973×10^3
n=700	5.195×10^4	5.186×10^4	1.207×10^4	3.279×10^3	3.610×10^3
n=800	7.754×10^4	7.757×10^4	1.673×10^4	3.885×10^3	4.327×10^3

Table 5: The computational times (s) for the five methods corresponding to different training sizes n on the `NewsGroups` data set with $k_{max} = 200$.

k_{max} of active vectors	FBGKM		SGPC	
	err $(\pm std)$	$time$	err $(\pm std)$	$time$
$k_{max} = 300$	4.55 (±0.46)	6.522×10^3	4.27 (±0.47)	5.592×10^3
$k_{max} = 400$	4.62 (±0.45)	7.189×10^3	4.80 (±0.49)	6.808×10^3
$k_{max} = 500$	4.64 (±0.37)	8.536×10^3	4.75 (±0.38)	7.704×10^3
$k_{max} = 600$	4.75 (±0.48)	1.033×10^4	4.57 (±0.43)	9.469×10^3
$k_{max} = 700$	4.72 (±0.28)	1.170×10^4	4.74 (±0.31)	1.057×10^4

Table 6: Experimental results for FBGKM and SGPC corresponding to different maximum numbers k_{max} of active vectors on the `NewsGroups` data set with $n = 800$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation; $time-$ the corresponding computational time (s).

depicts the output of the numbers n_γ of active vectors corresponding to the first 6000 sweeps in the MCMC inference procedure on `BCI`, `Digit1`, `Letters` {`(A vs.B)`} and `NewsGroups`. The results in Figure 2 clearly show that the FBGKM and SGPC methods mix rapidly in these experiments, yielding reliable estimates after the first 3000 sweeps.



(a) FBGKM, $k_{max} = 100$ (b) SGPC, $k_{max} = 100$

(c) FBGKM, $k_{max} = 200$ (d) SGPC, $k_{max} = 200$

(e) FBGKM, $k_{max} = 100$ (f) SGPC, $k_{max} = 100$

(g) FBGKM, $k_{max} = 200$ (h) SGPC, $k_{max} = 200$

Figure 2: MCMC Output for the numbers n_γ of active vectors of our FBGKM and SGPC methods on the four data sets: (a/b) `BCI`; (c/d) `Digit1`; (e/f) `Letters (A vs. B)`; (g/h) `NewsGroups`.

In probit-type models, since the posterior distribution of each s_i is truncated normal, we are able to update the s_i using the Gibbs sampler. Recall that we employ an efficient auxiliary variable approach proposed by Holmes and Held (2006) for the implementation of the Gibbs sampler (see Appendix A). For the other models, however, a MH sampler is required to update the s_i. This makes the corresponding MCMC algorithms take longer to mix. Thus, our models, which are based on the probit link, can be expected to be more efficient computationally than the BSVM and CSVM. However, to standardize the experimental comparison, we use the same setup for MCMC sweeps and burn-in for all algorithms.

Table 7 describes distributions of active vectors to appear after the burn-in (in the last 5,000 sweeps). As we can see, the number n_γ of active vectors jumps between a small range for different data sets, due to the rapid mixing. The maximum frequency of active vectors corresponding to the number n_γ of active vectors is also given in Table 7.

Data Set	FBGKM			SGPC		
	Max	Min	Most	Max	Min	Most
BCI	100	85	99 (918)	100	87	99 (1145)
g241d	200	184	199 (1263)	200	153	196 (339)
Digit1	200	187	199 (1163)	200	184	188 (515)
COIL_2	151	138	146 (1493)	149	139	147 (2870)
USPS(0 vs. 1)	170	151	165 (1080)	151	132	140 (1288)
USPS(0 vs. 9)	200	172	195 (1258)	200	177	199 (1095)
Letters(A vs. B)	100	80	99 (796)	100	89	99 (1422)
Letters(A vs. C)	100	80	97 (551)	100	84	99 (1006)
NewsGroups	200	186	199 (918)	200	182	199 (720)

Table 7: Distributions of active vectors after the burn-in under FBGKM and SGPC. Max—the maximum number of active vectors to appear; Min—the minimum number of active vectors to appear; Most—the number of active vectors with the maximum frequency and the corresponding frequency shown in brackets.

5.3 Evaluation 2

We further evaluate the performance of our sparse Bayesian kernel methods under kernel parameter learning, and compare the FBGKM and SGPC with SGP+FIC and full GP (FGP) (Rasmussen and Williams, 2006). In particular, we use the Gaussian RBF kernel with multiple parameters, that is, $K(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(-\sum_{l=1}^{p}(x_{il}-x_{jl})^2/\theta_l^2\right)$, and estimate those parameters $\theta = (\theta_1,\ldots,\theta_p)$ in all compared Bayesian kernel methods. In order to distinguish from the Bayesian methods with the fixed kernel parameters, we label the Bayesian methods with the learned parameters via "$\star$+KL." We conduct experimental analysis on the Adult, Mushrooms, Splice, and Astroparticle data sets.

Since for FGP+KL learning the kernel parameters results in a huge computational cost, we set the sizes of the training and test data as 1000 (i.e., $n = m = 1000$) in each data set. In this setting, there is no distinction between Adult_1 and Adult_2, so we just use Adult to denote the corresponding data set. Also, since it is infeasible to use MCMC inference for FGP+KL, we employ the

expectation propagation (EP) algorithm (Minka, 2001) for FGP+KL. However, to provide an apples-to-apples comparison with our sparse Bayesian kernel methods, we employ MCMC inference for SGP+FIC+KL. For the sparse methods compared here, we fix the size of active set to 100, that is, $k_{max} = 100$. Our implementations for SGP+FIC and FGP are based on the Matlab codes from `http://www.lce.hut.fi/research/mm/gpstuff/` and `http://www.gaussianprocess.org/gpml/`, respectively.

Tables 8 and 9 and Figure 3 report the performance of the SGP+FIC+KL, FGP+KL, FBGKM+KL, and SGPC+KL methods on the four data sets in terms of the average classification error rate (%), the standard deviation and the corresponding average computational time (s). Figure 3 depicts the logarithm scale of the corresponding average computational time (s) on the different data sets. It is clear that FBGKM+KL and SGPC+KL outperform other methods on the whole. Additionally, the computational times of all compared methods tend to increase when the number p of the kernel parameters increases. We note that the computational times of SGP+FIC+KL and FGP+KL would become huge if we directly applied them to the large data sets listed in Table 1—$\texttt{Adult}_1$, $\texttt{Adult}_2$, `Mushrooms`, `Splice`, and `Astroparticle`.

Data Set	SGP+FIC+KL		FGP+KL		FBGKM+KL		SGPC+KL	
	err	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$
`Splice`	18.05	($\pm$0.77)	9.19	($\pm$1.25)	7.49	($\pm$0.14)	11.54	($\pm$0.51)
`Astrop`	4.10	($\pm$0.36)	4.57	($\pm$0.41)	3.45	($\pm$0.31)	3.52	($\pm$0.31)
`Mushrooms`	1.70	($\pm$0.27)	0.20	($\pm$0.20)	0.24	($\pm$0.18)	0.45	($\pm$0.30)
`Adult`	18.50	($\pm$0.56)	17.85	($\pm$0.35)	15.94	($\pm$0.45)	15.56	($\pm$0.43)

Table 8: Experimental results for the four Bayesian kernel methods on the four data sets with learned kernel parameters θ, $k_{max} = 100$, $n = 1000$, and $m = 1000$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

Data Set	SGP+FIC+KL	FGP+KL	FBGKM+KL	SGPC+KL
`Splice`	2.542×10^5	1.228×10^5	1.132×10^4	1.121×10^4
`Astrop`	4.081×10^4	2.531×10^4	7.431×10^3	7.103×10^3
`Mushrooms`	4.639×10^5	1.583×10^5	1.551×10^4	1.534×10^4
`Adult`	4.713×10^5	1.605×10^5	1.606×10^4	1.557×10^4

Table 9: The computational times (s) of the four Bayesian kernel methods on the four data sets with learned kernel parameters θ, $k_{max} = 100$, $n = 1000$, and $m = 1000$.

In order to further evaluate the performance of our sparse Bayesian kernel methods on some larger data sets, we also conduct comparative experiments of FBGKM, SGPC, and SGP+FIC (Snelson and Ghahramani, 2006) on the $\texttt{Adult}_1$, $\texttt{Adult}_2$, `Mushrooms`, `Splice`, and `Astroparticle` data sets. Here, we consider both MCMC and EP inference methods for SGP+FIC to provide a fuller comparison, referring to them as SGP+FIC+MCMC and SGP+FIC+EP, respectively. For these sparse methods, we fix the size of active set k_{max} to 200.

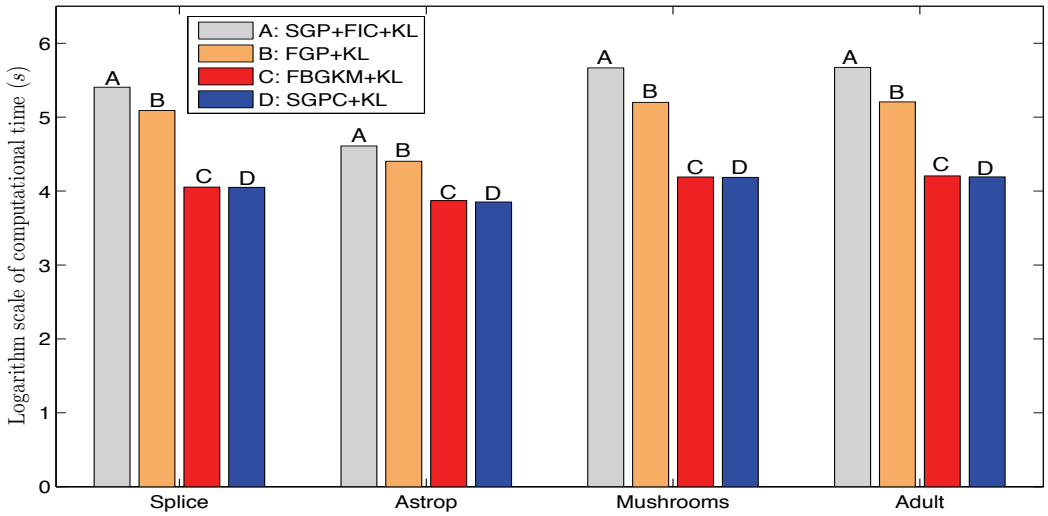


Figure 3: The computational times (s) for the four Bayesian kernel methods on the four data sets with learned kernel parameter θ, $k_{max} = 100$, $n = 1000$, and $m = 1000$.

Table 10 reports the classification performance of the SGP+FIC+MCMC, SGP+FIC+EP, FBGKM and SGPC methods on the five data sets. It should be pointed out here that we do not report the corresponding results of SGP+FIC+MCMC on the $\texttt{Adult}_2$ data set due to the huge computational times of performing it on this data set. From Table 10, we can see that our FBGKM and SGPC methods outperform other methods on the whole. Furthermore, it is still difficult for SGP+FIC+MCMC and SGP+FIC+EP to calculate the optimal solution for sparse approximation of full Gaussian process, due to the sensitivity of the performance to the initial active set.

Data Set	SGP+FIC+EP		SGP+FIC+MCMC		FBGKM		SGPC		SVM	
	err	$(\pm std)$	err	$(\pm std)$	err	$(\pm std)$	err	$(\pm std)$	err	$(\pm std)$
Splice	14.38	(±1.10)	16.32	(±0.15)	12.53	(±0.57)	12.07	(±0.45)	13.01	(±0.69)
Astrop	5.20	(±0.26)	3.38	(±0.11)	3.59	(±0.18)	3.34	(±0.16)	3.37	(±0.14)
Mushrooms	1.55	(±0.21)	1.38	(±0.13)	0.19	(±0.08)	0.21	(±0.06)	0.55	(±0.37)
$\texttt{Adult}_1$	15.89	(±0.38)	15.79	(±0.26)	15.24	(±0.21)	15.59	(±0.16)	16.64	(±0.33)
$\texttt{Adult}_2$	15.49	(±0.21)	–	–	15.01	(±0.17)	15.26	(±0.19)	16.27	(±0.28)

Table 10: Experimental results for the five methods on the Splice, Astroparticle, Mushrooms, $\texttt{Adult}_1$, and $\texttt{Adult}_2$ data sets with $k_{max} = 200$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

Table 11 and Figure 4 report the average computational times of the compared sparse Bayesian kernel methods on the five data sets, with Figure 4 depicting the computational times on a logarithm scale. Table 11 and Figure 4 show that the SGP+FIC+MCMC has larger computational times than

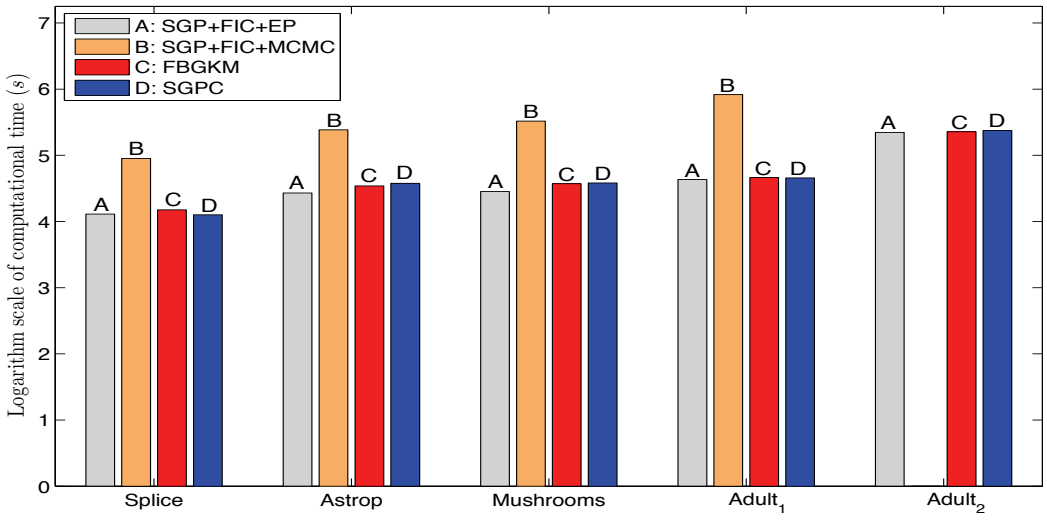


Figure 4: The computational times (s) of the four sparse Bayesian kernel methods on the `Splice`, `Astroparticle`, `Mushrooms`, $\texttt{Adult}_1$, and $\texttt{Adult}_2$ data sets with $k_{max} = 200$.

other methods on the five different data sets, and that the computational times of SGP+FIC+EP, FBGKM and SGPC are very close to each other.

Data Set	SGP+FIC+EP	SGP+FIC+MCMC	FBGKM	SGPC
`Splice`	1.293×10^4	8.952×10^4	1.497×10^4	1.262×10^4
`Astrop`	2.701×10^4	2.425×10^5	3.443×10^4	3.779×10^4
`Mushrooms`	2.830×10^4	3.290×10^5	3.726×10^4	3.814×10^4
$\texttt{Adult}_1$	4.298×10^4	8.283×10^5	4.620×10^4	4.548×10^4
$\texttt{Adult}_2$	2.221×10^5	–	2.276×10^5	2.369×10^5

Table 11: The computational times (s) of the four sparse Bayesian kernel methods on the `Splice`, `Astroparticle`, `Mushrooms`, $\texttt{Adult}_1$, and $\texttt{Adult}_2$ data sets with $k_{max} = 200$.

In addition, we conduct a quantitative assessment of convergence of the MCMC algorithms for the three sparse Bayesian kernel methods. We employ a method proposed by Brooks (1998). The method uses a cusum criterion with "hairiness" definition to monitor convergence. The length of chain for convergence is determined, once the sequence on the "hairiness" definition statistically lies within the 90% confidence intervals under the binomial distribution. Table 12 reports the convergence assessment results on the five data sets. From Table 12, we can see that all MCMC algorithms in the three sparse Bayesian kernel methods can achieve convergence on the five data sets after the first 5000 sweeps. Moreover, the convergence time is similar for each method, while the corresponding computational times of SGP+FIC+MCMC are obviously largest on the five data sets.

Data Set	SGP+FIC+MCMC		FBGKM		SGPC	
	time	*burn-in*	*time*	*burn-in*	*time*	*burn-in*
Splice	4.017×10^4	4566	1.019×10^3	906	2.195×10^3	1898
Astrop	9.265×10^4	3874	7.016×10^3	2125	1.123×10^4	3218
Mushrooms	9.857×10^4	3192	1.362×10^4	3924	9.677×10^3	2616
Adult_1	1.865×10^5	2451	8.605×10^3	1906	1.206×10^4	2873
Adult_2	–	–	5.217×10^4	2424	8.142×10^4	3605

Table 12: Monitoring convergence of MCMC algorithms for the three sparse Bayesian kernel methods on the Splice, Astroparticle, Mushrooms, Adult_1, and Adult_2 data sets with $k_{max} = 200$: *time*– the computational time (*s*) for convergence; *burn-in*– the length of chain for convergence.

5.4 Bayesian vs. Non-Bayesian

Since FBGKM and SGPC are Bayesian alternatives to IVM and SVM, it is useful to compare our FBGKM and SGPC with the conventional IVM and SVM. We compared these methods on the following data sets: Ringnorm, Thyroid, Twonorm and Waveform. These data sets were also used by Zhu and Hastie (2005) and a detailed presentation of results can be found in Rätsch et al. (2001). Each data set is randomly partitioned into two disjoint subsets as training and test data sets according to the training and test sizes n and m given in Table 1. In addition, the maximum number k_{max} of active vectors is set according to Table 1. The results in Table 13 are based on the average of twenty realizations and the results with the conventional IVM and SVM are cited from Zhu and Hastie (2005). We also conduct a comparison of FBGKM and SGPC with the conventional SVM on the Splice, Astroparticle, Mushrooms, Adult_1, and Adult_2 data sets. The classification results are given in Table 10. From Tables 10 and 13 we can see that the Bayesian approaches slightly outperform the non-Bayesian approaches.

Data Set	SVM		IVM		FBGKM		SGPC	
	err	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)
Ringnorm	2.03	($\pm$0.19)	1.97	($\pm$0.29)	1.51	($\pm$0.10)	1.56	($\pm$0.12)
Thyroid	4.80	($\pm$2.98)	5.00	($\pm$3.02)	4.60	($\pm$2.65)	4.51	($\pm$2.32)
Twonorm	2.90	($\pm$0.25)	2.45	($\pm$0.15)	2.86	($\pm$0.21)	2.79	($\pm$0.23)
Waveform	9.98	($\pm$0.43)	10.13	($\pm$0.47)	9.80	($\pm$0.31)	9.73	($\pm$0.30)

Table 13: Experimental results for the four methods on the four data sets: *err*– the test error rates (%); *std*– the corresponding standard deviation.

6. Extensions

In this section we consider several extensions of the modeling framework that we have discussed thus far. One extension is immediate: We can obtain a fully Bayesian approach to model selection for the SVM by combining our work with the treatment of Mallick et al. (2005). That is, we form a

conditional likelihood from the hinge loss (also see Sollich, 2001) and assign a mixture of the point-mass distribution and the Silverman g-prior to the regression vector. In the following subsections we consider two additional extensions.

6.1 Multiple Kernel Learning

Kernel learning has emerged as an important theme in the machine learning community. We have provided a Bayesian foundation for kernel learning in Section 3. In particular, given a kernel function, we can estimate parameters of the kernel function. We now discuss how to extend this capability to the learning of combinations of kernels; the *multiple kernel learning problem* (Bach et al., 2004).

Assume that we are given q distinct kernel functions $K_l(\mathbf{x}_i, \mathbf{x}_j)$, for $l = 1, \ldots, q$. Correspondingly, we have q feature functions (say $\psi_l(\mathbf{x})$). In this case, the predictive function is expressed as

$$f(\mathbf{x}) = u + \sum_{l=1}^{q} \psi_l(\mathbf{x})' \mathbf{b}_l.$$

Letting $\mathbf{b}_l = g_l \Psi_l' \beta_l$ where $\Psi_l = [\psi_l(\mathbf{x}_1), \ldots, \psi_l(\mathbf{x}_n)]'$, $\beta_l = (\beta_{l1}, \ldots, \beta_{ln})'$ and $g_l \geq 0$, we have

$$f(\mathbf{x}) = u + \sum_{l=1}^{q} g_l \sum_{i=1}^{n} K_l(\mathbf{x}, \mathbf{x}_i) \beta_{li}.$$

Now we assign $\beta_l \sim N_n\big(\mathbf{0},\ \sigma^2 (\mathbf{K}^{(l)})^{-1}\big)$ and

$$g_l \sim \rho \delta_0(g_l) + (1-\rho) Ga(g_l | a_g/2,\ b_g/2),$$

where $\mathbf{K}^{(l)} = \Psi_l \Psi_l'$ ($n \times n$) is the lth kernel matrix, $\delta_0(\cdot)$ is a point-mass at zero and the user-specific parameter $\rho \in (0,1)$ controls the levels of the nonzero g_l. Thus, we only need to update the g_l instead of g in the Bayesian computation in Section 3. Note that kernel parameter learning and multiple kernel learning can be incorporated together.

6.2 Multi-class Learning

We consider the extension of our fully Bayesian modeling approach to a c-class ($c > 2$) classification problem where the class label $\mathbf{y}_i$ is a binary c-vector with values all zero except a one in position j if $\mathbf{x}_i$ belongs to the jth class. In this case, we define c regression vectors $\beta_j = (\beta_{1j}, \ldots, \beta nj)' \in \mathbf{R}^n$ and c auxiliary vectors $\mathbf{s}_j = (s_{1j}, \ldots, s_{nj})' \in \mathbf{R}^n$, $j = 1, \ldots, c$, for each class. We then have

$$\mathbf{s}_j = \mathbf{1}_n u_j + \mathbf{K} \beta_j + \mathbf{e}_j, \quad j = 1, \ldots, c,$$

where the $\mathbf{e}_j$ are i.i.d. from $N_n(\mathbf{0}, \mathbf{I}_n)$.

We now denote $\mathbf{u} = (u_1, \ldots, u_c)'$, $\mathbf{B} = [\beta_1, \ldots, \beta_c]$, $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_c]$ and $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_c]$. As in the binary problem, we also introduce a binary n-vector γ with either $\gamma_i = 1$ if $\mathbf{x}_i$ is an active vector or $\gamma_i = 0$ if $\mathbf{x}_i$ is not an active vector. Let $\mathbf{K}_\gamma'$ and $\mathbf{B}_\gamma$ be $\mathbf{K}'$ and $\mathbf{B}$ with the rows for which $\gamma_i = 0$ deleted. Thus, we can form the following sparse model:

$$\mathbf{S} = \mathbf{1}_n \mathbf{u}' + \mathbf{K}_\gamma \mathbf{B}_\gamma + \mathbf{E} = \widetilde{\mathbf{K}}_\gamma \widetilde{\mathbf{B}}_\gamma + \mathbf{E},$$

where $\widetilde{\mathbf{K}}_\gamma = [\mathbf{1}_n, \mathbf{K}_\gamma]$ and $\widetilde{\mathbf{B}}'_\gamma = [\mathbf{u}, \mathbf{B}'_\gamma]$. Now given γ, we treat the $\widetilde{\mathbf{B}}_\gamma$ as being independently from $N_{n_\gamma+1}(\mathbf{0}, \Sigma_\gamma^{-1})$.

To make the model identifiable, the constraint $\sum_{j=1}^c (u_j \mathbf{1}_n + \mathbf{K}\beta_j) = \mathbf{0}$ is typically required (see, e.g., Lee et al., 2004). Clearly, a sufficient condition for this constraint is that $\sum_{j=1}^c u_j = 0$ and $\sum_{j=1}^c \beta_j = \mathbf{0}$. To address this issue we impose the constraint $\sum_{j=1}^c \mathbf{s}_j = \mathbf{S}\mathbf{1}_c = \mathbf{0}$ and consider the following error model:

$$\mathbf{S} = \mathbf{1}_n \mathbf{u}'\mathbf{H} + \mathbf{K}_\gamma \mathbf{B}_\gamma \mathbf{H} + \mathbf{E}\mathbf{H} = \tilde{\mathbf{K}}_\gamma \tilde{\mathbf{B}}_\gamma \mathbf{H} + \mathbf{E}\mathbf{H}, \tag{13}$$

where $\mathbf{H} = \mathbf{I}_c - \frac{1}{c}\mathbf{1}_c\mathbf{1}'_c$ is the $c \times c$ centering matrix. Since $\tilde{\mathbf{B}}_\gamma \mathbf{H} \sim N_{n_\gamma+1,c}(\mathbf{0}, \Sigma_{\gamma\gamma}^{-1} \otimes \mathbf{H})$ and $\mathbf{E}\mathbf{H} \sim N_{n,c}(\mathbf{0}, \mathbf{I}_n \otimes \mathbf{H})$, we have $\mathbf{S} = \mathbf{S}\mathbf{H} \sim N_{n,c}(\mathbf{0}, \mathbf{Q}_\gamma \otimes \mathbf{H})$. Here we use the formal of matrix-variate normal distributions; that is, $\mathbf{Z} \sim N_{m,p}(\mathbf{0}, \mathbf{M} \otimes \mathbf{N})$ if and only if $\mathrm{vec}(\mathbf{Z}') \sim N_{mp}(\mathbf{0}, \mathbf{M} \otimes \mathbf{N})$ where $\mathbf{Z} = [z_{ij}]$ is an $m \times p$ matrix, $\mathrm{vec}(\mathbf{Z}') = (z_{11}, z_{12}, \ldots, z_{mp})'$ is its arrangement in a stack, and $\mathbf{M} \otimes \mathbf{N}$ represents the Kronecker product of $\mathbf{M}$ and $\mathbf{N}$. Note that both $N_{n,c}(\mathbf{0}, \mathbf{Q}_\gamma \otimes \mathbf{H})$ and $N_{n_\gamma+1,c}(\mathbf{0}, \Sigma_{\gamma\gamma}^{-1} \otimes \mathbf{H})$ are singular matrix-variate distributions, because $\mathbf{H}$ is singular. Please refer to Gupta and Nagar (2000) for matrix-variate normal distributions and singular matrix-variate normal distributions.

We rewrite (13) in vector form as

$$\mathrm{vec}(\mathbf{S}) = (\mathbf{H} \otimes \widetilde{\mathbf{K}}_\gamma)\mathrm{vec}(\widetilde{\mathbf{B}}_\gamma) + (\mathbf{H} \otimes \mathbf{I}_n)\mathrm{vec}(\mathbf{E}).$$

We can apply the MCMC algorithm in Section 3.2 to the multi-class case. The main difference is in Step (a) for the update of $\mathbf{S}$. That is, in the multi-class probit setting, the relationship between the class labels and the auxiliary vectors becomes

$$y_{ij} = \begin{cases} 1 & \text{if } j = \mathrm{argmax}_{1 \le k \le c}\{s_{ik}\}, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the posterior distribution of each s_{ij} is truncated normal, $[s_{ij} | \gamma, u_j, \beta_j, y_{ij}] \sim N(u_j + \mathbf{k}'_i \beta_j, 1)$ subject to $s_{ij} > s_{il}$ for all $l \neq j$ if $y_{ij} = 1$.

Finally, it is straightforward to develop a sparse GP method for multi-class classification problems. We should note that Girolami and Rogers (2006) proposed a Bayesian multinomial probit regression model and derived a fully variational Bayesian method for multi-class Gaussian process classification. Specifically, $\mathbf{S}$ and $\mathbf{KB}$ respectively correspond to latent and manifest Gaussian random matrices in Bayesian multinomial probit regression. However, they did not consider the constraint $\sum_{j=1}^c (u_j \mathbf{1}_n + \mathbf{K}\beta_j) = \mathbf{0}$, which is theoretically necessary for making the multi-class classification problem identifiable.

7. Conclusion

In this paper we have discussed Bayesian generalized kernel mixed models, including Bayesian generalized kernel models and Gaussian processes for classification. In particular, we have proposed fully Bayesian kernel methods based on the Silverman g-prior and a Bayesian model averaging method. We have developed an MCMC algorithm for parameter estimation, model selection and posterior prediction. Because of the connection between kernel methods and Gaussian processes, the MCMC algorithm can be immediately applied to sparse Gaussian processes.

Sparsity is often treated using machinery that is not straightforward to emulate within the Bayesian paradigm (e.g., loss functions with discontinuous derivatives). In the current paper we

have provided a framework in which sparsity is treated explicitly using standard Bayesian tools. Our empirical results show that this framework can yield prediction performance that is comparable with the best non-Bayesian methods, while retaining the advantages (e.g., the natural treatment of hyperparameters and of uncertainty) of the Bayesian approach. The computational requirements of the framework are reasonable at the scale of the experiments we have performed; moreover, as emphasized in non-Bayesian treatments, the imposition of sparsity has computational advantages within our framework, advantages that we have only partially exploited in the work described here.

Acknowledgments

The authors would like to thank the Action Editor and three anonymous referees for their constructive comments and suggestions on the original version of this paper. Zhihua Zhang acknowledges support from the Natural Science Foundations of China (No. 61070239), the 973 Program of China (No. 2010CB327903), the Doctoral Program of Specialized Research Fund of Chinese Universities (No. 20090101120066), and the Fundamental Research Funds for the Central Universities.

Appendix A. Pseudo Matlab Code for the Updates of s and β_γ

Algorithm 1 Pseudo Matlab code for updates of $\mathbf{s}$ and β_γ

Input: $\tilde{\mathbf{K}}_\gamma, \Sigma_\gamma, \mathbf{s}$;
Calculate and $\Upsilon_\gamma^{-1} = (\Sigma_\gamma + \tilde{\mathbf{K}}_\gamma' \tilde{\mathbf{K}}_\gamma)^{-1}$ and $\mathbf{Q}_\gamma^{-1} = \mathbf{I}_n - \tilde{\mathbf{K}}_\gamma \Upsilon_\gamma^{-1} \tilde{\mathbf{K}}_\gamma'$;
Calculate $\mathbf{A} = \Upsilon_\gamma^{-1} \tilde{\mathbf{K}}_\gamma'$, $\mathbf{b} = \mathbf{A}\mathbf{s}$ and $\mathbf{H} = \tilde{\mathbf{K}}_\gamma \mathbf{A}$;
for $i = 1$ **to** n **do**
  $a \leftarrow \mathbf{s}(i)$;
  $w \leftarrow \mathbf{H}(i,i)/(1 - \mathbf{H}(i,i))$;
  $\rho \leftarrow 1 + w$;
  $\mu \leftarrow (1+w)\tilde{\mathbf{K}}_\gamma(i,:)\mathbf{b} - w\mathbf{s}(i)$;
  if $y(i) == 1$ **then**
    $s(i) \leftarrow ltnormrnd(\mu, \rho, 0)$; ▷ Generate from a left-truncated normal distribution
  else
    $s(i) \leftarrow rtnormrnd(\mu, \rho, 0)$; ▷ Generate a right-truncated normal distribution
    $\mathbf{b} \leftarrow \mathbf{b} + (\mathbf{s}(i) - a)\mathbf{A}(:,i)$;
  end if
end for
$\beta_\gamma \leftarrow mvnormrnd(\mathbf{b}, \Upsilon_\gamma^{-1}, 1)$. ▷ Generate from a multivariate normal distribution
Output: $\mathbf{s}$ and β_γ.

Appendix B. The Proof of Proposition 1

Proof. Recall that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{r} \psi_k(\mathbf{x}_i)\psi_k(\mathbf{x}_j).$$

The Karhunen-Loève expansion of $\zeta(\mathbf{x})$ is then given by

$$\zeta(\mathbf{x}) = \sum_{k=1}^{r} b_k \psi_k(\mathbf{x}),$$

where the b_k are random variables, which are given by $b_k = \frac{1}{l_k} \int \zeta(\mathbf{x}) \psi_k(\mathbf{x}) d\mathbf{x}$. It follows that the b_k are independent Gaussian variables with $E(b_k) = 0$ and $E(b_k^2) = g^{-1}$. We thus have the first part. To prove the second part of this proposition, we consider any n-dimensional vector $\zeta = (\zeta(\mathbf{x}_1), \ldots, \zeta(\mathbf{x}_n))'$. It is obvious that $E(\zeta(\mathbf{x}_i)) = 0$ and

$$\begin{aligned} E\big(\zeta(\mathbf{x}_i)\zeta(\mathbf{x}_j)\big) &= E\Big(\sum_{k=1}\sum_{k'} b_k \psi_k(\mathbf{x}_i) b_{k'} \psi_{k'}(\mathbf{x}_j)\Big) \\ &= g^{-1} \sum_{k=1}^{r} \psi_k(\mathbf{x}_i)\psi_k(\mathbf{x}_j) = g^{-1} K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned}$$

This implies that ζ follows a multivariate normal distribution $N_n(\mathbf{0},\ g^{-1}\mathbf{K})$. Consequently, $\zeta(\mathbf{x})$ follows a Gaussian process.

References

J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the Twenty-first International Conference on Machine Learning*, 2004.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley and Sons, New York, 1994.

S. P. Brooks. Quantitative convergence diagnosis for MCMC via CUSUMS. *Statistics and Computing*, 8(3):267–274, 1998.

C. M. Carvalho, N. G. Polson, and J. G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

S. Chakraborty, M. Ghosh, and B. K. Mallick. Bayesian nonlinear regression for large p small n problems. Technical report, Department of Statistics, University of Florida, 2005.

P. J. Diggle, J. A. Tawn, and R. A. Moyeed. Model-based geostatistics (with discussions). *Applied Statistics*, 47(3):299–350, 1998.

M. Figueiredo. Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1150–1159, 2003.

E. I. George and R. E. McCulloch. Approaches for Bayesian variable selection. *Statistica Sinica*, 7:339–374, 1997.

M. A. Girolami and S. Rogers. Variational Bayesian multinomial probit regression with Gaussian process priors. *Neural Computation*, 18:1790–1817, 2006.

P. J. Green. Discussion of Dr. Silverman's paper. *Journal of the Royal Statistical Society, Series B*, 47(1):29, 1985.

P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

A.K. Gupta and D.K. Nagar. *Matrix Variate Distributions*. Chapman & Hall/CRC, 2000.

C. Hans. Bayesian lasso regression. *Biometrika*, 96(4):835–845, 2009.

D. A. Harville. Maximum likelihood approaches to variance component estimation and to related problems. *Journal of the American Statistical Association*, 72(358):320–338, 1977.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, 2001.

C. C. Holmes and L. Held. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, 1(1):145–168, 2006.

R. Kohn, M. Smith, and D. Chan. Nonparametric regression using linear combinations of basis functions. *Statistics and Computing*, 11:313–322, 2001.

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: the informative vector machine. In *Advances in Neural Information Processing Systems 15*, 2003.

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.

Q. Li and N. Lin. The Bayesian elastic net. *Bayesian Analysis*, 5(1):151–170, 2010.

F. Liang, R. Paulo, G. Molina, M. A. Clyde, and J. O. Berger. Mixtures of g-priors for Bayesian variable selection. *Journal of the American Statistical Association*, 103(481):410–423, 2008.

F. Liang, K. Mao, M. Liao, R. F. MacLehose, and D. B. Dunson. Nonparametric Bayesian kernel models. In *Discussion Paper 2005-09, Duke University ISDS*, 2009.

R. F. MacLehose and D. B. Dunson. Nonparametric Bayes kernel-based priors for functional data analysis. *Statistica Sinica*, 19:611–629, 2009.

B. K. Mallick, D. Ghosh, and M. Ghosh. Bayesian classification of tumours by using gene expression data. *Journal of the Royal Statistical Society Series B*, 67:219–234, 2005.

K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, New York, 1979.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman and Hall, New York, 1989.

T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 362–369, 2001.

R. M. Neal. Regression and classification using Gaussian process priors (with discussion). In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics*, volume 6, pages 475–501. Oxford University Press, 1999.

D. J. Nott and P. J. Green. Bayesian variable selection and the Swendsen-Wang algorithm. *Journal of Computational and Graphical Statistics*, 13(1):1–17, 2004.

T. Park and G. Casella. The Bayesian lasso. *Journal of the American Statistical Association*, 103 (482):681–686, 2008.

N. S. Pillai, Q. Wu, F. Liang, S. Mukherjee, and R. L. Wolpert. Characterizing the function space for Bayesian kernel models. *Journal of Machine Learning Research*, 8:1769–1797, 2007.

J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

A. E. Raftery, D. Madigan, and D. Hoeting. Bayesian model averaging for linear regression. *Journal of the American Statistical Association*, 92:179–191, 1997.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

G. Rätsch, T. Onoda, and K. Müller. Soft margins for Adaboost. *Machine Learning*, 42:287–320, 2001.

N. Sha, M. Vannucci, M. G. Tadesse, P. J. Brown, I. Dragoni, N. Davies T. C. Roberts, A. Contestabile, M. Salmon, C. Buckley, and F. Falciani. Bayesian variable selection in multinomial probit models to identify molecular signatures of disease stage. *Biometrics*, 60:812–819, 2004.

B. W. Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting (with discussion). *Journal of the Royal Statistical Society, Series B*, 47(1):1–52, 1985.

M. Smith and R. Kohn. Nonparametric regression using Bayesian variable selection. *Journal of Econometrics*, 75:317–344, 1996.

A. J. Smola and P. Bartlett. Sparse greedy Gaussian process regression. In *Advances in Neural Information Processing Systems 13*, 2001.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Neural Information Processing Systems 18*, 2006.

E. L. Snelson. *Flexible and Efficient Gaussian Process Models for Machine Learning*. PhD thesis, University College London, 2007.

P. Sollich. Bayesian methods for support vector machines: evidence and predictive class probabilities. *Machine Learning*, 46:21–52, 2001.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

G. Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, 1990.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian Statistics 7*, pages 723–732. Oxford University Press, 2003.

C. K. I. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Trans. PAMI*, 20(12):1342–1351, 1998.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, 2001.

A. Zellner. On assessing prior distributions and Bayesian regression analysis with $g-$prior distributions. In P. K. Goel and A. Zellner, editors, *Bayesian Inference and Decision Techniques: Essays in Honor of Bruno de Finetti*, pages 233–243. North-Holland, Amsterdam, 1986.

Z. Zhang and M. I. Jordan. Bayesian multicategory support vector machines. In *the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI)*, 2006.

Z. Zhang, G. Wu, and E. Y Chang. Semiparametric regression using Student t processes. *IEEE Transactions on Neural Networks*, 18(6):1572–1588, 2007.

Z. Zhang, M. I. Jordan, and D.-Y. Yeung. Posterior consistency of the Silverman g-prior in Bayesian model choice. In *Advances in Neural Information Processing Systems 22*, 2008.

J. Zhu and T. Hastie. Kernel logistic regression and the import vector machines. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005.

Journal of Machine Learning Research 12 (2011) 141-202 Submitted 3/09; Revised 7/10; Published 1/11

Training SVMs Without Offset

Ingo Steinwart INGO.STEINWART@MATHEMATIK.UNI-STUTTGART.DE
Institut für Stochastik und Anwendungen
Fakultät für Mathematik und Physik
Universität Stuttgart
Pfaffenwaldring 57
D-70569 Stuttgart, Germany

Don Hush DHUSH@LANL.GOV
ISR-2, Mail Stop B244
Los Alamos National Laboratory
Los Alamos, NM 87545, USA

Clint Scovel JCS@LANL.GOV
CCS-3, Mail Stop B265
Los Alamos National Laboratory
Los Alamos, NM 87545, USA

Editor: Sathiya Keerthi

Abstract

We develop, analyze, and test a training algorithm for support vector machine classifiers without offset. Key features of this algorithm are a new, statistically motivated stopping criterion, new warm start options, and a set of inexpensive working set selection strategies that significantly reduce the number of iterations. For these working set strategies, we establish convergence rates that, not surprisingly, coincide with the best known rates for SVMs with offset. We further conduct various experiments that investigate both the run time behavior and the performed iterations of the new training algorithm. It turns out, that the new algorithm needs significantly less iterations and also runs substantially faster than standard training algorithms for SVMs with offset.

Keywords: support vector machines, decomposition algorithms

1. Introduction

Historically, support vector machines (SVMs) were motivated by a geometrical illustration of their linear decision surface in the feature space. This illustration justified the use of an offset b that moves the decision surface from the origin. However, in recent years it has become increasingly evident that this geometrical interpretation has serious flaws, see, for example, Steinwart (2003) for some illustrations, when considering kernels that have a large feature space such as the Gaussian RBF kernels. In addition, the current approach, see, for example, Steinwart and Christmann (2008), for investigating the generalization performance of SVMs for classification does not suggest that the offset offers any improvement for such kernels. On the other hand, the SVM optimization problem with offset imposes more restrictions on solvers than the optimization problem without offset does. For example, the offset leads to an additional equality constraint in the dual optimization problem, which in turn makes it necessary to update at least two dual variables at each iteration of commonly used solvers such as sequential minimal optimization (SMO). In addition, such solvers

©2011 Ingo Steinwart, Don Hush and Clint Scovel.

can only update certain pairs of dual variables, namely the pairs whose update still satisfies the equality constraint. Moreover, the offset makes it relatively expensive to calculate the duality gap, see Cristianini and Shawe-Taylor (2000), which may serve as a stopping criterion for these solvers, and hence one usually considers upper bounds of this gap such as the one from the maximal violating pair algorithm, see, for example, Lin (2002b).

Despite these issues, research on algorithmic solutions has, with a few exceptions such as Kecman et al. (2005), Vogt (2002) and Huang et al. (2006), so far mostly focused on SVM formulations with offset. We refer to Lin (2001), Keerthi et al. (2001), Lin (2002a), Hush and Scovel (2003), List and Simon (2004), Fan et al. (2005), List and Simon (2005), Chen et al. (2006), Hush et al. (2006), Glasmachers and Igel (2006), List et al. (2007), List and Simon (2007) and the references therein. One motivation for this focus may be the fact that certain other SVM formulations such as one-class SVMs and SVMs for finding the smallest ball enclosing all data points do have an offset, and hence these formulations can be dealt with (almost) simultaneously. Moreover, it was noted early on that for SVMs with offset, the resulting equality constraint in the dual optimization problem can be avoided, if the offset is also penalized in the regularizer. The package BSVM by Hsu and Lin (2002) and Hsu and Lin (2006) implements this idea for the hinge loss, while Mangasarian and Musicant (2001) and Fung and Mangasarian (2001) use this idea in conjunction with other margin-based loss functions.

The goal of this work is to fill the described gap by developing algorithms for SVMs without offset. As it turns out, these algorithms not only achieve a classification accuracy that is comparable to the one for SVMs with offset, but also run significantly faster. This improvement is made possible by a couple of new algorithmic ideas that are not straightforward to implement for SVMs with offset. Inspired by recent results on the statistical performance of SVMs, see (Steinwart and Christmann, 2008, Chapter 7.4), the first idea is a new stopping criterion, which is, roughly speaking, a clipped duality gap. The second idea is a new working set selection strategy. As mentioned above, SMO type approaches for SVMs without offset can, in principle, update a single dual variable at each iteration. However, our experiments show that this approach does not lead to sufficiently fast training algorithms, and hence we will describe in detail, how an SMO type approach for two dual variables works. Of course, such an approach requires a good working set selection strategy. To identify one, we describe and test various strategies that try to find a pair of dual variables whose update approximately maximizes the gain in the dual objective function. Basically all these strategies first identify *one* dual variable whose update maximizes the gain in the dual objective and then search for a second variable that matches well to the first variable. Clearly, the first search is $O(n)$, where n is the number of samples, while the order for the second search will be between $O(1)$ and $O(n)$ depending on the particular strategy. Interestingly, we will see that certain *combinations* of $O(1)$ strategies for finding the second variable need almost as few iterations as an $O(n^2)$ search over *all* pairs. In particular, these combinations essentially need the same number of iterations as some natural $O(n)$ strategies for choosing the second dual variable do. Since each iteration of the latter strategies is obviously more expensive, the $O(1)$ combinations enjoy significantly shorter run times as will be seen in the experiments.

For solvers using the new stopping criterion and (combinations of) the working set strategies mentioned above, we further establish theoretical guarantees on the number of iterations performed. Not surprisingly, it turns out that the analysis without bias is less complicated than the one for the offset case, while the resulting guarantees coincide with the best known guarantees for solvers with offset. Recall that the latter can be obtained by combining the analysis of so-called rate certifying

algorithms, see List and Simon (2005), Hush et al. (2006) and List and Simon (2007), with a recent analysis of the duality gap, see List et al. (2007). Unlike the rate certifying algorithms for SVMs with offset, however, our algorithms not only possess these guarantees, but also run significantly faster than typically implemented training algorithms, as our experimental section shows.

We also consider the possibility to initialize the solver with (transformed) previous solutions when working on a grid of hyper-parameters. Here it first turns out that the missing equality constraint gives us more freedom to transform these solutions. We describe and test several such transformations ranging from relatively simple to quite complex procedures. In the experiments, we then see that SVMs without offset profit more from simple warm start initializations than SVMs with offset do. In addition, the more complex warm start strategies, which cannot be directly implemented for SVMs with offset, lead to further improvements. In particular, for data sets containing a few thousand samples, SVMs without offset profit about twice as much from a good warm start strategy than SVMs with offset do. As a result, our SVMs without offset are approximately 7 times faster than SVMs with offset on these data sets, if the hyper-parameters are determined by a cross-validation approach.

This work is organized as follows: Section 2 introduces an SMO type algorithm for SVMs without offset that performs one dual variable update per iteration. We further describe the new stopping criterion based on a clipped duality gap as well as several warm start strategies. Section 3 then generalizes this algorithm to handle two variables at each iteration. In particular, we describe how to solve the corresponding two dimensional optimization problem exactly. Furthermore, we present several working set selection strategies. Section 4 contains our theoretical analysis, while the experiments can be found in Section 5. Finally, concluding remarks can be found in Section 6 and an appendix contains detailed data from our experiments.

2. The Basic Algorithm: Optimizing One Coordinate

Throughout this paper, we write $[t]_a^b := \max\{a, \min\{b,t\}\}$, $t \in \mathbb{R}$, $b > a$, for the clipping operation that clips a real number t whenever it is outside the interval $[a,b]$. To introduce SVMs without offset term, let us consider a training set $T = ((x_1,y_1),\ldots,(x_n,y_n)) \in (X \times \{-1,1\})^n$ and a function $f : X \to \mathbb{R}$. Then the empirical hinge risk of f is defined by

$$\mathcal{R}_{L,T}(f) := \frac{1}{n}\sum_{i=1}^{n} w_i L(y_i, f(x_i))\,,$$

where L denotes the hinge loss $L(y,t) := \max\{0, 1-yt\}$, and $w_i > 0$ is a weight associated to the sample (x_i,y_i). For example, in ordinary binary classification we have $w_i = 1$ for all $i = 1,\ldots,n$, whereas in weighted binary classification we have two real numbers $w_{\text{pos}} > 0$ and $w_{\text{neg}} > 0$ such that $w_i = w_{\text{pos}}$ if $y_i = 1$ and $w_i = w_{\text{neg}}$ if $y_i = -1$. In the following, we will exclusively consider the case of weighted binary classification, which, of course, includes the case of ordinary binary classification. Now the SVM without offset solves the problem

$$f_{T,\lambda} \in \operatorname*{argmin}_{f \in H} \lambda \|f\|_H^2 + \mathcal{R}_{L,T}(f)\,, \tag{1}$$

where H is the reproducing kernel Hilbert space (RKHS) of a kernel k. The statistical analysis of SVMs shows, see (Steinwart and Christmann, 2008, Corollary 5.34), that a necessary condition for learning in the sense of universal consistency is the strict positive definiteness of the kernel k. In

the following, we adopt this point of view, partially also because for kernels that fail to be strictly positive definite the offset may actually improve the learning performance, both theoretically and practically. In other words, we assume throughout this paper that the *Gram matrix* $(k(x_i,x_j))_{i,j=1}^n$ is strictly positive definite whenever the data points $x_1,\ldots,x_n$ are mutually distinct.[1] Considering the case $n=1$, it is then easy to conclude that $k(x,x)>0$ for all $x\in X$, and hence we may and will additionally assume that k is *normalized*, that is, $k(x,x)=1$ for all $x\in X$. Although this assumption is not really necessary, it makes the description of the algorithm significantly simpler. In addition, it is satisfied by many popular kernels on $X=\mathbb{R}^d$ such as the Gaussian RBF kernel $k(x,x') := \exp\bigl(-\sigma^2\|x-x'\|_2^2\bigr)$, and the Poisson kernel $k(x,x') := \exp\bigl(-\sigma\|x-x'\|_2\bigr)$, where in both cases $\sigma>0$ is called the width parameter. Furthermore, note that for strictly positive definite and normalized kernels we have $|k(x,x')|=1$ if and only if $x=x'$. For the Gaussian and Poisson kernel, this characterization is, of course, trivial, and in the general case, it quickly follows when considering the case $n=2$.

To derive an algorithm that produces an approximate solution of (1) we first multiply the objective function in (1) by $\frac{1}{2\lambda}$ and introduce slack variables. This leads to the following optimization problem:

$$\begin{aligned} \operatorname*{arg\,min}_{(f,\xi)} \quad & P_C(f,\xi) := \frac{1}{2}\|f\|_H^2 + \sum_{i=1}^n C_i\,\xi_i \\ \text{s.t.} \quad & \xi_i \geq 0\,, && i=1,\ldots,n, \\ & \xi_i \geq 1-y_i f(x_i)\,, && i=1,\ldots,n, \end{aligned} \tag{2}$$

where $C_i := \frac{w_{\text{pos}}}{2\lambda n}$ if $y_i=1$ and $C_i := \frac{w_{\text{neg}}}{2\lambda n}$ otherwise. Analogously to the offset case, see, for example, (Cristianini and Shawe-Taylor, 2000, p. 107f), one can then show that the dual of this problem is

$$\begin{aligned} \max_{\alpha\in\mathbb{R}^n} \quad & W(\alpha) := \langle e,\alpha\rangle - \frac{1}{2}\langle\alpha,K\alpha\rangle \\ \text{s.t.} \quad & 0\leq\alpha_i\leq C_i\,, && i=1,\ldots,n\,, \end{aligned} \tag{3}$$

where $e := (1,\ldots,1)\in\mathbb{R}^n$ and K is the $n\times n$ matrix with entries $K_{i,j} := y_i y_j k(x_i,x_j)$, $i,j=1,\ldots,n$. In addition, the Karush-Kuhn-Tucker (KKT) conditions are

$$\begin{aligned} \bigl(y_i f(x_i)+\xi_i-1\bigr)\alpha_i &= 0, && i=1,\ldots,n, \\ (C_i-\alpha_i)\xi_i &= 0, && i=1,\ldots,n, \end{aligned}$$

and a solution $\alpha^*\in[0,C] := [0,C_1]\times\cdots\times[0,C_n]$ of (3) yields a solution (f^*,ξ^*) of (2) by setting

$$f^* := \sum_{i=1}^n y_i\alpha_i^* k(x_i,\cdot)$$

and $\xi_i^* := \max\{0,1-y_i f^*(x_i)\}$, $i=1,\ldots,n$. Obviously, (3) is identical to the standard dual SVM problem besides the missing equality constraint $\langle y,\alpha\rangle=0$. Now recall that this equality constraint makes it necessary to update at least two coordinate values at a time to ensure feasibility, while in (3) we can update *single* coordinates. Some ideas for such a single direction update will be recalled in the following subsections to provide the background for working sets of size two considered in Section 3.

1. If we have samples with $x_i=x_j$ for some $i\neq j$, the Gram matrix of a strictly positive definite kernel k, is, of course, no longer strictly positive definite. The algorithmic consequences of this observation will be discussed in detail in Section 3. Here, we only note that our solver will need a strictly positive kernel, but not a strictly positive Gram matrix.

2.1 Working Sets of Size One

To recall the one-dimensional update step, see also (Cristianini and Shawe-Taylor, 2000, p. 131ff), we define

$$\nabla W_i(\alpha) := \frac{\partial W}{\partial \alpha_i}(\alpha) = 1 - \sum_{j=1}^{n} \alpha_j K_{i,j}\,.$$

Moreover, for an $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$ and an index $i \in \{1, \ldots, n\}$ we write $\alpha^{\setminus i} := \alpha - \alpha_i e_i$, where e_i denotes the i-th vector of the standard basis of $\mathbb{R}^n$, that is, $\alpha^{\setminus i}$ equals α in all coordinates except the i-th, where it equals zero. Now basic calculus together with $K_{i,i} = 1$ for normalized kernels shows that

$$\tilde{\alpha}_i \mapsto W(\alpha^{\setminus i} + \tilde{\alpha}_i e_i) = \langle \alpha^{\setminus i}, e \rangle + \tilde{\alpha}_i - \frac{1}{2}\langle \alpha^{\setminus i}, K\alpha^{\setminus i} \rangle - \tilde{\alpha}_i \langle e_i, K\alpha^{\setminus i} \rangle - \frac{1}{2}\tilde{\alpha}_i^2$$

attains its *global* maximum over $\mathbb{R}$ at

$$\alpha_i^* = 1 - \langle e_i, K\alpha^{\setminus i} \rangle = 1 - \sum_{j \neq i} \alpha_j K_{i,j} = \nabla W_i(\alpha) + \alpha_i\,.$$

Obviously, if $\alpha_i^* \in [0, C_i]$, the function $\alpha_i \mapsto W(\alpha^{\setminus i} + \alpha_i e_i)$ also attains its maximum over $[0, C_i]$ at α_i^*. On the other hand, if, for example, $\alpha_i^* > C_i$, then a simple concavity argument shows that the function attains its maximum over $[0, C_i]$ at C_i. By this and an analogous consideration in the case $\alpha_i^* < 0$ we hence see that the function $\alpha_i \mapsto W(\alpha^{\setminus i} + \alpha_i e_i)$ attains its maximum over $[0, C_i]$ at

$$\alpha_i^{new} := [\nabla W_i(\alpha) + \alpha_i]_0^{C_i}\,. \tag{4}$$

The next question is in which coordinate i should we perform the update. A simple and straightforward approach, see, for example, (Cristianini and Shawe-Taylor, 2000, p. 132), is to update for each coordinate $i = 1, \ldots, n$ iteratively. A more advanced idea, see Vogt (2002) and also (Huang et al., 2006, Chapter 3), is to choose KKT violators for the update, that is, indices that, for a specified $\varepsilon > 0$, satisfy

$$\begin{aligned} \alpha_i < C_i \quad &\text{and} \quad \nabla W_i(\alpha) > \varepsilon\,, \\ \text{or} \quad \alpha_i > 0 \quad &\text{and} \quad \nabla W_i(\alpha) < -\varepsilon\,. \end{aligned} \tag{5}$$

Obviously, the extreme case of this approach is to look for the indices

$$\begin{aligned} i_{\text{up}}^* &\in \arg\max\bigl\{\nabla W_i(\alpha) : \alpha_i < C_i\bigr\}\,, \\ i_{\text{down}}^* &\in \arg\min\bigl\{\nabla W_i(\alpha) : \alpha_i > 0\bigr\} \end{aligned}$$

and to pick the index of these two candidates whose gradient has the larger absolute value. Another idea, which is motivated by Glasmachers and Igel (2006), Hush et al. (2006), Hush and Scovel (2003) and List and Simon (2005), is to choose the coordinate i^* whose update achieves the largest improvement for the objective dual value $W(\alpha)$. In other words, it performs the update in the direction

$$i^* \in \arg\max_{i=1,\ldots,n} W(\alpha + \delta_i e_i) - W(\alpha)\,, \tag{6}$$

where $\delta_i := \alpha_i^{new} - \alpha_i$ denotes the difference between the new and the old value of α_i. Using the following trivial lemma, it is easy to see that Procedure 1 solves (6).

Lemma 1 *For $\delta \in \mathbb{R}$ and $i = 1, \ldots, n$ we have*

$$W(\alpha + \delta e_i) - W(\alpha) = \delta \cdot (\nabla W_i(\alpha) - \delta/2)\,.$$

Proof By the symmetry of K we find

$$\langle \alpha, K\alpha \rangle - \langle \alpha + \delta e_i, K(\alpha + \delta e_i) \rangle = -2\delta \langle \alpha, Ke_i \rangle - \delta^2\,.$$

Combining this with $\langle e, \alpha + \delta e_i \rangle - \langle e, \alpha \rangle = \delta$ yields the assertion. ■

Procedure 1 Calculate $i^* \in \arg\max_{i=1,\ldots,n} \delta_i \cdot (\nabla W_i(\alpha) - \delta_i/2)$

```
bestgain ← −1
for i = 1 to n do
    α_i^* ← [∇W_i(α) + α_i]_0^{C_i}
    δ ← α_i^* − α_i
    gain ← δ · (∇W_i(α) − δ/2)
    if gain > bestgain then
        bestgain ← gain
        i^* ← i
    end if
end for
```

2.2 Stopping Criteria

Several stopping criteria for the SVM *with* offset have been proposed and a straightforward approach is to adapt one of these. For example, one can stop if both $\nabla W_{i^*_{\text{up}}}(\alpha) \le \varepsilon$ and $\nabla W_{i^*_{\text{down}}}(\alpha) \ge -\varepsilon$, that is, if the KKT conditions are satisfied up to some predefined $\varepsilon > 0$. Another simple idea is to use the duality gap as a stopping criterion, see, for example, (Cristianini and Shawe-Taylor, 2000, p. 109 & 128). For SVMs *without* offset this duality gap is of the form

$$\text{gap}(\alpha) := \langle \alpha, K\alpha \rangle - \langle e, \alpha \rangle + \sum_{i=1}^{n} C_i \, [\nabla W_i(\alpha)]_0^\infty \;\le\; \varepsilon\,, \tag{7}$$

where $\varepsilon > 0$ does not necessarily have the same value as above.

In this work, however, we consider a little more involved stopping criterion that is based on recent results from the statistical analysis of SVMs in Steinwart et al. (2007). Namely, it was shown in Steinwart et al. (2007) that an $f^* \in H$ satisfying

$$\lambda \|f^*\|_H^2 + \mathcal{R}_{L,T}([f^*]_{-1}^1) \le \min_{f \in H} \lambda \|f\|_H^2 + \mathcal{R}_{L,T}(f) + \varepsilon \tag{8}$$

for yet another pre-defined $\varepsilon > 0$ satisfies the same oracle inequality up to 4ε as the true solution $f_{T,\lambda}$. Moreover, a more careful analysis of Steinwart et al. (2007) shows that the factor 4 can be essentially removed, so that for say $\varepsilon := 0.001$ the learning guarantees for the approximate solution f^* are at most 0.1% worse than those for the true solution $f_{T,\lambda}$. To develop a stopping criterion from

this observation, we denote the minimum of the objective function P_C in (2) by P_C^*. Moreover, for a dual point $\alpha \in [0,C]$ we define, as usual, a corresponding primal function by

$$f := \sum_{i=1}^{n} \alpha_j y_j k(x_j, \cdot)$$

and its corresponding slack variables by $\xi_i := \max\{0, 1 - y_i f(x_i)\}$, $i = 1, \ldots, n$. Using $1 - y_i f(x_i) = \nabla W_i(\alpha)$ and $\|f\|_H^2 = \langle \alpha, K\alpha \rangle$ as well as $P_C^* \geq W(\alpha) = \langle e, \alpha \rangle - \langle \alpha, K\alpha \rangle / 2$ and

$$\max\bigl\{0, 1 - y[t]_{-1}^{1}\bigr\} = 1 - y[t]_{-1}^{1} = [1 - yt]_0^2$$

for all $y = \pm 1$, $t \in \mathbb{R}$, we hence see that (8) is satisfied if

$$S(\alpha) := \langle \alpha, K\alpha \rangle - \langle e, \alpha \rangle + \sum_{i=1}^{n} C_i [\nabla W_i(\alpha)]_0^2 \;\leq\; \frac{\varepsilon}{2\lambda}. \tag{9}$$

Note that the statistical analysis of Steinwart et al. (2007) also suggests that the right hand side of (7) can be replaced by $\frac{\varepsilon}{2\lambda}$, where ε has the same value as in (9). Consequently, the difference between these two stopping criteria is the fact that (9) considers *clipped* slack variables, which may be substantially smaller than the unclipped slack variables used in (7). Moreover, unlike the duality gap stopping criterion for SVMs *with* offset, see (Cristianini and Shawe-Taylor, 2000, p. 109f), both (7) and (9) are directly computable since they do not require the offset.

To efficiently compute $S(\alpha)$ we first observe that the first two terms of the updated $S(\alpha + \delta e_i)$ can be easily computed from the first two terms of $S(\alpha)$. Indeed, if we write

$$\begin{aligned} T(\alpha) &:= \langle \alpha, K\alpha \rangle - \langle e, \alpha \rangle, \\ E(\alpha) &:= \sum_{i=1}^{n} C_i [\nabla W_i(\alpha)]_0^2, \end{aligned}$$

then we have $S(\alpha) = T(\alpha) + E(\alpha)$, and the calculations in the proof of Lemma 1 immediately show

$$T(\alpha + \delta e_i) = T(\alpha) - \delta\bigl(2\nabla W_i(\alpha) - 1 - \delta\bigr).$$

From this it is easy to derive an $O(n)$ procedure that updates $\nabla W(\alpha)$ and calculates $S(\alpha)$. Procedure 2 provides pseudocode for this task.

Procedure 2 Update $\nabla W(\alpha)$ in direction i by δ and calculate $S(\alpha)$

```
T(α) ← T(α) − δ(2∇W_i(α) − 1 − δ)
E(α) ← 0
for j = 1 to n do
    ∇W_j(α) ← ∇W_j(α) − δK_{i,j}
    E(α) ← E(α) + C_i · [∇W_i(α)]_0^2
end for
S(α) ← T(α) + E(α)
```

Now the basic idea of the 1D-SVM described in Algorithm 1 is to repeatedly look for the best direction i^* and update in this direction until the stopping criterion (9) is satisfied. However, a closer look at this algorithm shows that it contains one piece of pseudo-code that has not been discussed so far, namely the initialization of the solver. This initialization will be considered in the following subsection.

Algorithm 1 1D-SVM solver

initialize α, $\nabla W(\alpha)$, $T(\alpha)$, and $S(\alpha)$ by one of the Procedures from Section 2.3
while $S(\alpha) > \frac{\varepsilon}{2\lambda}$ **do**
  $i^* \leftarrow \arg\max_{i=1,\ldots,n} W(\alpha + \delta_i e_i) - W(\alpha)$
  $\delta \leftarrow [\nabla W_{i^*}(\alpha) + \alpha_{i^*}]_0^C - \alpha_{i^*}$
  $\alpha_{i^*} \leftarrow [\nabla W_{i^*}(\alpha) + \alpha_{i^*}]_0^C$
  use Procedure 2 to update $\nabla W(\alpha)$ in direction i^* by δ and calculate $S(\alpha)$
end while

2.3 Initialization

We also have to decide how to initialize α. Of course, there exist various approaches for this task, and in the following, we describe a few methods we have considered in this work.

I0 & W0: Cold Start With Zeros. Obviously, the most simple initialization is the *cold start* $\alpha \leftarrow 0$. Procedure 3 provides the pseudocode for this approach, which in the following we call I0 or W0.

Procedure 3 Initialize by $\alpha_i \leftarrow 0$ and compute $\nabla W(\alpha)$, $S(\alpha)$, and $T(\alpha)$.

$T(\alpha) \leftarrow 0$
$S(\alpha) \leftarrow 0$
for $i = 1$ to n **do**
  $\alpha_i \leftarrow 0$
  $\nabla W_i(\alpha) \leftarrow 1$
  $S(\alpha) \leftarrow S(\alpha) + C_i$
end for

I1 & W1: Cold Start With Kernel Rule. Another simple cold start is to initialize with $\alpha_i \leftarrow C_i$ for all $i = 1, \ldots, n$. Procedure 4 provides the pseudocode for this approach. In the following, we call this approach I1 or W1.

Procedure 4 Initialize by $\alpha_i \leftarrow C_i$ and compute $\nabla W(\alpha)$, $S(\alpha)$, and $T(\alpha)$.

$T(\alpha) \leftarrow 0$
$E(\alpha) \leftarrow 0$
for $i = 1$ to n **do**
  $\alpha_i \leftarrow C_i$
  $\nabla W_i(\alpha) \leftarrow 1$
  for $j = 1$ to n **do**
    $\nabla W_i(\alpha) \leftarrow \nabla W_i(\alpha) - C_j \cdot K_{i,j}$
  end for
  $T(\alpha) \leftarrow T(\alpha) - C_i \cdot \nabla W_i(\alpha)$
  $E(\alpha) \leftarrow E(\alpha) + C_i \cdot [\nabla W_i(\alpha)]_0^2$
end for
$S(\alpha) \leftarrow T(\alpha) + E(\alpha)$

Obviously, Procedure 3 is $O(n)$, whereas Procedure 4 is $O(n^2)$, and hence the latter seems to be prohibitive. On the other hand, Procedure 4 basically initializes with the classical kernel rule, see

(Devroye et al., 1996, Chapter 10), and hence its initial training error may be significantly smaller than that of Procedure 4. This in turn might lead to a smaller initial stopping criterion value $S(\alpha)$ and hence to less iterations of the solver. Of course, here is a lot of room for speculation, and hence we need to investigate the efficiency of both approaches in the experiments. However, it is worth noting that unlike Procedure 3, Procedure 4 cannot be directly implemented for SVMs *with* offsets. In addition, Procedure 4 requires the entire kernel matrix to be computed, and hence it may actually be prohibitive if this matrix does not fit into memory.

W2: Warm Start By Recycling Old Solution. Besides the cold starts mentioned above, there are also a couple of simple *warm starts* possible. To explain these, let us recall that often the hyper-parameter λ is chosen by a search over a grid $\Lambda = \{\lambda_1, \ldots, \lambda_m\}$ of candidate values. Let us assume that these values are ordered in the form $\lambda_1 > \cdots > \lambda_m$, and that we train the SVM in the order $\lambda_1, \ldots, \lambda_m$. Then the resulting n-dimensional vectors $C^{(1)}, \ldots, C^{(m)}$ defined by

$$C_i^{(j)} := \begin{cases} \frac{w_{\text{pos}}}{2\lambda_j n} & \text{if } y_i = 1 \\ \frac{w_{\text{neg}}}{2\lambda_j n} & \text{if } y_i = -1 \end{cases}$$

have the property $C_i^{(j)} < C_i^{(j+1)}$ for all $j = 1, \ldots, m-1$ and $i = 1, \ldots, n$. For $C^{(1)}$ we can then initialize with one of the above cold starts. Now observe that for $j \geq 2$ the approximate solution α^* obtained by training with $C^{\text{old}} := C^{(j-1)}$ is feasible for $C^{\text{new}} := C^{(j)}$, that is, $\alpha^* \in [0, C^{\text{new}}]$. Consequently, for $j \geq 2$ we can either initialize with a cold start, or with the warm start $\alpha \leftarrow \alpha^*$. Obviously, in this case we can also recycle $\nabla W(\alpha)$ and $T(\alpha)$. In addition, the ratio

$$\frac{C_i^{\text{new}}}{C_i^{\text{old}}} = \frac{\lambda_{j-1}}{\lambda_j}$$

is *independent* of i and hence this warm start can be very easily implemented as Procedure 5 shows.

Procedure 5 Initialize by $\alpha_i \leftarrow \alpha_i^*$ and compute $\nabla W(\alpha)$, $S(\alpha)$, and $T(\alpha)$.

$S(\alpha) \leftarrow T(\alpha^*) + \frac{C_1^{\text{new}}}{C_1^{\text{old}}} \cdot (S(\alpha^*) - T(\alpha^*))$

W4: Warm Start By Partially Expanding And Partially Recycling Old Solution. Apart from the simple warm start above there is another conceptionally simple warm start for expanding box constraints. Namely, if α^* denotes an approximate solution to C^{old} and $C^{\text{old}} < C^{\text{new}}$ this warm start initializes by $\alpha_i \leftarrow \alpha_i^*$ if $\alpha_i^* < C_i^{\text{old}}$ and by $\alpha_i \leftarrow C_i^{\text{new}}$ if $\alpha_i^* = C_i^{\text{old}}$. The idea behind this warm start is that *bounded* support vectors, that is, indices in

$$bSV := \{j : \alpha_j^* = C_j^{\text{old}}\}$$

may have the tendency to become larger, when the box constraint is loosened, while *unbounded* support vectors, that is, vectors in

$$uSV := \{j : 0 < \alpha_j^* < C_j^{\text{old}}\}$$

may not have this tendency.

The basic idea of an efficient implementation of this warm start method is to avoid calculating the gradient from scratch by recycling parts of the gradient from C^{old}. To be more precise, observe that, for fixed i, the sum $\sum_{j \in uSV} \alpha_j^* K_{i,j}$ remains unchanged by the described warm start, while

Procedure 6 Initialize bounded SVs by $\alpha_i \leftarrow C_i^{\text{new}}$ while keeping the rest unchanged and compute $\nabla W(\alpha)$, $S(\alpha)$, and $T(\alpha)$.

```
T(α) ← 0
E(α) ← 0
for i = 1 to n do
  if α_i = C_i^old then
    α_i ← C_i^new
  end if
end for
if 2 · #uSV < #bSV then
  for i = 1 to n do
    ∇W_i(α) ← (C_1^new / C_1^old) · ∇W_i(α) + (1 − C_1^new / C_1^old)(1 − Σ_{j∈uSV} α_j K_{i,j})
    T(α) ← T(α) − α_i · ∇W_i(α)
    E(α) ← E(α) + C_i^new · [∇W_i(α)]_0^2
  end for
else
  for i = 1 to n do
    ∇W_i(α) ← ∇W_i(α) + (C_i^old − C_i^new) Σ_{j∈bSV} K_{i,j}
    T(α) ← T(α) − α_i · ∇W_i(α)
    E(α) ← E(α) + C_i^new · [∇W_i(α)]_0^2
  end for
end if
S(α) ← T(α) + E(α)
```

$\sum_{j\in bSV} \alpha_j^* K_{i,j}$ is simply multiplied by $C_i^{\text{new}}/C_i^{\text{old}}$. Recall that the latter ratio is independent of i, and consequently we can update the gradients by either

$$\nabla W_i(\alpha) \leftarrow 1 - \frac{C_1^{\text{new}}}{C_1^{\text{old}}}\left(1 - \nabla W_i(\alpha^*) - \sum_{j\in uSV} \alpha_j^* K_{i,j}\right) - \sum_{j\in uSV} \alpha_j^* K_{i,j}$$

for all $i = 1,\ldots,n$, or

$$\nabla W_i(\alpha) \leftarrow \nabla W_i(\alpha) + (C_i^{\text{old}} - C_i^{\text{new}}) \sum_{j\in bSV} K_{i,j}\,, \qquad i = 1,\ldots,n\,,$$

where in the first formula we used

$$1 - \nabla W_i(\alpha^*) - \sum_{j\in uSV} \alpha_j^* K_{i,j} = \sum_{j\in bSV} \alpha_j^* K_{i,j}\,. \tag{10}$$

Note that the first method implicitly recycles $\sum_{j\in bSV} \alpha_j^* K_{i,j}$ by (10), while the second method implicitly recycles $\sum_{j\in uSV} \alpha_j^* K_{i,j}$. Obviously, depending on the number of bounded and unbounded support vectors either the first or the second method is more efficient, and hence should be chosen. We decided to pick the first or second method depending on whether $2 \cdot \#uSV < \#bSV$ or not. This decision was based on counts of the involved floating point operations and the fact that in all our experiments we stored the entire kernel matrix in the memory. However note that both methods

require to access some rows of the kernel matrix, and hence there is most likely a more efficient cut-off when only parts of the kernel matrix are stored in memory by caching. Since in general, the costs of computing a row of the kernel matrix depends on data set specific features, such as its dimensionality when using Gaussian kernels, there does not seem to exists a simple rule of thumb in this case, though. Consequently, we decided not to analyze this case carefully. Procedure 6 displays the corresponding pseudocode for this warm start, which we call W4. It is not hard to see, that in the worst case Procedure 6 is $O(n^2)$, while in the best case it is only $O(n)$. Since the average case cannot be easily analyzed, we need to experimentally evaluate whether this warm start is efficient or not.

W6: Warm Start By Partially Shrinking And Partially Recycling Old Solution. Let us now assume that we run through the λ-grid in reverse order. Then we have $C^{\text{old}} > C^{\text{new}}$, and hence we

Procedure 7 Initialize directions that violate the new box constrained by $\alpha_i \leftarrow C_i^{\text{new}}$ while keeping the rest unchanged and compute $\nabla W(\alpha)$, $S(\alpha)$, and $T(\alpha)$.

for $i = 1$ to n **do**
  if $\alpha_i > C_i^{\text{new}}$ **then**
    $\alpha_i \leftarrow C_i^{\text{new}}$
  end if
end for
$T(\alpha) \leftarrow 0$
$E(\alpha) \leftarrow 0$
if *#nuSV* $<$ *#bSV* **then**
  for $i = 1$ to n **do**
    $\nabla W_i(\alpha) \leftarrow 1 - \frac{C_1^{\text{new}}}{C_1^{\text{old}}} \cdot \left(1 - \nabla W_i(\alpha) - \sum_{j \in nuSV} \alpha_j^* K_{i,j} - \sum_{j \in nbSV} \alpha_j^* K_{i,j}\right)$
    $\nabla W_i(\alpha) \leftarrow \nabla W_i(\alpha) - \sum_{j \in nuSV} \alpha_j^* K_{i,j} - \sum_{j \in nbSV} C_j^{\text{new}} K_{i,j}$
    $T(\alpha) \leftarrow T(\alpha) - \alpha_i \cdot \nabla W_i(\alpha)$
    $E(\alpha) \leftarrow E(\alpha) + C_i^{\text{new}} \cdot [\nabla W_i(\alpha)]_0^2$
  end for
else
  for $i = 1$ to n **do**
    $\nabla W_i(\alpha) \leftarrow \nabla W_i(\alpha) + \sum_{j \in bSV} (C_j^{\text{old}} - C_j^{\text{new}}) K_{i,j}$
    $\nabla W_i(\alpha) \leftarrow \nabla W_i(\alpha) + \sum_{j \in nbSV} (\alpha_j^* - C_j^{\text{new}}) K_{i,j}$
    $T(\alpha) \leftarrow T(\alpha) - \alpha_i \cdot \nabla W_i(\alpha)$
    $E(\alpha) \leftarrow E(\alpha) + C_i^{\text{new}} \cdot [\nabla W_i(\alpha)]_0^2$
  end for
end if
$S(\alpha) \leftarrow T(\alpha) + E(\alpha)$

cannot immediately recycle the old approximate solution α^*. Nonetheless, there is a certain analogue to Procedure 6 possible. Indeed, we can initialize by $\alpha_i \leftarrow \alpha_i^*$ if $\alpha_i^* \leq C^{\text{new}}$ and by $\alpha_i \leftarrow C^{\text{new}}$ if $\alpha_i^* > C^{\text{new}}$. Again, the corresponding warm start needs some work to find an efficient implementation that recycles suitable parts of the gradient. In order to explain such an implementation we

split the set *uSV* into

$$\begin{aligned} nuSV &:= \{j : 0 < \alpha_j^* \le C_j^{\text{new}}\}, \\ nbSV &:= \{j : C_j^{\text{new}} < \alpha_j^* < C_j^{\text{old}}\}, \end{aligned}$$

where we note that we use a slight abuse of the letters u and b in this notation. Now note that the initialization above multiplies all $\alpha_j^* \in bSV$ by the factor $C_1^{\text{new}}/C_1^{\text{old}}$, while it keeps all $\alpha_j^* \in nuSV$ unchanged. Obviously, both update rules make it possible to recycle parts of the gradient. Unfortunately, however, for $\alpha_j^* \in nbSV$, the situation is more complicated and no simple recycling is possible. Thus, Procedure 7, which displays the corresponding pseudocode, is a little more complicated than Procedure 6. Nonetheless, all remarks concerning the computational requirements of Procedure 6 also apply to Procedure 7, and the same holds true for the rule that decides which part of the gradient is recycled. In the following, we call this approach displayed in Procedure 7, W6.

W3 & W5: Warm Start By Scaling Old Solution. Finally, there is an easy warm start option that works regardless of the direction we run through the λ-grid. Indeed, we can always initialize by $\alpha_i \leftarrow \alpha_i^* \cdot C_1^{\text{new}}/C_1^{\text{old}}$. The Procedure 8 shows the corresponding $O(n)$ pseudocode. Depending on whether $C_1^{\text{old}} < C_1^{\text{new}}$ or $C_1^{\text{old}} > C_1^{\text{new}}$ we call this approach W3 or W5, respectively.

Procedure 8 Initialize by $\alpha_i \leftarrow \alpha_i^* \cdot C_1^{\text{new}}/C_1^{\text{old}}$ and compute $\nabla W(\alpha)$, $S(\alpha)$, and $T(\alpha)$.

$T(\alpha) \leftarrow 0$
$E(\alpha) \leftarrow 0$
for $i = 1$ to n **do**
  $\alpha_i \leftarrow \frac{C_1^{\text{new}}}{C_1^{\text{old}}} \cdot \alpha_i^*$
  $\nabla W_i(\alpha) \leftarrow 1 - \frac{C_1^{\text{new}}}{C_1^{\text{old}}} \cdot \big(1 - \nabla W_i(\alpha)\big)$
  $T(\alpha) \leftarrow T(\alpha) - \alpha_i \cdot \nabla W_i(\alpha)$
  $E(\alpha) \leftarrow E(\alpha) + C_i^{\text{new}} \cdot [\nabla W_i(\alpha)]_0^2$
end for
$S(\alpha) \leftarrow T(\alpha) + E(\alpha)$

3. Working Sets of Size Two

So far, our algorithm performs an update in one coordinate per iteration. Let us now consider an algorithm which performs an update in *two* coordinates per iteration. To this end, let us first present the following, simple lemma that computes the gain of a 2-dimensional update.

Lemma 2 *For $\delta_i, \delta_j \in \mathbb{R}$ and $i, j = 1, \ldots, n$ we have*

$$W(\alpha + \delta_i e_i + \delta_j e_j) - W(\alpha) = \delta_i \cdot (\nabla W_i(\alpha) - \delta_i/2) + \delta_j \cdot (\nabla W_j(\alpha) - \delta_j/2) - \delta_i \delta_j K_{i,j}\,.$$

Proof Applying Lemma 1 twice and using the formula $\nabla W_j(\alpha + \delta_i e_i) = \nabla W_j(\alpha) - \delta_i K_{i,j}$ we find the assertion. ■

3.1 Solving the Two-Dimensional Problem Exactly

In order to describe an algorithm that updates two variables at each iteration we first have to investigate how the two-variable update looks like in detail. To this end, we fix two coordinates $i, j \in \{1, \ldots, n\}$ with $i \neq j$ and consider the function

$$(\tilde{\alpha}_i, \tilde{\alpha}_j) \mapsto W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j) := W(\alpha^{\setminus i,j} + \tilde{\alpha}_i e_i + \tilde{\alpha}_j e_j)\,,$$

where $\alpha^{\setminus i,j} := \alpha - \alpha_i e_i - \alpha_j e_j$ is a fixed vector whose i-th and j-th coordinates equal zero. A simple calculation then shows

$$\begin{aligned} W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j) &= \langle e, \alpha^{\setminus i,j} \rangle + \tilde{\alpha}_i + \tilde{\alpha}_j - \frac{1}{2}\langle \alpha^{\setminus i,j}, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_i \langle e_i, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_j \langle e_j, K\alpha^{\setminus i,j} \rangle \\ &\quad - \frac{1}{2}\left(\tilde{\alpha}_i^2 + 2\tilde{\alpha}_i\tilde{\alpha}_j K_{i,j} + \tilde{\alpha}_j^2\right), \end{aligned}$$

where we used $K_{i,i} = K_{j,j} = 1$. Consequently, the partial derivatives are given by

$$\begin{aligned} \frac{\partial W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j)}{\partial \tilde{\alpha}_i} &= 1 - \langle e_i, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_i - \tilde{\alpha}_j K_{i,j}\,, \\ \frac{\partial W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j)}{\partial \tilde{\alpha}_j} &= 1 - \langle e_j, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_j - \tilde{\alpha}_i K_{i,j}\,. \end{aligned}$$

In order to derive the maximum of $W_{i,j}$ on $[0, C_i] \times [0, C_j]$ from these derivatives, we need to consider three different cases.

The Case $K_{i,j} = 1$. By setting the above derivatives to zero, we obtain the following system of linear equations

$$\begin{aligned} \alpha_i^* + \alpha_j^* &= 1 - \langle e_i, K\alpha^{\setminus i,j} \rangle\,, \\ \alpha_i^* + \alpha_j^* &= 1 - \langle e_j, K\alpha^{\setminus i,j} \rangle \end{aligned}$$

that have to be satisfied for all global maxima $(\alpha_i^*, \alpha_j^*) \in \mathbb{R}^2$ of $W_{i,j}$. Now recall that we assumed that the kernel k is strictly positive definite, and therefore we see that $K_{i,j} = 1$ implies $x_i = x_j$, and hence $y_i = y_j$. From this we conclude $K_{i,\ell} = K_{j,\ell}$ for all $\ell = 1, \ldots, n$, and thus we obtain $1 - \langle e_i, K\alpha^{\setminus i,j} \rangle = 1 - \langle e_j, K\alpha^{\setminus i,j} \rangle$. Consequently, $W_{i,j}$ attains its global maximum at every point of the affine subspace

$$\left\{(\alpha_i^*, \alpha_j^*) : \alpha_i^* + \alpha_j^* = 1 - \langle e_i, K\alpha^{\setminus i,j} \rangle\right\}, \tag{11}$$

which is a translated version of the anti-diagonal subspace $\{(\alpha, -\alpha) : \alpha \in \mathbb{R}\}$.

Now recall that $y_i = y_j$ implies $C_i = C_j$, and hence we are actually interested in finding a pair $(\tilde{\alpha}_i, \tilde{\alpha}_j)$ that maximizes $W_{i,j}$ on the square $[0, C_i]^2$. If $1 - \langle e_i, K\alpha^{\setminus i,j} \rangle \in [0, 2C_i]$, it is easy to see that the subspace (11) intersects the square, and hence $W_{i,j}$ attains the desired maximum at every element in this intersection. In particular, (α_i^*, α_i^*), where

$$\alpha_i^* := \frac{1 - \langle e_i, K\alpha^{\setminus i,j} \rangle}{2}$$

is such a pair. Let us now assume that $1 - \langle e_i, K\alpha^{\setminus i,j} \rangle > 2C_i$. Then the subspace (11) lies "above" the square $[0, C_i]^2$, and since $W_{i,j}$ is concave, $W_{i,j}$ then attains its maximum over $[0, C_i]^2$ at a point

of the set of edges $\{C_i\} \times [0,C_i] \cup [0,C_i] \times \{C_i\}$. Let us fix a pair $(\tilde{\alpha}_i, \tilde{\alpha}_j) \in \{C_i\} \times [0,C_i]$. Then we have

$$\frac{\partial W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j)}{\partial \tilde{\alpha}_j} = 1 - \langle e_j, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_j - \tilde{\alpha}_i K_{i,j} = 1 - \langle e_j, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_j - C_i > 0\,,$$

and hence $W_{i,j}$ attains its maximum over $\{C_i\} \times [0,C_i]$ at the corner (C_i, C_i). Interchanging the roles of i and j we can thus conclude that $W_{i,j}$ attains its maximum over $[0,C_i]^2$ at (C_i, C_i). Since we can analogously show that, for $1 - \langle e_i, K\alpha^{\setminus i,j} \rangle < 0$, the function $W_{i,j}$ attains its maximum over $[0,C_i]^2$ at $(0,0)$, we finally find the update rule

$$\alpha_i^{new} := \alpha_j^{new} := \left[\frac{1 - \langle e_i, K\alpha^{\setminus i,j} \rangle}{2} \right]_0^{C_i} = \left[\frac{\nabla W_i(\alpha) + \alpha_i + \alpha_j}{2} \right]_0^{C_i}.$$

The Case $K_{i,j} = -1$. In this case, we have $x_i = x_j$, and hence $y_i = -y_j$. From this we conclude $K_{i,\ell} = -K_{j,\ell}$ for all $\ell = 1, \ldots, n$, and thus we obtain $\langle e_i, K\alpha^{\setminus i,j} \rangle = -\langle e_j, K\alpha^{\setminus i,j} \rangle$. Consequently, the derivatives above reduce to

$$\begin{aligned} \frac{\partial W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j)}{\partial \tilde{\alpha}_i} &= 1 - \langle e_i, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_i + \tilde{\alpha}_j\,, \\ \frac{\partial W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j)}{\partial \tilde{\alpha}_j} &= 1 + \langle e_i, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_j + \tilde{\alpha}_i\,, \end{aligned}$$

and from this it is easy to conclude that $W_{i,j}$ does not have a global maximum. However, a closer inspection of $W_{i,j}$ yields the formula

$$W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j) = \langle e, \alpha^{\setminus i,j} \rangle + \tilde{\alpha}_i + \tilde{\alpha}_j - \frac{1}{2}\langle \alpha^{\setminus i,j}, K\alpha^{\setminus i,j} \rangle - (\tilde{\alpha}_i - \tilde{\alpha}_j)\langle e_i, K\alpha^{\setminus i,j} \rangle - \frac{1}{2}\big(\tilde{\alpha}_i - \tilde{\alpha}_j\big)^2\,,$$

and hence we see that, for fixed $\beta \in \mathbb{R}$, we have

$$W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_i + \beta) = \langle e, \alpha^{\setminus i,j} \rangle + 2\tilde{\alpha}_i + \beta - \frac{1}{2}\langle \alpha^{\setminus i,j}, K\alpha^{\setminus i,j} \rangle + \beta\langle e_i, K\alpha^{\setminus i,j} \rangle - \frac{1}{2}\beta^2\,.$$

In other words, $W_{i,j}$ is a affine linear function with positive slope on the affine subspaces

$$\{(\tilde{\alpha}_i, \tilde{\alpha}_i + \beta) : \tilde{\alpha}_i \in \mathbb{R}\}\,, \qquad \beta \in \mathbb{R},$$

and therefore $W_{i,j}$ attains its maximum over $[0,C_i] \times [0,C_j]$ at a point from the set of edges $\{C_i\} \times [0,C_j] \cup [0,C_i] \times \{C_j\}$. Let us first consider a pair $(\tilde{\alpha}_i, \tilde{\alpha}_j) \in \{C_i\} \times [0,C_j]$. Then we have

$$\frac{\partial W_{i,j}(\tilde{\alpha}_i, \tilde{\alpha}_j)}{\partial \tilde{\alpha}_j} = 1 - \langle e_j, K\alpha^{\setminus i,j} \rangle - \tilde{\alpha}_j + C_i\,,$$

and hence $W_{i,j}$ attains its maximum over $\{C_i\} \times [0,C_j]$ at (C_i, α_j^*), where

$$\alpha_j^* = [1 - \langle e_j, K\alpha^{\setminus i,j} \rangle + C_i]_0^{C_j} = [\nabla W_j(\alpha) + \alpha_j - \alpha_i + C_i]_0^{C_j}\,.$$

Moreover, for $\delta_i := C_i - \alpha_i$ and $\delta_j := \alpha_j^* - \alpha_j$ we obtain the gain of this update by Lemma 2. Analogously, we can show that $W_{i,j}$ attains its maximum over $[0,C_i] \times \{C_j\}$ at (α_i^*, C_j), where

$$\alpha_i^* = [1 - \langle e_i, K\alpha^{\setminus i,j} \rangle + C_j]_0^{C_i} = [\nabla W_i(\alpha) + \alpha_i - \alpha_j + C_j]_0^{C_i}\,.$$

Again, the gain of the corresponding update can be computed by Lemma 2, and by comparing both gains we can then decide which two-dimensional update yields the larger gain. The corresponding update is chosen in the algorithm.

The Case $K_{i,j} \neq \pm 1$. To solve the two dimensional problem in this case we fix an $\alpha \in \mathbb{R}^n$ and write

$$\begin{aligned}\gamma_i &:= 1 - \langle e_i, K\alpha^{\setminus i,j}\rangle = 1 - \sum_{\ell \neq i,j} \alpha_\ell K_{i,\ell} = \nabla W_i(\alpha) + \alpha_i + \alpha_j K_{i,j}\,,\\ \gamma_j &:= 1 - \langle e_j, K\alpha^{\setminus i,j}\rangle = 1 - \sum_{\ell \neq i,j} \alpha_\ell K_{j,\ell} = \nabla W_j(\alpha) + \alpha_j + \alpha_i K_{i,j}\,.\end{aligned}$$

Using the derivatives of $W_{i,j}$ it is then easy to see that $W_{i,j}$ attains its global maximum at each point (α_i^*, α_j^*) that satisfies $\gamma_i = \alpha_i^* + \alpha_j^* K_{i,j}$ and $\gamma_j = \alpha_j^* + \alpha_i^* K_{i,j}$. Furthermore, simple algebraic transformations show

$$\alpha_i^* = \frac{\gamma_i - \gamma_j K_{i,j}}{1 - K_{i,j}^2} \qquad \text{and} \qquad \alpha_j^* = \frac{\gamma_j - \gamma_i K_{i,j}}{1 - K_{i,j}^2}\,,$$

and by re-substituting the definition of γ_i and γ_j we hence obtain

$$\begin{aligned}\alpha_i^* &= \alpha_i + \tfrac{\nabla W_i(\alpha) - \nabla W_j(\alpha) K_{i,j}}{1 - K_{i,j}^2},\\ \alpha_j^* &= \alpha_j + \tfrac{\nabla W_j(\alpha) - \nabla W_i(\alpha) K_{i,j}}{1 - K_{i,j}^2}\end{aligned} \tag{12}$$

for the uniquely determined point at which $W_{i,j}$ attains its global maximum. Now if $(\alpha_i^*, \alpha_j^*) \in [0, C_i] \times [0, C_j]$ we can simply update by $(\alpha_i^{new}, \alpha_j^{new}) := (\alpha_i^*, \alpha_j^*)$. However, if $(\alpha_i^*, \alpha_j^*) \notin [0, C_i] \times [0, C_j]$ we have to make further calculations. For example, for $\alpha_i^* > C_i$ and $\alpha_j^* \in [0, C_j]$, the function $W_{i,j}$ attains its maximum over $[0, C_i] \times [0, C_j]$ at a point of the line $\{C_i\} \times [0, C_j]$ by the concavity of $W_{i,j}$. Consequently, in this case the update is

$$(\alpha_i^{new}, \alpha_j^{new}) := \left(C_i, [\nabla W_j(\alpha) + (\alpha_i - C_i) K_{i,j} + \alpha_j]_0^{C_j}\right),$$

that is, we first update the i-th coordinate, which leads to the temporary gradient

$$\nabla W_j(\alpha) + (\alpha_i - C_i) K_{i,j}\,,$$

and then perform a one-dimensional optimization over the j-th coordinate. The remaining three cases where exactly one direction of (α_i^*, α_j^*) violates the box constraint can be handled analogously. Finally, let us consider the cases, where both coordinates violate the constraint, for example, $\alpha_i^* > C_i$ and $\alpha_j^* > C_j$. In this case, the concavity of $W_{i,j}$ shows that $W_{i,j}$ attains its maximum over $[0, C_i] \times [0, C_j]$ at a point of the set $\{C_i\} \times [0, C_j] \cup [0, C_i] \times \{C_j\}$. Consequently, we have to temporarily perform the one-dimensional optimization above twice, namely one over the i-th coordinate and one over the j-th coordinate. By computing the resulting gain of W for both optimizations, we can then decide which optimization we have to choose for the update. Again, the remaining three cases can be handled analogously.

Algorithm 2 2D-SVM solver

initialize $\big(\alpha, \nabla W(\alpha), T(\alpha), S(\alpha)\big)$
while $S(\alpha) > \frac{\varepsilon}{2\lambda}$ **do**
  select directions i^* and j^*
  update α in the directions i^* and j^*
  update $\nabla W(\alpha)$ in the directions i^* and j^* and calculate $T(\alpha)$ and $E(\alpha)$
  $S(\alpha) \leftarrow T(\alpha) + E(\alpha)$
end while

3.2 Selecting a Working Set of Size Two

The 2D-SVM-solver displayed in Algorithm 2 is conceptionally very similar to the 1D-SVM-solver presented in Algorithm 1. However, so far we have not addressed how to choose the directions i^* and j^* in which the 2D-SVM-solver performs an update. Obviously, several possibilities exists for this task, and we discuss a few of them in the following.

WSS 0: Choose The Pair Of Directions With Maximal Gain. Given a pair of directions (i, j), Lemma 2 can be used to compute the gain of W resulting from the exact two dimensional optimization described in Section 3.1. Now one could consider all pairs of directions and choose the one with the largest gain. Of course, in practice this approach is prohibitive, since the search is an $O(n^2)$ operation, which has to be performed in each iteration. Nonetheless, in some sense this approach may be viewed as an "optimal" two dimensional strategy, and all subset selection strategies developed below can be interpreted as low cost approximations to this approach. Consequently, we tested it to get a baseline number of iterations, to which all other subset selection strategies are compared to.

WSS 1: 1D-direction With Maximal Gain And Previously Found 1D-direction. A careful analysis of the behavior of the 1D-SVM-solver shows that it often comes into a regime in which it picks alternating indices i^* and j^* for a while. In other words, it tries to approximately solve the 2D-problem in the directions i^* and j^*. In order to avoid this cost-intensive alternating we can look for the best 1D-direction i^* and then perform a 2D-update over i^* and the 1D-direction i^*_{old} chosen in the previous iteration. Conceptionally, this approach is very close to the maximum-gain procedure mentioned in Glasmachers and Igel (2006) for SVMs with offset. The advantage of this approach is that it preserves the low-cost search from the 1D-SVM-solver. On the downside, however, it may not reduce the number of iterations very effectively.

WSS 2: Two 1D-directions With Maximal Gain From Separate Subsets. Another simple way to preserve the low cost search from the 1D-SVM-solver is to split the index set $\{1, \ldots, n\}$ into two parts $\{1, \ldots, n/2\}$ and $\{n/2+1, \ldots, n\}$ and search for the 1D-directions with maximal gain over these two parts separately. In other words, we can choose the directions i^* and j^* by

$$\begin{aligned} i^* &\in \arg\max_{i \le n/2} W(\alpha + \delta_i e_i) - W(\alpha)\,, \\ j^* &\in \arg\max_{i > n/2} W(\alpha + \delta_i e_i) - W(\alpha)\,, \end{aligned}$$

where δ_i is defined as in the 1D-SVM-solver. Clearly, this approach preserves the low cost search from the 1D-SVM-solver, but again it is not clear whether it reduces the number of iterations very effectively.

WSS 4: 1D-direction With Maximal Gain And A Direction Of A Nearby Sample. Yet another approach to preserve the low cost search from the 1D-SVM-solver is to first look for the 1D-direction i^* with maximal gain, and then, in a second step, to pick a direction j^* such that x_{j^*} is close to x_{i^*} with respect to the metric

$$d_k(x,x') := \sqrt{2-2k(x,x')}, \qquad x,x' \in X,$$

induced by the kernel. Note that x is close to x' in this metric, if and only if $k(x,x')$ is close to 1. Consequently, the gradients of the samples close to x_{i^*} are the ones that are most affected by an update in direction i^*. Therefore, if these gradients are close to zero *before* the update, they will most likely be no longer close to zero *after* the update, and hence the corresponding directions will have a good chance of being chosen in a subsequent iteration. In our experiments, we considered the k-nearest neighbors of x_i^*, where $k = 10$, and picked the neighbor x_{j^*} for which the 2D-update in the directions (i^*, j^*) yielded the largest gain. Note that, as soon as the direction i^* is found, it is clear that one subsequently needs to access the i^*-th kernel row for updating the gradient. Therefore, this working set selection strategy does not require further kernel computations. Moreover, computing the 2D-gain over k candidates is also relatively inexpensive, if k remains small. Nonetheless, initial experiments suggested that searching over the k-nearest neighbors only makes sense when the solver mainly updates *inner* support vectors, that is, directions i with $0 < \alpha_i < C_i$. Consequently, we implemented a Boolean flag that was recomputed every 10 iterations. In this re-computation, the flag was set to true, if and only if in at least 5 of the previous 10 iterations the picked directions i^* and j^* both were inner support vectors. We then considered the k-nearest neighbors only if this Boolean flag was set, while in the other case we applied the working set selection strategy WSS 1.

WSS x: Combinations Of 1D-direction-based Approaches. It is easy to see that one can combine the previous three methods that are based on finding the 1D-direction with maximal gain. For example, in each iteration one can combine WSS 1 and WSS 2 by computing the 2D-gain of both methods and pick the one with the larger gain. Obviously, this still preserves the low cost search from the 1D-SVM-solver and only adds little cost for computing the 2D-gain for the two candidate pairs. Similarly, all three methods can be combined. Combinations of these methods are called WSS x, where x is the sum of the combined methods. For example, by combining WSS 1, WSS 2, and WSS 4 we obtain WSS 7, and by combining WSS 1, WSS 2, WSS 4 with WSS 512 below, we obtain WSS 519 . In the following, we keep this binary numbering system which makes it possible to easily describe arbitrary combinations of basic working set selection strategies.

WSS 8: 1D-direction With Maximal Gain And One-step-ahead 1D-direction. Another way to extend the 1D-SVM subset selection strategy to two directions is to first look for the 1D-direction i^* with maximal gain, and then to look for the 1D-direction j^* with maximal gain that would be found after having updated in direction i^*. Obviously, this strategy, which we call WSS 8, is closely related to WSS 1 in that the update and search routines are partially permuted. However, it has a higher cost for the search part per iteration, while intuitively it should reduce the number of iterations.

WSS 16: Maximal Violating Pair. A completely different subset selection strategy is based on the maximal violating pair (MVP) idea, see Keerthi et al. (2001) and Joachims (1999). For the SVM without offset, this means that the pair (i^*, j^*) is chosen that violates (5) most. In other words, for both index sets $\{i : \alpha_i < C_i\}$ and $\{i : \alpha_i > 0\}$ the two indices with the largest, respectively smallest, gradients are picked, and the final pair (i^*, j^*) consists of the indices that have the gradient with the largest absolute value among the four candidate directions. In order to implement this working set selection strategy efficiently, the sets $\{i : \alpha_i < C_i\}$ and $\{i : \alpha_i > 0\}$ should be kept in memory and

updated in every iteration. This may add some cost per iteration compared to the previous working set selection strategies, while it is unclear how the number of iterations behave compared to these strategies.

WSS 32: 1D-direction With Maximal Gain And Corresponding "Optimal" 2D-direction. None of the methods introduced so far try to seriously approximate the 2D-subset selection strategy WSS 0, which intuitively picks the best possible pair of indices. The first method that seriously strives for such an approximation is WSS 32, which first picks the 1D-direction i^* with maximal gain, and then searches for the $j^* \in \{1,\ldots,n\}$ such that (i^*, j^*) maximizes the corresponding 2D-gain. Obviously, the cost for this search method is significantly higher than those of WSS 1 to WSS 7, but it is still $O(n)$. On the other hand, the better choice of (i^*, j^*) may substantially reduce the number of iterations of the 2D-SVM-solver, and hence it is not a-priori clear how WSS 32 performs compared to the earlier methods. Finally, note that WSS 32 is related to the second order working set selection strategy of Fan et al. (2005), which was proposed for SVMs with offset.

WSS 64: 1D-direction With Maximal Gain And Random "Optimal" 2D-direction. Instead of considering all pairs (i^*, j), $j = 1,\ldots,n$, as WSS 32 does, it may suffice to reduce the search over the pairs (i^*, j), $j \in J$, where $J \subset \{1,\ldots,n\}$ is a random subset. In our experiments we considered the case $\#J = n/5$.

WSS 128: 1D-direction With Maximal Gain And Approximately "Optimal" 2D-direction. One of the disadvantages of WSS 32 is that computing the 2D-gain is quite expensive due to the relatively large number of branches and floating point operations. One way to address this issue is to compute the 2D-gain in WSS 32 only approximately. WSS 128 uses the following approximation: for indices i and j with $K_{i,j} = \pm 1$ it computes the exact gain, while for the other pairs it first computes α_i^* and α_j^* by (12), and then applies the simple clipping operation

$$\begin{aligned} \alpha_i^{new} &:= [\alpha_i^*]_0^{C_i}, \\ \alpha_j^{new} &:= [\alpha_j^*]_0^{C_j}. \end{aligned}$$

For these new α's, WSS 128 finally computes the gain by Lemma 2. Clearly, this gain is in general less than the exact gain, but it still may be a good approximation. In particular, if both α_i^* and α_j^* satisfy the box constraints, then the approximation is actually exact. On the other hand, the approximation is clearly less expensive, but we expect more iterations compared to WSS 32.

WSS 256: Random 2D-directions With Maximal Gain. Another way to approximate WSS 0 is to consider k random pairs (i, j), and pick the pair (i^*, j) that yields the largest exact 2D-gain among them. In WSS 256 we followed this idea for $k := n$.

WSS 512: 1D-direction With Maximal Gain And 2D-direction Over Inner SVs. Although the approximate computation of the 2D-gain in WSS 128 is cheaper than the exact computation in WSS 32, it may still be too expensive. One way to further decrease these costs is based on the observation that the 2D-gain is given by

$$\frac{1}{2} \cdot \frac{|\nabla W_i(\alpha)|^2 + |\nabla W_j(\alpha)|^2 - 2\nabla W_i(\alpha)\nabla W_j(\alpha) K_{i,j}}{1 - K_{i,j}^2}$$

if $K_{i,j} \neq \pm 1$ and α_i^* and α_j^* computed by (12) satisfy the box constraints. WSS 512 uses this simplified formula in the following way. Again, it first searches for the 1D-direction i^* with maximal gain. If α_{i^*} is an inner support vector, see WSS 4 for a definition, and the Boolean flag of WSS 4 is

set, WSS 512 searches for the direction

$$j^* \in \{j : 0 < \alpha_j < C_j \text{ and } K_{i^*,j} \neq \pm 1\}$$

that optimizes the above formula of the 2D-gain for fixed $i := i^*$. Since in some iterations WSS 512 reduces to the 1D-SVM-solver we further considered some combinations with WSS 3, and WSS 7 in our experiments. Following the naming convention of combinations mentioned earlier, these strategies are called WSS 515 and WSS 519.

WSS 1024: 1D-direction With Maximal Gain And Random 2D-direction Over Inner SVs. The next subset selection strategy, WSS 1024, is quite similar to WSS 512, except that it does not consider all inner support vectors in the search for j^*, but only k random inner support vectors. In our experiments we used the k that equaled 20% of the current number of inner support vectors. In addition, we initiated the search whenever α_{i^*} was an inner support vector, that is, the search was initiated *independently* of the Boolean flag of WSS 4. Again, in some iterations WSS 1024 reduces to the 1D-SVM-solver, and hence we further considered some combinations with WSS 1, WSS 2, and WSS 4, where again the naming convention above was used.

WSS 2048: Add Random 2D-directions Over Inner SVs. The final subset selection strategy, WSS 2048, is actually not a subset selection strategy of its own, but only a strategy that works in combination with others. Once one of the previous subset selection strategies has picked a pair (i^*, j^*) and α_{i^*} has turned out to be an inner support vector, WSS 2048 considers k random pairs of inner support vectors, and picks the pair (i^{**}, j^{**}) that has largest approximate gain, where the approximation was computed as in WSS 512. Then the exact gain of (i^*, j^*) and (i^{**}, j^{**}) is computed and the pair with the larger exact gain was chosen. We considered this method in combination with WSS 1, WSS 2, and WSS 4, where again the naming convention above was used.

4. Convergence Analysis

In this section we establish an upper bound on the number of iterations for both the 1D-SVM and the 2D-SVM. Our approach is heavily based on earlier ideas[2] developed for the analysis of rate-certifying decomposition algorithms, see, for example, Hush and Scovel (2003), List and Simon (2005), Hush et al. (2006) and List and Simon (2007), but it may be possible to partially use results on block coordinate descent algorithms such as the one by Luo and Tseng (1992) for the analysis, instead.[3]

Let us begin by recalling from the first papers mentioned that the σ-functional for a vector $\alpha \in [0,C] = [0,C_1] \times \cdots \times [0,C_n]$ and an index set $I \subset \{1,\ldots,n\}$ is defined by

$$\sigma(\alpha|I) := \sup_{\substack{\tilde{\alpha}\in[0,C] \\ \tilde{\alpha}_i=\alpha_i \forall i \notin I}} \left\langle \nabla W(\alpha), \tilde{\alpha} - \alpha \right\rangle .$$

2. Despite this, we decided to include the analysis, since: *a)* it still requires a little work and thus we felt that it was a bit unfair to the reader to simply say that the analysis is straightforward; *b)* we thought that it was nice to see how the relatively complicated techniques for the offset case significantly simplify; *c)* our goal was to provide a full and self-contained work for the proposed algorithm.

3. Note, however, that their results only control the convergence to a *dual* optimal solution, while for statistical reasons, we are actually interested in the convergence control of the corresponding primal sequence. Consequently, their results are at least not directly applicable.

Since our algorithms are based on gain optimization rather than rate certification, we further need the γ-functional

$$\gamma(\alpha|I) := \sup_{\substack{\tilde{\alpha}\in[0,C] \\ \tilde{\alpha}_i=\alpha_i \forall i\notin I}} W(\tilde{\alpha}) - W(\alpha)\,,$$

which expresses the gain in the dual objective function resulting from an optimization over the directions contained in I. To simplify notations, we write $\sigma(\alpha|i) := \sigma(\alpha|\{i\})$ and $\gamma(\alpha|i) := \gamma(\alpha|\{i\})$. Note that we have

$$\sigma(\alpha|i) = \sup_{\tilde{\alpha}_i\in[0,C_i]} (\tilde{\alpha}_i - \alpha_i)\nabla W_i(\alpha)\,,$$

while $\gamma(\alpha|i)$ expresses the gain

$$W\bigl(\alpha + (\alpha_i^{new} - \alpha_i)e_i\bigr) - W(\alpha)$$

of the 1D-update in direction i, where α_i^{new} is defined by (4). In addition, $\gamma(\alpha|\{i,j\})$ is the gain obtained by the update discussed in Section 3.1. Moreover, for $I = \{1,\ldots,n\}$ we write $\sigma(\alpha) := \sigma(\alpha|I)$ and $\gamma(\alpha) := \gamma(\alpha|I)$, respectively. Note that both σ and γ are monotonic in I, that is, for $I \subset J$ we have $\sigma(\alpha|I) \le \sigma(\alpha|J)$ and $\gamma(\alpha|I) \le \gamma(\alpha|J)$. Finally, we need the obvious relation

$$\gamma(\alpha) = W(\alpha^*) - W(\alpha)\,,$$

where we recall from Section 2 that $\alpha^* \in [0,C]$ denotes a solution of the dual problem (3). In other words, $\gamma(\alpha)$ expresses the dual sub-optimality of α.

Let us now begin our analysis by presenting two lemmata that establish relationships between these quantities.

Lemma 3 *For all $\alpha \in [0,C]$ we have*

$$\sum_{i=1}^{n} \sigma(\alpha|i) = \sigma(\alpha) = \mathrm{gap}(\alpha)\,,$$

where $\mathrm{gap}(\alpha)$ *denotes the duality gap defined in (7). In particular, there exists an index $i^\star \in \{1,\ldots,n\}$ such that*

$$\sigma(\alpha|i^\star) \ge n^{-1}\sigma(\alpha)\,.$$

This lemma can be easily derived from results in List et al. (2007) and List and Simon (2007). However, in the case of SVMs without offset, its proof is very elementary and hence we present it here for the sake of completeness.

Proof For $i \in \{1,\ldots,n\}$ it is easy to see that the supremum used to define $\sigma(\alpha|i)$ is attained at

$$\alpha_i := \begin{cases} C_i & \text{if } \nabla W_i(\alpha) \ge 0 \\ 0 & \text{if } \nabla W_i(\alpha) < 0\,. \end{cases} \tag{13}$$

Moreover, the vector $\alpha := (\alpha_1,\ldots,\alpha_n) \in [0,C]$ realizes the supremum defining $\sigma(\alpha)$, and hence we obtain

$$\sum_{i=1}^{n} \sigma(\alpha|i) = \sum_{i=1}^{n} \bigl\langle \nabla W(\alpha), (\alpha_i - \alpha_i)e_i \bigr\rangle = \langle \nabla W(\alpha), \alpha - \alpha\rangle = \sigma(\alpha)\,.$$

Furthermore, we have

$$\begin{aligned}\sigma(\alpha) = \langle \nabla W(\alpha), \alpha - \alpha \rangle &= \langle \alpha, K\alpha \rangle - \langle e, \alpha \rangle + \sum_{i=1}^{n} \alpha_i \cdot \nabla W_i(\alpha) \\ &= \langle \alpha, K\alpha \rangle - \langle e, \alpha \rangle + \sum_{i=1}^{n} C_i \left[\nabla W_i(\alpha)\right]_0^\infty ,\end{aligned}$$

and therefore we have shown $\sigma(\alpha) = \mathrm{gap}(\alpha)$. The last assertion is a trivial consequence of the first assertion. ■

The second lemma relates $\sigma(\alpha|I)$ to the gain $\gamma(\alpha|I)$. For its formulation we need the quantity $B_{\max} := \max_{i=1,\ldots,n} C_i$.

Lemma 4 *For all $\alpha \in [0,C]$ and $I \subset \{1,\ldots,n\}$ we have*

$$\sigma(\alpha|I) \;\geq\; \gamma(\alpha|I) \;\geq\; \frac{\sigma(\alpha|I)}{2} \min\left\{1, \frac{\sigma(\alpha|I)}{|I|^2 B_{\max}^2}\right\},$$

where $|I|$ denotes the cardinality of I.

In a slightly different form, this lemma has been established in, for example, Hush et al. (2006), and it was somewhat implicitly used in List and Simon (2007). Again, we present its proof for the sake of completeness.

Proof Let α_i be defined by (13) and $d := \sum_{i \in I} (\alpha_i - \alpha_i) e_i$. For $\lambda \in [0,1]$, we then have $\alpha + \lambda d \in [0,C]$, and a calculation analogous to the one in the proof of Lemma 1 yields

$$\gamma(\alpha|I) \geq W(\alpha + \lambda d) - W(\alpha) = \lambda \big\langle \nabla W(\alpha), d \big\rangle - \frac{\lambda^2}{2} \langle d, Kd \rangle \geq \lambda \sigma(\alpha|I) - \frac{\lambda^2 |I|^2 B_{\max}^2}{2} .$$

Now the right hand side is maximized at

$$\lambda^* := \begin{cases} 1 & \text{if } \sigma(\alpha|I) > |I|^2 B_{\max}^2 \\ |I|^{-2} B_{\max}^{-2} \sigma(\alpha|I) & \text{if } \sigma(\alpha|I) \leq |I|^2 B_{\max}^2 . \end{cases}$$

In the case $\sigma(\alpha|I) > |I|^2 B_{\max}^2$ we hence find

$$\gamma(\alpha|I) \geq \sigma(\alpha|I) - \frac{|I|^2 B_{\max}^2}{2} > \frac{\sigma(\alpha|I)}{2} ,$$

while in the other case $\sigma(\alpha|I) \leq |I|^2 B_{\max}^2$ we obtain

$$\gamma(\alpha|I) \geq \frac{\sigma^2(\alpha|I)}{2|I|^2 B_{\max}^2} .$$

Combining all estimates we then obtain the inequality on the right hand side.

To show the inequality on the left hand side we fix an $\tilde{\alpha} \in [0,C]$ such that $\tilde{\alpha}_i = \alpha_i$ for all $i \notin I$. Then we have

$$W(\tilde{\alpha}) - W(\alpha) = \langle \nabla W(\alpha), \tilde{\alpha} - \alpha \rangle - \frac{1}{2} \langle \tilde{\alpha} - \alpha, K(\tilde{\alpha} - \alpha) \rangle \leq \langle \nabla W(\alpha), \tilde{\alpha} - \alpha \rangle \leq \sigma(\alpha|I) ,$$

and by maximizing the left hand side of this inequality over $\tilde{\alpha}$ we find $\gamma(\alpha|I) \leq \sigma(\alpha|I)$. ■

With these preparations we can now present a preliminary result on iterative algorithms that have a certain control of their gain.

Proposition 5 *Let $\alpha^{(0)}, \alpha^{(1)}, \cdots \in [0,C]$ be a sequence of feasible vectors that satisfies*

$$W(\alpha^{(\ell+1)}) - W(\alpha^{(\ell)}) \geq \gamma(\alpha^{(\ell)}|i_\ell^\star)\,, \qquad \ell \geq 0, \tag{14}$$

where for each ℓ the index $i_\ell^\star \in \{1,\ldots,n\}$ is the one described in Lemma 3, that is, it satisfies $\sigma(\alpha^{(\ell)}|i_\ell^\star) \geq n^{-1}\sigma(\alpha^{(\ell)})$. Then for all $\ell \geq 1$ we have

$$\gamma(\alpha^{(\ell+1)}) \leq \gamma(\alpha^{(\ell)})\left(1 - \frac{1}{2n}\min\left\{1, \frac{\gamma(\alpha^{(\ell)})}{nB_{\max}^2}\right\}\right).$$

Moreover, for all $\varepsilon > 0$ and all $\ell \geq \ell_\varepsilon$ we have $\gamma(\alpha^{(\ell)}) \leq \varepsilon$, where

$$\ell_\varepsilon := \left\lceil \frac{2n^2 B_{\max}^2}{\varepsilon} \right\rceil + \max\left\{0, \left\lceil 2n \ln \frac{W(\alpha^*) - W(\alpha^{(0)})}{\varepsilon} \right\rceil\right\}.$$

Proof By Lemmas 4 and 3 we find

$$\begin{aligned}
\gamma(\alpha^{(\ell)}) - \gamma(\alpha^{(\ell+1)}) = W(\alpha^{(\ell+1)}) - W(\alpha^{(\ell)}) &\geq \gamma(\alpha^{(\ell)}|i_\ell^\star) \\
&\geq \frac{\sigma(\alpha^{(\ell)}|i_\ell^\star)}{2}\min\left\{1, \frac{\sigma(\alpha^{(\ell)}|i_\ell^\star)}{B_{\max}^2}\right\} \\
&\geq \frac{\sigma(\alpha^{(\ell)})}{2n}\min\left\{1, \frac{\sigma(\alpha^{(\ell)})}{nB_{\max}^2}\right\} \\
&\geq \frac{\gamma(\alpha^{(\ell)})}{2n}\min\left\{1, \frac{\gamma(\alpha^{(\ell)})}{nB_{\max}^2}\right\}.
\end{aligned}$$

From this we easily obtain the first assertion.

The second assertion has already been shown in the second part of the proof of the first assertion of (List and Simon, 2007, Theorem 4), which can be found on the pages 312 and 313 of List and Simon (2007). ■

Note that $1/n$-rate certifying algorithms considered in List and Simon (2007) clearly satisfy assumption (14). Moreover, Proposition 5 can also be applied to the 1D-SVM and 2D-SVM:

Theorem 6 *Consider the* 1D-SVM *described in Algorithm 1 or a* 2D-SVM *in the sense of Algorithm 2 that uses a working set selection strategy whose gain at each iteration is not less than that of the* 1D-SVM. *Furthermore, assume that* $\max\{w_{\text{neg}}, w_{\text{pos}}\} \leq 1$. *Then for all $\varepsilon > 0$, $n \geq 1$, and all $\lambda > 0$ these algorithms terminate after at most*

$$\left\lceil \frac{2}{\lambda\varepsilon\min\{1, 2\lambda\varepsilon\}} \right\rceil + \max\left\{0, \left\lceil 2n \ln \frac{4\lambda(W(\alpha^*) - W(\alpha^{(0)}))}{\varepsilon\min\{1, 2\lambda\varepsilon\}} \right\rceil\right\}$$

iterations. In particular, in the most likely scenario $2\lambda\varepsilon \leq 1$ *these algorithms do not need more iterations than*

$$\left\lceil \frac{1}{\lambda^2\varepsilon^2} \right\rceil + \max\left\{0, \left\lceil 2n \ln \frac{2(W(\alpha^*) - W(\alpha^{(0)}))}{\varepsilon^2} \right\rceil \right\}.$$

Proof The 1D-SVM chooses at each iteration ℓ a direction i_ℓ^* that maximizes the 1D-gain $\gamma(\alpha^{(\ell)}|i)$. Consequently, we have

$$W(\alpha^{(\ell+1)}) - W(\alpha^{(\ell)}) = \gamma(\alpha^{(\ell)}|i_\ell^*) \geq \gamma(\alpha^{(\ell)}|i_\ell^\star),$$

where $i_\ell^\star$ is the direction described in Lemma 3. In other words, (14) is satisfied for this algorithm, and from this it is not hard to see that the considered 2D-SVM's also satisfy assumption (14). Let us now define

$$h(\sigma) := \frac{\sigma}{2} \min\left\{1, \frac{\sigma}{n^2 B_{\max}^2}\right\}, \qquad \sigma > 0.$$

For $\varepsilon := h(\frac{\varepsilon}{2\lambda})$ Proposition 5 together with Lemma 4 then shows that

$$h\Big(\frac{\varepsilon}{2\lambda}\Big) = \varepsilon \geq \gamma(\alpha^{(\ell)}) \geq h\big(\sigma(\alpha^{(\ell)})\big)$$

for all $\ell \geq \ell_\varepsilon$ and hence we obtain $S(\alpha^{(\ell)}) \leq \mathrm{gap}(\alpha^{(\ell)}) = \sigma(\alpha^{(\ell)}) \leq \frac{\varepsilon}{2\lambda}$ by the monotonicity of the function h. Using $B_{\max} \leq \frac{1}{2\lambda n}$ we then obtain the assertion by simple algebraic transformations. ■

Note that the working set selection strategies WSS 1, WSS 2, WSS 4, WSS 8, WSS 32, WSS 64, WSS 128, WSS 512, and WSS 1024, satisfy the assumptions of Theorem 6. Moreover, the same is true for all combinations of working set selection strategies that include at least one of the strategies listed. Finally, note that the upper bound established in Proposition 5 coincide (modulo constants that come from different working set sizes) with the bounds for rate certifying algorithms presented in List and Simon (2005), Hush et al. (2006) and List and Simon (2007). Moreover, the step from dual ε-optimality to primal ε-optimality considered in the proof of Theorem 6 coincides with the analysis (List et al., 2007) for SVMs with offset. Consequently, the bound presented in Theorem 6 equals the best known guarantees for solvers for SVMs with offset.

5. Experiments

The described 1D-SVM-solver and 2D-SVM-solver enjoy nice theoretical properties with respect to both generalization performance and required training time. However, it is unclear how tight these bounds are, so it remains unclear whether the proposed SVMs also perform well in practice. Therefore, we performed several experiments that address the following questions:

1. Which subset selection strategies lead to the smallest number of iterations or the shortest run time? How many more iterations than WSS 0 do these strategies perform?

2. How many less iterations needs the stopping criterion (9) compared to standard duality gap (7) and is there also an advantage in terms of run time?

3. How much more efficient is the 2D-SVM-solver than the technically much easier 1D-SVM-solver?

4. How well does the 2D-SVM-solver work compared to standard software packages such as LIBSVM by Chang and Lin (2009)?

5. What is the advantage of warm start initializations when the parameter search is performed over a grid?

To answer these questions we implemented the 1D-SVM- and the 2D-SVM-solver in C++, and downloaded LIBSVM version 2.82 by Chang and Lin (2009). The algorithms were compiled by LINUX's gcc version 4.3 with various software and hardware optimizations enabled. All experiments were conducted on a computer with INTEL XEON X5355 (2.66 GHz) quad core processor and 8GB RAM under a 64bit version of RedHat Linux Enterprise 4. During all experiments that incorporated measurements of run time, one core was used solely for the experiments, and the number of other processes running on the system was minimized. The run time itself was measured by the C function clock() from the library time.h. The resulting resolution was 0.01 seconds.

In some preliminary experiments we made a couple of observations that changed the described implementation strategy slightly: First, it turned out that the auto-vectorization of gcc only gave mediocre and sometimes even contradicting results, even if the implementation guidelines of gcc 4.3's auto-vectorization were strictly followed. Therefore, we decided to manually code SSE2-vectorized versions of the most important routines, namely: computing kernel values, searching for the optimal 1D-direction, updating the gradient, and computing the weighted sum $E(\alpha)$ of clipped slack variables. To this end, we used the library emmintrin.h together with properly aligned arrays of doubles.[4] Some of our preliminary experiments not reported here indicated that this specialized hardware instruction set yields a run time improvement by a factor between 1.3 and 1.8 depending on the working set selection strategy and the data set. Second, the initial experiments suggested substantial numerical instabilities on a few data sets when using single floats, so we decided to use double precision throughout the experiments. Third, we were rather disappointed by the run time behavior of LIBSVM, even when we enabled its shrinking heuristic.[5] After some investigations we found that the main reason for the disappointing run time performance was the fact that LIBSVM copies kernel rows into the kernel cache, if one uses pre-computed kernel matrices, which, as discussed below, we did throughout the experiments. This copying mechanism results in a small number of iterations per second when the LIBSVM-solver is started on a new parameter point, while with the kernel cache being filled up during the optimization, the solver starts performing more iterations per second. To ensure a fair comparison, we thus decided to implement our own version of LIBSVM's solver (without shrinking strategy). As a side effect, this new implementation also benefited from the SSE2 instructions for upgrading the gradient. Unlike the subset selection strategy of the 1D-SVM-solver, however, LIBSVM's subset selection strategy, though implementable, does

4. At first glance, this manual approach may seem to be too specialized, since it should clearly be not the goal of this paper to fine-tune an algorithm to a very specific hardware environment. On the other hand, a good compiler should make optimizations with respect to these nowadays standard instructions, which have been first introduced by Intel in 2001 and have been adopted by AMD in 2003, automatically. Unfortunately, it turned out that gcc 4.3 did not do this optimization reliably. Namely, depending on some minor and apparently independent changes in other parts of the code, the most crucial loops where sometimes optimized and sometimes not. This behavior rendered a reasonable comparison of different algorithms impossible. Therefore, our manual approach can also be viewed as a compilation with a more ideal compiler, which in the future is hopefully available.
5. In fact, it turned out that neither the number of iterations nor the run time was significantly affected by the shrinking heuristic. Corresponding results for the run time are reported in Figure 1.

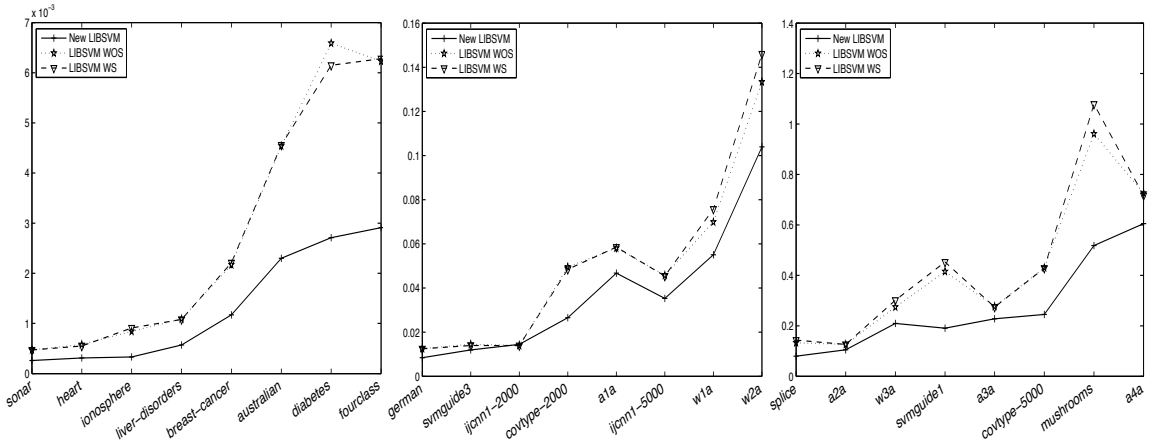


Figure 1: Performance of the original LIBSVM-solver (WOS: without shrinking; WS: with shrinking) compared to our own implementation of the LIBSVM-solver. The graphic displays the average run time in seconds (middle) over the 10 by 10 parameter grid described later in this section. Shrinking does not give an advantage on *this* parameter grid, while the new LIBSVM implementation runs in almost all cases significantly faster than the original LIBSVM .

not benefit from vectorization since not all indices are considered, and hence the relatively slow non-serial RAM access of the CPU outweighs the speed improvement of the SSE2 instructions.

We downloaded all data sets for binary classification from LIBSVM's homepage whose number of features did not exceed 1000. We made this cut because having data sets with a huge number of features would have required substantial extra effort for implementing our algorithms, and this effort was clearly out of the scope of this paper. In all cases, we used the scaled versions of these data sets, and if they were not available, we scaled the unscaled data sets with the help of LIBSVM's scaling tool. For data sets that were not split into a training and test set we generated a random split that contained approximately 70% training and 30% test samples. Moreover, for the already split data sets SPLICE, SVMGUIDE1, SVMGUIDE3, we decided to first merge the corresponding training and test set and then generate the random split above. For the large data sets COVTYPE and IJCNN1, we generated random subsets of the two data sets of sizes $n = 2000, 5000$, and then applied the random split above. Finally, we ignored some versions with larger training set of the AXA and WXA families, namely A5A–A9A, and W4A–W8A because of time and memory constraints. Moreover, for these two families of data sets we kept the split between training and test sets. Table 1 shows the corresponding characteristics of the considered data sets together with classification errors of the fastest version of the 2D-SVM and LIBSVM, respectively.

In all our experiments, we considered k-fold cross validation with folds randomly generated from the training set and hyper-parameters λ and σ each taken from a 10 by 10 grid. Since the choice of this grid has a significant influence on both the training time and the learning performance, special care is needed here. Despite such care, however, it seems likely that every choice will be subject to discussion. To pick the parameter grid less heuristically than in previous investigations, we decided to use recent statistical insights from Steinwart et al. (2007), which show that *asymptotically* good

	training size	test size	dimension	LIBSVM	2D-SVM	2D-SVM (duality gap)	2D-SVM (fine grid)
SONAR	146	62	60	12.68 $\pm$ 4.27	12.80 $\pm$ 4.04	12.62 $\pm$ 3.95	13.21 $\pm$ 4.19
HEART	188	82	13	17.58 $\pm$ 3.80	17.42 $\pm$ 4.39	17.47 $\pm$ 3.86	17.94 $\pm$ 4.21
LIVER-DISORDERS	248	97	6	29.31 $\pm$ 4.00	29.76 $\pm$ 4.31	28.90 $\pm$ 3.94	28.70 $\pm$ 4.25
IONOSPHERE	248	103	34	5.43 $\pm$ 2.16	8.59 $\pm$ 2.85	8.33 $\pm$ 2.78	8.76 $\pm$ 2.97
AUSTRALIAN	484	206	14	14.76 $\pm$ 2.21	14.54 $\pm$ 2.09	14.77 $\pm$ 1.90	14.45 $\pm$ 2.24
BREAST-CANCER	493	190	10	3.30 $\pm$ 1.06	3.15 $\pm$ 1.07	3.13 $\pm$ 1.01	3.15 $\pm$ 1.02
DIABETES	544	334	8	23.43 $\pm$ 2.38	23.68 $\pm$ 2.49	23.62 $\pm$ 2.33	23.80 $\pm$ 2.41
FOURCLASS	623	239	2	0.09 $\pm$ 0.18	0.04 $\pm$ 0.14	0.04 $\pm$ 0.14	0.08 $\pm$ 0.18
GERMAN.NUMER	718	282	24	24.95 $\pm$ 2.27	24.84 $\pm$ 2.29	24.93 $\pm$ 2.16	25.35 $\pm$ 2.10
SVMGUIDE3	892	392	21	16.48 $\pm$ 1.70	16.60 $\pm$ 1.77	16.55 $\pm$ 1.74	16.42 $\pm$ 1.68
COVTYPE-2000	1392	616	54	24.06 $\pm$ 1.60	23.92 $\pm$ 1.69	24.09 $\pm$ 1.49	24.14 $\pm$ 1.57
IJCNN1-2000	1424	584	33	4.38 $\pm$ 0.91	4.38 $\pm$ 0.93	4.37 $\pm$ 0.96	4.36 $\pm$ 0.86
A1A	1605	30956	123	15.78 $\pm$ 0.17	15.89 $\pm$ 0.21	15.75 $\pm$ 0.14	16.11 $\pm$ 0.51
SPLICE	2176	999	60	8.68 $\pm$ 0.87	8.93 $\pm$ 0.88	8.89 $\pm$ 0.90	8.79 $\pm$ 0.94
A2A	3365	30296	123	15.76 $\pm$ 0.30	15.74 $\pm$ 0.27	15.76 $\pm$ 0.28	15.97 $\pm$ 0.44
W1A	2477	47332	300	2.20 $\pm$ 0.07	2.18 $\pm$ 0.06	2.21 $\pm$ 0.07	2.22 $\pm$ 0.07
A3A	3185	29336	123	15.57 $\pm$ 0.08	15.82 $\pm$ 0.21	15.55 $\pm$ 0.10	15.85 $\pm$ 0.22
W2A	3470	46339	300	1.94 $\pm$ 0.09	1.95 $\pm$ 0.06	1.97 $\pm$ 0.06	1.94 $\pm$ 0.06
COVTYPE-5000	3472	1536	54	20.77 $\pm$ 0.88	20.73 $\pm$ 0.83	20.74 $\pm$ 0.86	20.79 $\pm$ 0.81
IJCNN1-5000	3486	1514	33	2.73 $\pm$ 0.42	2.70 $\pm$ 0.45	2.72 $\pm$ 0.41	2.70 $\pm$ 0.45
A4A	4781	33780	123	15.52 $\pm$ 0.07	15.80 $\pm$ 0.30	15.58 $\pm$ 0.13	15.64 $\pm$ 0.14
W3A	4912	44833	300	1.75 $\pm$ 0.05	1.75 $\pm$ 0.05	1.76 $\pm$ 0.05	1.72 $\pm$ 0.06
SVMGUIDE1	4959	2130	4	2.97 $\pm$ 0.32	3.01 $\pm$ 0.33	2.97 $\pm$ 0.32	3.03 $\pm$ 0.29
MUSHROOMS	5773	2351	112	0.00 $\pm$ 0.01	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	0.00 $\pm$ 0.01

Table 1: Characteristics of the considered data sets together with the test errors ($\pm$ standard deviations) on 100 random splits. The training and test set sizes refer to the splits used in the run time experiments. The considered algorithms were the 2D-SVM with WSS 7, I1-W4 and clipped duality gap stopping criterion (6th column), LIBSVM (5th column), the 2D-SVM with WSS 7, I1-W4 and duality gap stopping criterion (7th column), and another 2D-SVM with WSS 7, I1-W4 and clipped duality gap stopping criterion (8th column). The hyper-parameters for the first three test error columns were selected by 10-fold cross-validation on the 10 by 10 grid described in the text, while for the fourth test error column, 10-fold cross-validation on a finer and larger 25 by 30 grid was used.

values of λ and σ are contained in the intervals $[c_1 n^{-2}, 1]$ and $[c_2, c_3 n^{1/d}]$, respectively, where n is the number of training samples, d is the input dimension, and c_1, c_2, and c_3 are arbitrarily specifiable constants independent of n and d. Based on this result, we considered a geometrically spaced 10 by 10 grid in $[10n^{-2}, 1] \times [0.1, 2n^{1/d}]$, that is, the ratio of consecutive grid points was constant. Here, it is worth mentioning that during the k-fold cross validation λ was internally converted to C by the formula $C := \frac{k}{2(k-1)\lambda n}$ to accommodate the fact that the *actual* training set size for k-fold cross validation is approximately $(k-1)n/k$. To empirically validate the quality of this grid with respect to classification performance we repeated the 10-fold cross validation procedure 100 times for the fastest versions of the 2D-SVM and LIBSVM, respectively. We refer to Section 5.1 for an exact description of the experimental setup. The resulting average classification errors, which are reported in Table 1, show that both algorithms achieve comparable classification performance except on one small data set, namely IONOSPHERE. However, the size of this and some other data sets make it hard to draw conclusion from the corresponding, reported errors. While this experiment showed, that both algorithms performed equally well on the chosen grid, it does not allow statements about the absolute quality of the grid. We therefore conducted a control experiment[6] with the fastest version of the 2D-SVM on a geometrically spaced 25 by 30 grid in $[0.001n^{-2}, 1] \times [0.005, 20n^{1/d}]$. The size and boundaries of this control grid ensured that it was both significantly finer and larger than the 10 by 10 grid. Besides the different grid size, the experimental setup followed that described in Section 5.1 and the resulting average classification errors are reported in Table 1, too. The results in this table show that the classification performance is not improved when using the larger grid, which in turn means that our initial 10 by 10 grid is a good choice.

Like the choice of the grid, the stopping criterion and its threshold value have a significant influence of the number of iterations and the run time of an SVM solver. Unfortunately, the 2D-SVM and LIBSVM use different stopping criteria, which makes a direct comparison difficult. To address this problem, we again took a statistical perspective in the sense that the ultimate goal when solving the SVM optimization problem is not numerical but statistical accuracy. In other words, we may stop the iterative optimization procedures as soon as we know that the remaining inaccuracy does not significantly influence the classification performance. For the 1D-SVM and the 2D-SVM we thus used the stopping criterion (9) with $\varepsilon := 0.001$, while for our version of LIBSVM's solver we used, like the original LIBSVM, the classical MVP stopping criterion with $\varepsilon = 0.001$. Here we note that this was necessary since LIBSVM's solver deals with SVMs *with* offset b, and hence the stopping criterion (9) is no longer applicable. In addition, an appropriately modified stopping criterion seems to be computationally inefficient, while by (List et al., 2007, Lemma 8) the MVP stopping criterion with value $\varepsilon = 0.001$ also ensures (8) for $\varepsilon := 0.001$ and f^* instead of $[f^*]_{-1}^{1}$. In other words, LIBSVM's default value, which we picked throughout our experiments, actually has a good interpretation in terms of learning. Of course, the different stopping criteria used raise the question whether the results reported below are due to differences in the working set selection strategy, the different nature of the optimization problem, *or* the stopping criteria. In this regard, we note that in the experiments with LIBSVM our goal is to compare the *entire* 2D-SVM-solver with a state-of-the-art solver, rather than to, for example, compare different working set selection strategies. For this purpose, it is irrelevant whether the working set selection strategy, the nature of the optimization problem, or the different stopping criteria have a stronger influence on the run

6. This control experiment was extremely time-consuming, and hence we were forced to distribute the runs between different machines (with different hardware features). For the same reason, it was, unfortunately, infeasible to run the same control experiment for LIBSVM.

time. Nonetheless, it remains an interesting question for future work whether solver's for SVMs with offset can also benefit from some of the ideas of the working set selection strategies introduced for the 2D-SVM-solver.

In all experiments we pre-computed the kernel matrix in order to avoid that these solver-independent but data set-dependent computations are contained in the reported training times. To be more precise on the latter dependence, recall that the time needed to compute the matrix K significantly depends on the number of features of the samples and the implementation-specific internal representation of the samples. For example, we may have two data sets in $\mathbb{R}^{d_1}$ and $\mathbb{R}^{d_2}$, respectively, that generate the same matrix K. Now assume that $d_1 \ll d_2$. Without pre-computing the kernel matrix, the solver will need significantly more time for the second data set, while the run times for both data sets will be equal for pre-computed K. Moreover, the second data set may actually consist of samples for which most of the coordinates are zero. In this case, an internal data representation like LIBSVM's that exploits this sparseness may speed up the computation of both the entire K and single kernel matrix rows. On the other hand, if the data does *not* enjoy this kind of sparseness, a straightforward sample representation by arrays is typically superior, since it avoids costly branches, allows sequential RAM access, which, from our experience, is often 4 times faster than random access, and makes it possible to use vectorization features of modern processors such as SSE2 instructions. Last but not least, we observed recently, after the experiments of this paper were finished, that the pre-computation of K enjoys an almost linear speed-up, when it is distributed among the cores of modern multi-core processors, while for the computation of single rows of K the improvement may be significantly less due to the time spend for synchronization.[7] Obviously, these implementation options make it impossible to determine a canonical method for dealing with the kernel matrix K, whether it fits into memory or not. Consequently, by picking a particular method and including its run time into the measurements, one necessarily introduces a bias into the experiments, and hence run time results reported from a series of such experiments may be of little value for new, time-critical SVM applications with different data set characteristics.[8] In fact, for such applications all the considerations above need to be carefully taken into account, which in turn requires knowledge of the computational complexity of each *individual* component. In other words, for an informed decision one needs to know, among others, the run time complexity of the core solver, which in turn gives another argument for considering the core solver with pre-computed matrix.[9] On the downside, however, this approach is, of course, unrealistic for large data sets whose kernel matrices do not fit into the computers memory. On the other hand, for all considered data sets the matrices *did* fit into memory, and in addition, it turned out that for all data sets there were parameter regions of the grid where all or basically all vectors were support vectors. In these regions, the corresponding kernel rows would have been computed during the optimization,

7. This observation suggests that in the future, the computation of K, which is currently a significant part of the entire SVM training time, may be significantly less time consuming. This may be in particular true for highly parallel architectures such as graphical processing units.

8. In an extreme case, including the computation time for K opens the possibility that a new solver appears to be faster than existing ones simply because it has a better implementation for computing K.

9. Another approach would be to *a)* not pre-compute K and *b)* exclude the time needed to compute and cache kernel rows from the reported run time. Unfortunately, this approach is infeasible because of the relatively low resolution of the time-measuring functions in C.
   Yet another approach would be to actually precompute K, but to pretend that a cache of a certain size is used. One could then log cache misses of such a virtual cache. While the latter approach may actually be the silver bullet for future work, the idea for it only arose after the reviewers comments, and, unfortunately, a respective re-design of our experiments were too costly at that stage.

if we had not precomputed the kernel matrices, and consequently, training over the grid would have required the solver to compute the kernel matrix anyway. From this point of view, our experiments suggests that training over a grid with medium-sized data sets whose kernel matrix still fits into memory, there is no need to implement a caching strategy. In fact, we strongly conjecture that without pre-computing the kernel matrices, our experiments would have rendered computationally infeasible with one computer only. It is, of course, needless to say, that the situation may change, if other parameter selection strategies are used, or the data sets are too large.

5.1 Comparing Classification Performance

When comparing the standard SVM optimization problem with the version in (1), probably the first question is, whether the absence of the offset term has an influence on the classification performance. To answer this question we performed on each data set 100 runs for both a version of the 2D-SVM-solver and our implementation of LIBSVM's solver. We performed these experiments, though we report them first, actually at the very end of our investigations. This way, we could use for each solver the fastest version. For the 2D-SVM-solver it turned out, as we will see below, that this is the WSS 7 strategy together with I1-W4 initialization, while for the LIBSVM's solver we used, depending on the data set, either I1-W2 or I1-W5 initialization. Besides for the data sets of the AXA and WXA families, we generated for each data set 100 random splits, where each training set contained, modulo randomness, 70% of the samples. Moreover, on each of these training sets the hyper-parameter selection was performed by 10 fold cross-validation over the parameter grid described above. The test error was then computed on the test set part of the split, which, modulo randomness, contained 30% of the samples. The resulting average classification errors are reported in Table 1. As one quickly observes, LIBSVM yielded the better classification performance on the data set IONOSPHERE. On all other data sets, however, both algorithms performed almost indistinguishable. Therefore, it seems fair to conclude that the classification performance is not significantly influenced by the absence of the offset.

5.2 Comparisons to the 2D Selection Strategy with Maximal Gain

In our first set of experiments on 2D subset selection strategies, we investigated the number of iterations needed for the different strategies for selecting working sets. Our baselines were the 1D-SVM-solver and the 2D-SVM strategy WSS 0, which searches for the pair of indices with maximal dual gain. Since the latter is computationally very expensive, we decided to use only 2-fold cross validation. In addition, we actually trained only on one of the two folds, that is, our approach is best described by the training/validation SVM (TV-SVM) of Steinwart and Christmann (2008). Besides these modifications, however, we followed the approach outlined earlier. Finally, in all experiments of this subsection, we initialized by $\alpha \leftarrow 0$.

Let us now have a closer look at the results that are displayed in Figures 2 to 5. Figure 2 compares the 1D-SVM, WSS 0 and the simple 2D-modifications of the 1D-SVM. Not surprisingly, WSS 0 needs substantially less iterations than its one-dimensional equivalent 1D-SVM, while all of the simple 2D-modifications perform somewhere in between. More precisely, WSS 1 yields some significant improvement over the 1D-SVM. For WSS 2 the message is mixed; while on some data sets, WSS 2 performs significantly better, the difference is more marginal on other data sets. However, combining WSS 1 and WSS 2 into WSS 3 yields a clear overall improvement over both methods and the 1D-SVM. Another combination, WSS 5 that combines WSS 1 with a search over

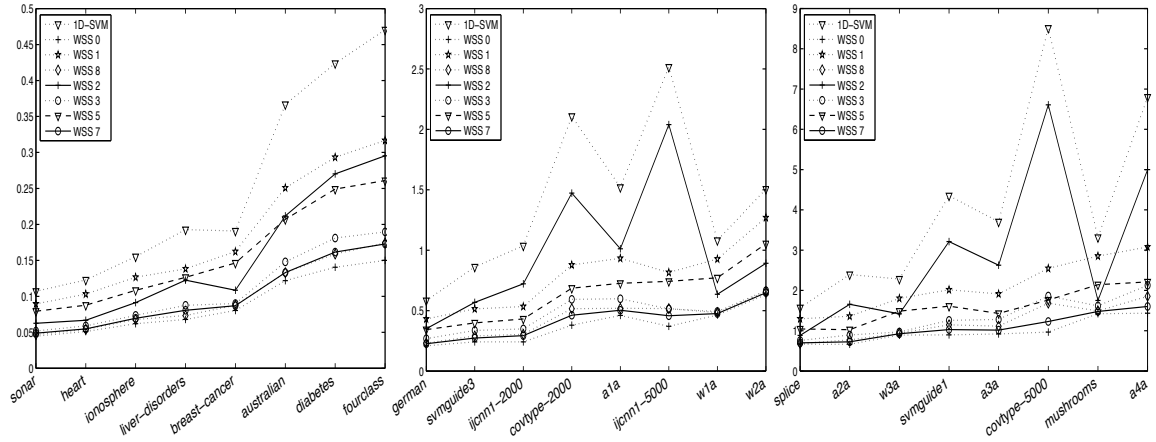


Figure 2: Performance of methods based on simple extensions of the 1D-search strategy for small (left), mid-sized (middle), and relatively large data sets (right). The graphic displays the average number of iterations in thousands for the different methods over the entire 10 by 10 parameter grid. All 2D-methods perform better than the 1D-SVM, but the degree of improvement differs significantly. WSS 2 performs sometimes better and sometimes substantially worse than WSS 1, but combining both methods into WSS 3 leads to uniform improvements. The same holds for WSS 5, though with less improvements. The combination WSS 7 uniformly yields the lowest number of iterations.

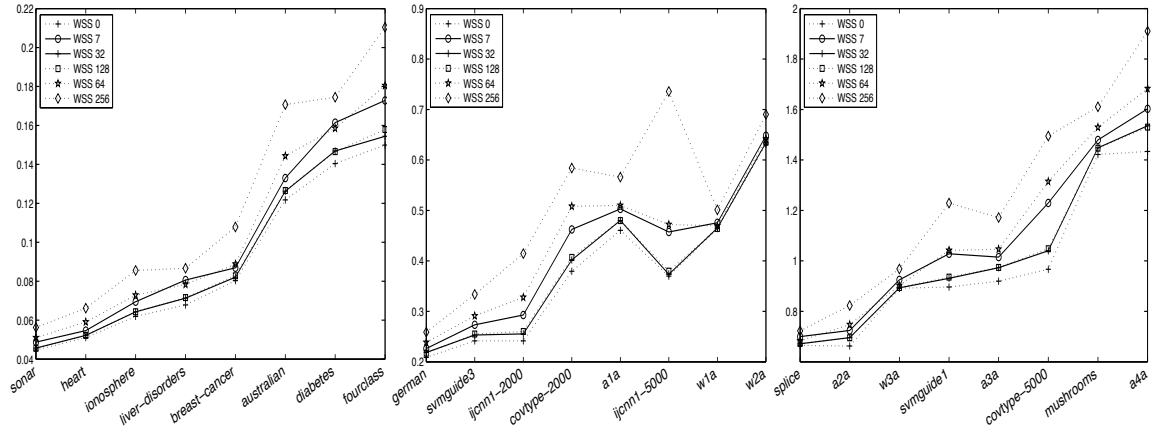


Figure 3: Performance of methods based on approximations of the 2D strategy WSS 0 (black). The graphic displays the average number of iterations in thousands for the different methods over the entire 10 by 10 parameter grid. WSS 0 performs uniformly best, but both deterministic strategies WSS 32 and WSS 128, which are basically indistinguishable, closely follow the performance of WSS 0. WSS 7 and the hybrid WSS 64 still capture most of the behavior of the previous methods with small advantages for WSS 7, while the complete randomization performs uniformly worst.

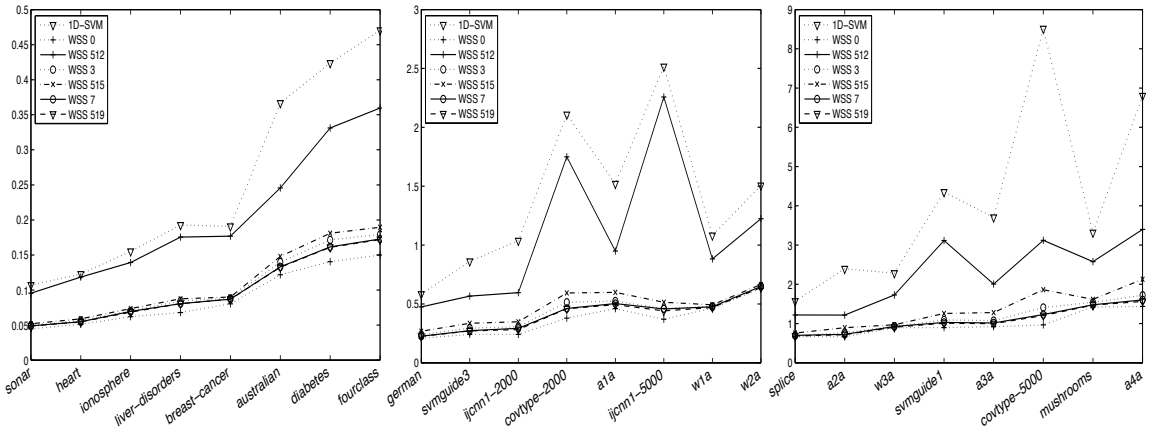


Figure 4: Combining methods based on simple 1D-extensions with WSS 512, which considers the approximate gain on inner SVs. The graphic displays the average number of iterations in thousands for the different methods over the entire 10 by 10 parameter grid. Without combining WSS 512 with other methods, it performs quite poorly, while combining WSS 512 with WSS 3 to WSS 515 yields an improvement over both methods. In contrast, combining WSS 512 and WSS 7 to WSS 519 does not give an improvement over WSS 7 as the almost indistinguishable two green lines show.

10 nearest neighbors, also needs substantially less iterations than WSS 1 and the 1D-SVM, but the improvements are less than those of WSS 3. However, the combination of all, WSS 7, does not only perform uniformly better than all participating methods, but also needs in most cases only a few more iterations than the optimal WSS 0. Finally, WSS 8, which is a variant of WSS 1, also reduces the number of iterations substantially, yet it fails to perform as well as WSS 7. Let us now have a closer look at Figure 3 that shows how the methods based on an approximation of the optimal WSS 0 perform. Here it turns that WSS 32, which uses the exact computation of the 2D-gain, and WSS 128, which uses an approximation of the 2D-gain, perform indistinguishably. In addition, they only need a few more iterations than WSS 0, and constantly outperform WSS 7, yet the latter improvement is in most cases only marginal. Finally, the random approaches WSS 64 and WSS 256 do not need less iterations than WSS 7, and the complete random approach of WSS 256 performs worse than the hybrid strategy WSS 64. However, by comparing with Figure 2 we see that WSS 256 still needs significantly less iterations than the 1D-SVM.

Another way to approximately compute the 2D-gain is implemented in WSS 512. Figure 4 compares the number of iterations of this method to the 1D-SVM, WSS 0, and some combinations of WSS 512 with simple 2D-extensions of the 1D-SVM approach. A closer look at this figure shows that WSS 512 alone is not a very good alternative to the 1D-SVM, while combinations do yield significant improvement. However, these improvements are not significantly better than WSS 7.

Finally, let us compare the 1D-SVM and the optimal 2D strategy WSS 2 with the MVP approach of WSS 16 and LIBSVM. Figure 5 shows that the 2D-MVP approach of WSS 16 performs only slightly better than the 1D-SVM. In contrast, LIBSVM needs, not surprisingly, substantially less iterations than the 1D-SVM, but it fails to perform as well as the simple WSS 3, and the more complicated WSS 7.

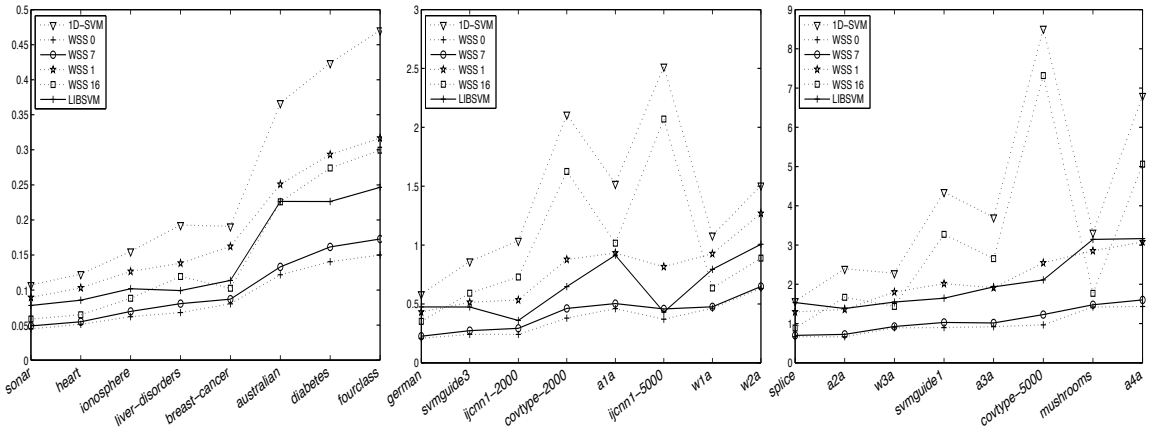


Figure 5: LIBSVM and MVP compared to some other approaches. The graphic displays the average number of iterations in thousands for the different methods over the entire 10 by 10 parameter grid. On all data sets considered, the 2D-MVP strategy WSS 16 has some advantage over the simple 1D-SVM, while LIBSVM often needs substantially less iterations and performs comparably to the WSS 1. However, neither of the methods approach the close-to-optimal performance of WSS 7 or even the optimal performance of WSS 0.

5.3 Comparisons of Different 2D Subset Selection Strategies

The experiments of the previous subsection identified some working set selection strategies that performed close to WSS 0 in terms of iterations. Unlike WSS 0, all these strategies were $O(n)$, yet is seems obvious, that their run time may substantially differ. Therefore, the goal of the experiments in this subsection is to evaluate the working set selection strategies in terms of their run time. To this end, we performed the already described 10-fold cross validation training on our data sets and measured both the number of iterations and the run time. Note that by considering both quantities simultaneously, it is possible to decide whether a strategy suffers from its large number of iterations or only from its computational requirements for selecting the working set. In the following, we only summarize our findings, since Appendix A.1 contains various graphics displaying our results of this subsection in detail. In this appendix, we always report the average requirements per grid point, where the average is either taken with respect to all 10 folds and the entire grid, or just with respect to the 10 folds and the grid points whose validation error is close to the minimal validation error. The latter averages are of particular interest, when one does not use grid search for the hyper-parameter selection, but some other, possibly faster methods, such as the one by Keerthi et al. (2007). In addition, the latter averages are also interesting for grid search, since after such a search one usually retrains the SVM on the entire training set with the hyper-parameters that performed best in terms of validation error.

Let us now have a closer look at the results. The first observation, see Figures 8 and 9 for details, is that WSS 2, which needs less iterations than the 1D-SVM, does not run substantially faster. However, this behavior can be relatively easily explained by the fact that in each iteration the 1D-SVM updates the gradients for one direction only, whereas WSS 2, due to its 2D-nature, performs

two such updates per iteration. Similarly, WSS 8 cannot translate its advantage over WSS 1 in terms of iterations into a substantial advantage in terms of run time. In this case, a closer look reveals that, compared to WSS 1, WSS 8 performs an additional, implicit gradient upgrade when looking for the second direction j. The other results displayed in Figures 8 and 9 confirm our results from Figure 2. In particular, WSS 7 not only need the fewest number of iterations, but also runs fastest on almost all data sets. Finally, Figure 10 reveals, for which hyper-parameters some combined methods achieve their speed-up compared to WSS 1. In particular, for large values of λ, WSS 3 and WSS 7 need only half of the iterations of WSS 1 and WSS 5, which indicates that in this regime, WSS 2 is the dominating strategy in the former two combinations. On the other hand, for small values of λ, the nearest neighbor strategy WSS 4 seems to be the dominating working set selection strategy of WSS 5 and WSS 7, since both methods need substantially less iterations than the methods WSS 1 and WSS 3, which do not include the nearest neighbor strategy. Finally, these advantages in terms of iterations do translate into almost the same advantage in terms of run time, since the additional costs of the nearest neighbor strategy only depend on the number k of considered nearest neighbors, which, in general, is quite small compared to the sample size. Nonetheless it is worth mentioning that for a few hyper-parameter pairs, it is faster not to use the nearest neighbor strategy.

Let us now turn to the methods that try to approximate the working set strategy of the optimal WSS 0. Here, see Figures 11 and 12 for the details, it turns out, that WSS 32 and WSS 128, whose required number of iterations were closest to WSS 0, have a significant higher run time than WSS 7. Since the number of iterations of these three methods behave quite similarly, the only explanation for this different run time behavior is the additional cost per iteration for computing all (approximate) 2D-gains. This explanation is further confirmed by the fact that WSS 128, which involves the cheaper approximate 2D-gain has a better run time behavior than WSS 32, which uses the exact computation of the 2D-gain. Furthermore, WSS 64, which computes only a fifth of the 2D-gains WSS 32 computes, runs substantially faster than WSS 32, despite the fact the the former needs more iterations. In this regard, we finally note that WSS 256 runs over-proportionally slowly compared to, for example, WSS 128. Most likely, this behavior can be explained by less effective hardware caching for the random pair selection of WSS 256. To get a better impression, on how effective WSS 7 chooses its working sets, let us now have a closer look at the number of iterations of the different working set selection strategies. The bottom graphics of Figure 11 show that over the entire grid, WSS 7 only needs 5% to 20% more iterations than the best performing WSS 32. However, if one considers only the grid points with small validation error, this good behavior becomes worse. Indeed, the bottom graphics of Figure 12 show that for such hyper-parameters, WSS 7 typically needs more than 20% more iterations than WSS 32, and in some cases even more than 50% more. Finally, Figure 13 reveals that in particular for small values of λ and flat kernels, WSS 7 requires substantially more iterations than WSS 32. However, at least on the data set SVMGUIDE1 this worse behavior takes place at grid points that do not need a lot of iterations anyway, and hence the advantage of WSS 32 is marginal.

The next question, which naturally arises from the observations above, is whether the number of iterations used in WSS 7 can be reduced by combining WSS 7 with some methods that mimic WSS 32 on the inner support vectors. Here, Figure 11 shows that the number of iterations can be reduced by such combinations in a few cases, but this never pays off in terms of run time, if one considers the entire grid. On the grid points with small validation error, however, the situation is slightly more involved. Clearly, the combination with WSS 2048 performs worst, yet combining WSS 7 with WSS 512 or WSS 1024 sometimes yield a shorter run time. Finally, Figure 16 shows

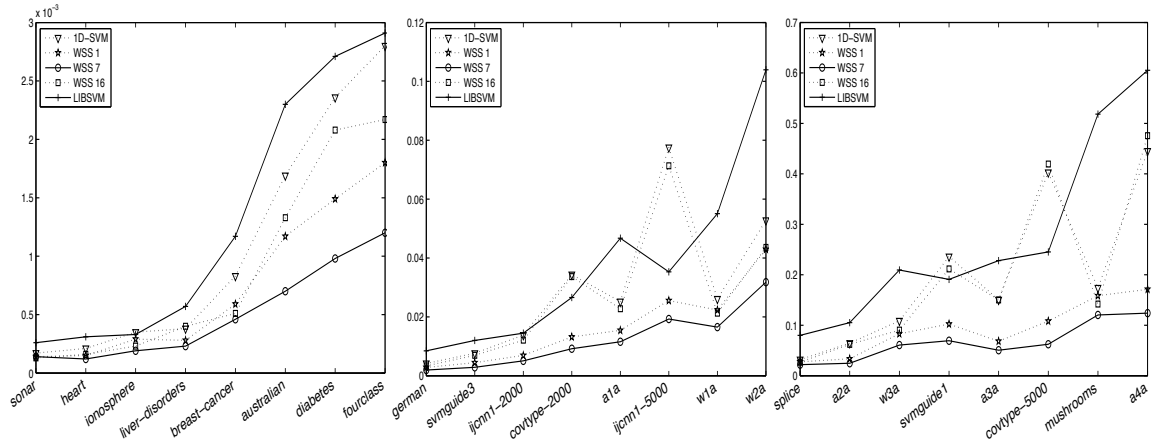


Figure 6: Average run time in seconds per grid point of LIBSVM and MVP compared to some other approaches over the entire 10 by 10 grid. The 2D-MVP approach of WSS 16 is not a good alternative to the 1D-SVM or even the two-dimensional WSS 7. Moreover, LIBSVM is significantly slower than WSS 7.

that, at least for the data set SVMGUIDE1, the improvements achieved by these combinations are mainly at grid points that do not require a lot of iterations. On the other hand, it also illustrates that the computational overhead of these combinations is significant.

Finally, let us compare LIBSVM with some subset selection strategies such as the MVP approach of WSS 16 and the overall best performing WSS 7. Here, see Figure 6 for a short impression and Figures 17 to 19 for the details, the most interesting observation is that although WSS 1 and LIBSVM have comparable behavior in terms of iterations, they substantially differ in terms of their run time. Because we used our own implementation of LIBSVM's solver, which employed the same SSE2 optimizations as the 2D-SVM methods, the only way to explain this behavior is that the subset selection strategy of LIBSVM is significantly more expensive than the simple WSS 1. To understand the latter, recall that LIBSVM's strategy is based on computing an approximate 2D-gain, which is quite expensive as we have seen in Figures 11 and 12 for the 2D-SVM methods WSS 32, WSS 64, WSS 128, and WSS 256. In addition, LIBSVM's strategy cannot be efficiently vectorized, which is another disadvantage compared to WSS 1. Finally, it is interesting to note that WSS 7 is between 2 and 4 times faster than LIBSVM, when the average over all grid points is considered. Moreover, on the grid points with small cross validation error, the improvement is rarely less than a factor of 4, and as Figure 19 illustrates, this is most likely not an artifact caused by different optimal grid points. Indeed, on some grid points LIBSVM needs more than 10 times the run time WSS 7 requires.

5.4 Influence of the Stopping Criterion

In this subsection, we investigate the influence of the stopping criterion (9) on the computational requirements. To this end, we considered the 10-fold cross validation procedure described earlier. Moreover, in order to save time, we only considered the best performing working set selection strategy, namely WSS 7. For this method, we considered our stopping criterion (9) and the classical

duality gap stopping criterion (7), where we set the right hand side of both stopping criteria to be $\varepsilon/(2\lambda)$ with $\varepsilon := 0.001$. Note that this is exactly the same set-up as in our previous experiments, and it is not hard to show that for the duality gap (7), this choice again leads to the same theoretical bounds on the generalization performance.

The results of our experiments are summarized in Figures 20 to 22. A quick look shows that, not surprisingly, the stopping criterion (9) never leads to more iterations, but the improvements depend very much on the data set. Moreover, these smaller number of iterations also pay off in terms of run time, though the effect is less pronounced when we consider the entire grid. We believe this is due to the fact that computing (9) is a little more expensive than computing (2), since it involves two instead of just one clipping operations. In this regard, it is interesting to note that the SSE2 instruction set in emmintrin.h makes it possible to avoid expensive branches for the computation of the clipping by providing *min*() and *max*() operations. In turn, this results in a relatively cheap stopping criterion; from some ad-hoc measurements made for a different purpose, we estimate that this computation costs about 10% of an entire iteration, though the exact numbers are most likely hardware dependent. When we only consider the grid points with small validation error, the positive effect of the clipped duality gap is amplified as Figure 21 shows. The reason for this behavior is illustrated in Figure 22 for the SVMGUIDE1 data set. Indeed, this figure shows that for small values of λ, the stopping criterion (9) leads to both substantially less iterations and shorter run times, whereas for larger λ, the computational requirements for both stopping criteria are essentially the same. Although uniformly superior, the positive effect of using (9) is thus highly inhomogeneously distributed over the parameter grid.

5.5 Comparing Different Numbers of Nearest Neighbors

So far we have considered WSS 7 for 10 nearest neighbors only. Of course, this was a relatively arbitrary choice, and hence it is interesting to investigate how the computational requirements change with the number of nearest neighbors. This is the goal of this subsection.

To this end, we again used the 10-fold cross validation procedure described earlier. Moreover, we considered the behavior of WSS 7 for N-nearest neighbors, where $N = 5, 10, 15, 20, 25, 30$. Our first observation was that, for $N = 25$ and $N = 30$, there was rarely an improvement in terms of iterations, but the required run time tended to slightly increase compared to smaller N. To keep the figures clean, we hence plotted the results for $N = 5, 10, 15, 20$, only. Figures 23 and 24 show that WSS 7 behaves slightly worse for $N = 5$, while for larger N the behavior over the *entire* grid is essentially indistinguishable. The latter observation mildly changes, if one only considers the hyper-parameters with small cross validation error, yet it is unclear to which extend this effect is caused by possibly different hyper-parameters picked by the different methods. In addition, a detailed look at Figure 25 does not really clarify the situation, since many of the run times measured are close to the finest resolution of time.h. Consequently, it seems safe to say that, at least in the range $N = 10 \ldots 20$, the performance of WSS 7 is essentially independent of N.

5.6 Comparing Different Initializations

Let us now investigate the influence of different initialization strategies on the computational requirements. To this end, we trained 2D-SVM with WSS 7 and with different combinations of cold and warm start options on the data sets summarized in Table 1. Moreover, we again used the 10-fold cross validation procedure described earlier.

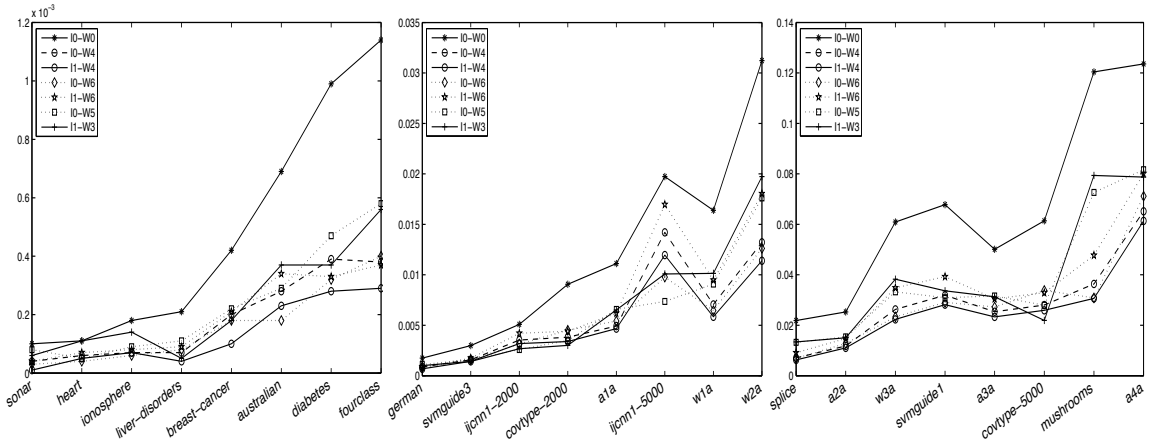


Figure 7: Average computational requirements per grid point of more complex initialization strategies for the 2D-SVM with WSS 7 for small (left), mid-sized (middle), and relatively large data sets (right). The graphics display the run time in seconds. The cold start initializations with zeros (I0- plots) need less iterations but in most cases more run time. In almost all cases, the more complicated initialization strategies perform better than the simple warm start approaches. Overall, I0-W4, I1-W4, and I0-W6 are the most efficient methods in terms of run time.

The first observation is, see Figure 26 for details, that initializing with zeros always leads to less iterations than initializing with a kernel rule. Surprisingly, however, the required run time for both initialization strategies is substantially less different. A closer inspection revealed[10] that this is caused by the fact that the solver initialized with the kernel rule method I1-W1 spends most of its iterations during initialization, that is, most of the iterations counted are from the outer loop of Procedure 4. Since these iterations do not involve the working set selection and the computation of the stopping criterion, they are relatively cheap compared to the iterations of the actual solver described in Algorithm 2. Moreover, Figure 26 shows that the simple warm start strategies W2, W3, and W5 do reduce the computational requirements significantly, where in almost all cases the scaling approach of W3 and W5 performs superior.

Interestingly, the computational requirements can often be further reduced by one of the more complicated initialization strategies W4 and W6 as Figure 7 illustrates, see also Figure 27 for more details. In particular, the combinations I0-W4, I1-W4, and I0-W6 run in most cases faster than the simple combination I0-W3, and overall it seems fair to say that I1-W4 performs best. However note that this approach requires access to the entire kernel matrix, and hence the combinations I0-W4 and I0-W6 may be the better choice, if storing this matrix is not an option.

We also conducted a control experiment in which the warm start options available for SVMs with offset are compared. Figure 28, which displays the corresponding results, shows that in most cases scaling by W3 and W5 is better than keeping the solution, that is, W2. This is similar to our results for SVMs without offset, but a closer look reveals, that the run time gain for SVMs with offset is both less pronounced and less consistent. In particular for the larger data sets, the gain by

10. For brevity's sake we omitted the display of the corresponding plots.

using a warm start for SVMs with offset is about 20%, while for SVMs without offset it is about 45% even if only the simple warm start option W5 is used. Moreover, the more complex strategies for SVMs without offset can reduce the run time by about 60% on these data sets. Consequently, it seems fair to say that SVMs without offset benefit substantially more from warm start strategies than SVMs with offset do.

Let us finally investigate the effect of some of the initialization strategies for different hyper-parameter pairs. Here Figure 29 reveals that the warm start options perform almost uniformly better than the cold start option I0-W0. Moreover, the complex warm start strategy W4 achieves a significant gain for small values of λ. Since these λ are computationally more demanding than large values of λ, the success of W4 can be explained. On the other hand, the strategies W5 and W6 start with the smallest value of λ, and hence they do not achieve any improvement over I0-W0 for this λ. However, they achieve a significant improvement for basically all other values of λ, which in turn explains their success. By combining these observations and the fact that the cold start I0 requires a relatively small number of iterations on medium values for λ, it thus seems promising to use a hybrid strategy that starts with such a medium value for λ, and then performs W4 for smaller λ and W6 for larger values. However, investigating such a strategy is out of the scope of this paper.

6. Conclusions

We have thoroughly investigated SVMs without offset term b that use the hinge loss and Gaussian kernels. It turned out that these SVMs have convergence rates and classification performance that are comparable to SVMs with offset, while the absence of the offset gives more freedom in the algorithm design. In particular, we identified three areas, where this additional freedom results in faster algorithms:

- **Working set selection.** In principle, an SMO-type solver for SVMs without offset can update one variable at each iteration. However, we saw that this approach does not lead to run times that were shorter than those of an SMO-type solver for SVMs with offset. We then identified some selection methods for working sets of size two, that modified the search for working sets of size one only very slightly. It further turned out that these modifications decreased the number of iterations substantially, and since updating the gradient and computing the stopping criterion for two variables did not change the costs of an iteration dramatically, these modification also resulted in a significantly shorter run time. It is further worth mentioning that the most successful selection strategies for workings sets of size two were actually combinations of a couple of such simple modifications. The reason for the latter was that some strategies worked particularly well for large values of the regularization parameter λ, while others worked better for small values of λ. The good combinations then contained both types of strategies and identified the better one at each iteration automatically.

- **Stopping criterion.** Another improvement of the run time behavior of our algorithm came from a new stopping criterion that has a clear justification from recent statistical analysis of SVMs. This stopping criterion, which is essentially a relaxed duality gap, never leads to more iterations than the classical duality gap stopping criterion, but it often decreased the number of iterations. Moreover, its computational costs were almost identical to those of the classical duality gap, and hence it often resulted in shorter run times.

- **Warm start initializations.** SVMs without offset also allow more freedom in the design of warm start initializations when the hyper-parameters are determined over a grid of hyper-parameters. We investigated a couple of such initialization methods and saw that some of them led to a substantially shorter run time. Moreover, by comparing to some warm start initializations for SVMs with offset, we observed that SVMs without offset benefit significantly more from such strategies.

It seems fair to remind the reader that in our experiments we only considered data sets for which the kernel matrix fit completely in the RAM of a desktop computer. With present configurations of, say up to 8GB RAM, this limits the data set size somewhere between 25,000 and 30,000 samples. While such sizes are typically not considered to be extremely large, they already constitute a decent challenge for existing off-the-shelf SVM software, if the training time is an issue. Moreover, even for smaller data sets a fully automated hyper-parameter selection run for SVMs with offset is, for some applications, too time intensive. Our new SVM solver yields a significant reduction in time for medium-sized data sets, thus opening the applicability of SVMs to such problem domains. However, it seems fair to say that although many data sets actually fall in this range of size, some other applications demand processing substantially larger data sets. So far, it remains unclear, how well our new solver performs for such data sets, and since our experimental study was already quite extensive and expensive, we postpone this question to future research.

Some other directions of future research include the following questions: *a)* Are there cheap modifications of our 2D-working set selection strategy that identify working sets of larger size for which the number of iterations and the run time is further reduced? *b)* Can some of our ideas be used or modified for other SVMs, that, for example, use different kernels and/or loss functions? *c)* Can the run time of the solver be further improved by not only using vectorization via SSE instructions but by also distributing tasks between different cores of a modern processor?

Appendix A. Detailed Experimental Results

On the following pages, we present more graphics illustrating our experimental findings. To keep these graphics in order, we divided the appendix in several subsections, which follow the order of the subsections of Sections 5.

A.1 Results for the Different Working Set Selection Methods

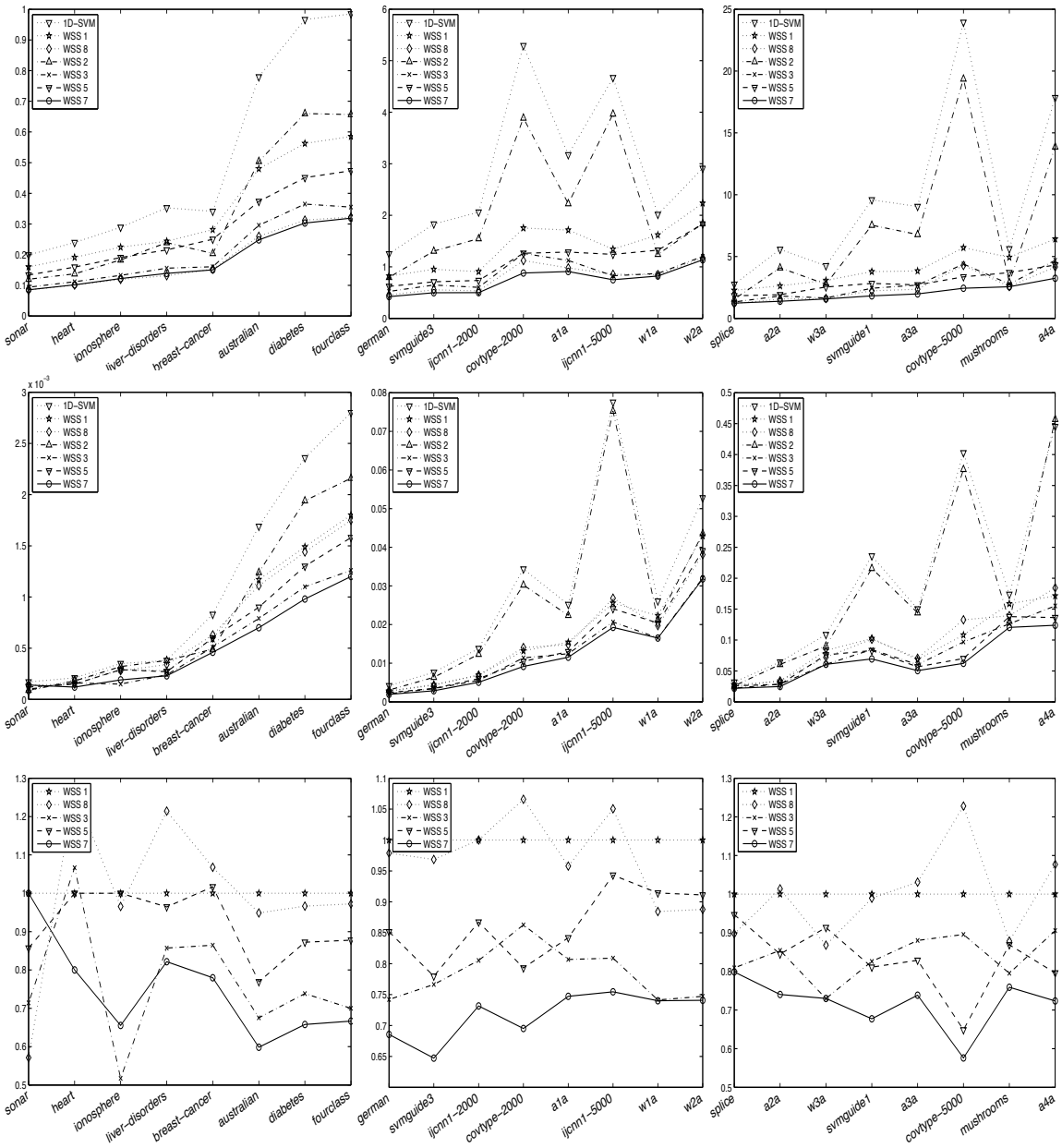


Figure 8: Average computational requirements per grid point of simple extensions of the 1D-search strategy over the entire 10 by 10 grid. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios WSS x/WSS 1 of the run times (bottom). WSS 7 performs almost uniformly best in both metrics.

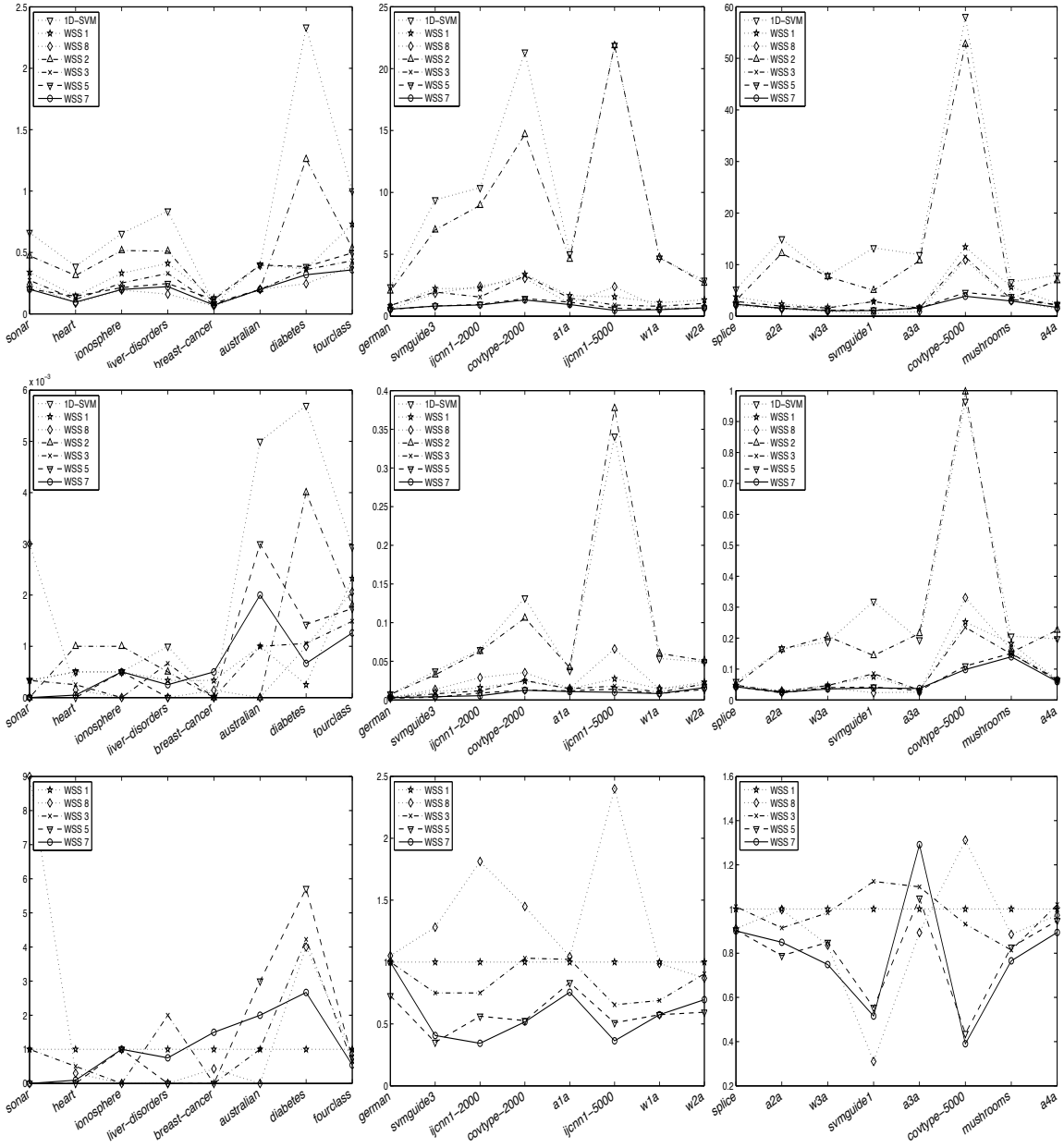


Figure 9: Computational requirements of simple extensions of the 1D-search strategy on the grid points whose cross validation error is not larger than 1.05 the minimal cross validation error. The graphics display the average number of iterations in thousands (top), the run time in seconds (middle), and the ratios WSS x/WSS 1 of the run times (bottom). Unfortunately, for the small data sets, the run time measurements are not very reliable. In addition, the set of considered grid points may slightly vary for the different methods, which in turn may influence the computational requirements and hence the graphic at the bottom left has little informative value. It seems fair to say that overall, WSS 7 performs best in both metrics, but is closely followed by WSS 5 in terms of run time.

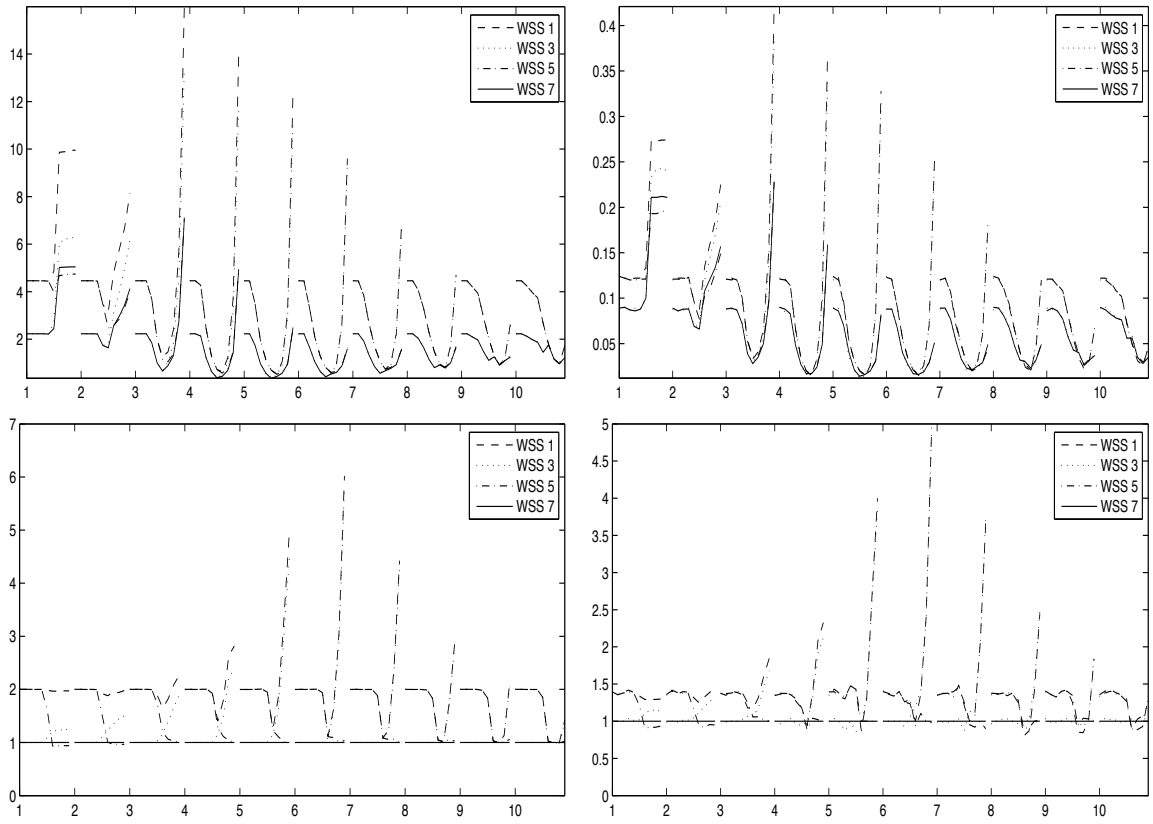


Figure 10: Computational requirements per single grid point of simple extensions of the 1D-search strategy for the SVMGUIDE1 data set. Each horizontal cell numbered by 1 to 10 corresponds to a single kernel parameter σ and an ordered run through the 10 λ-values, where the left of each cell corresponds to the largest λ-value, and the right to the smallest. Analogously, cell 1 corresponds the the largest σ-value, and cell 10 on the right corresponds to the smallest σ-value. The graphics at the top display the number of iterations in thousands (left) and the run time in seconds (right), both averaged over the 10 folds, for WSS 1, WSS 3, WSS 5, WSS 7, and WSS 8. WSS 7 performs almost uniformly the best in both metrics. However, for large λ, WSS 3 performs comparable, while for small λ, WSS 7 is closely followed by WSS 5. The graphics at the bottom show the ratios WSS x/WSS 7, x= $1,3,5,7$, for the number of iterations (left) and the run time (right) to illustrate the performance gain of WSS 7.

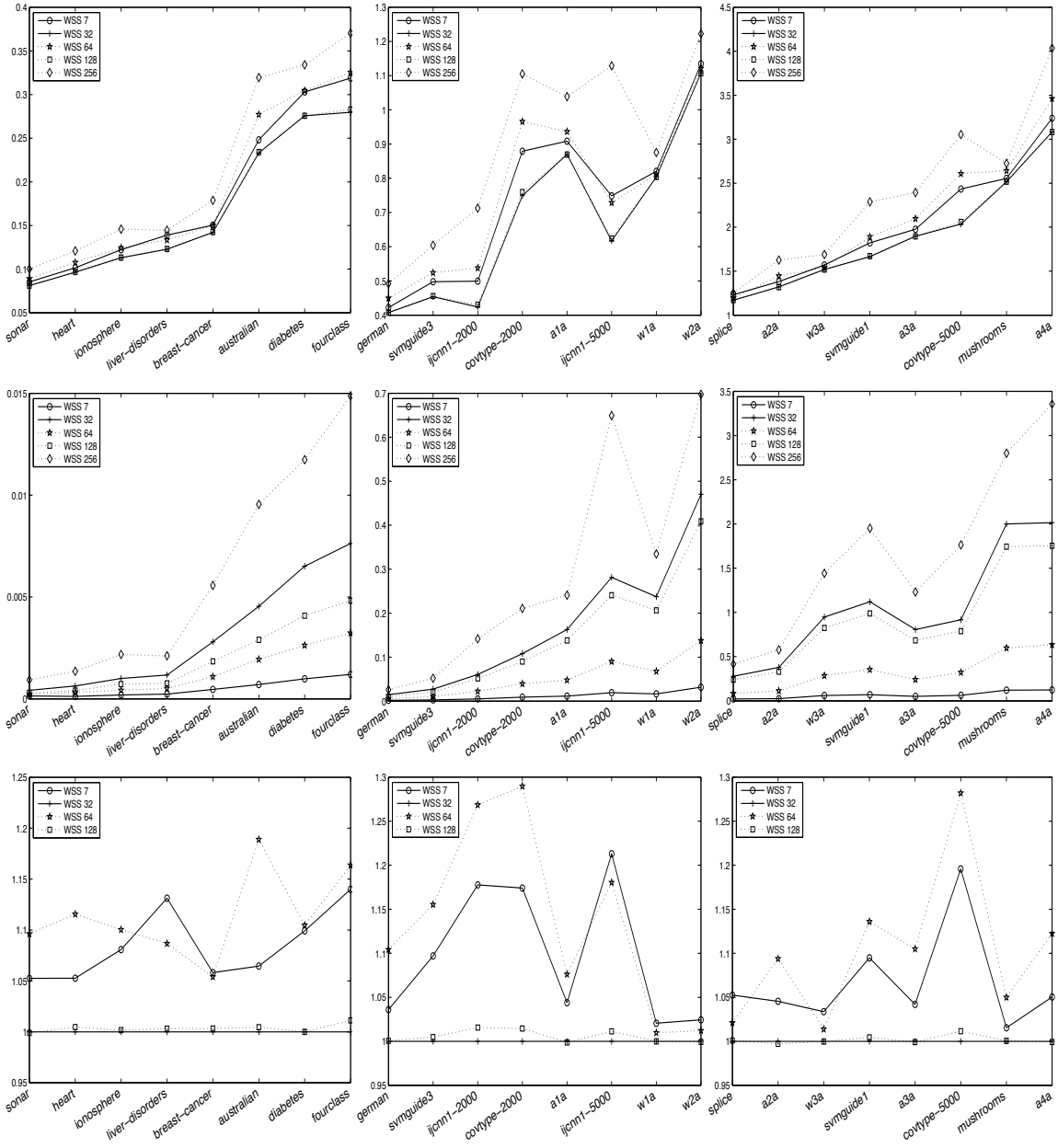


Figure 11: Average computational requirements per grid point of methods based on approximations of the maximal gain strategy WSS 0. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios WSS x/WSS 32 of the number of iterations (bottom). Although WSS 7 and the semi-random WSS 64 need slightly more iterations than WSS 32 and WSS 128, their costs per iteration is substantially less, which results in a significantly shorter run time. The completely random WSS 256 needs over-proportionally more run time, possibly because of the less effective hardware cache.

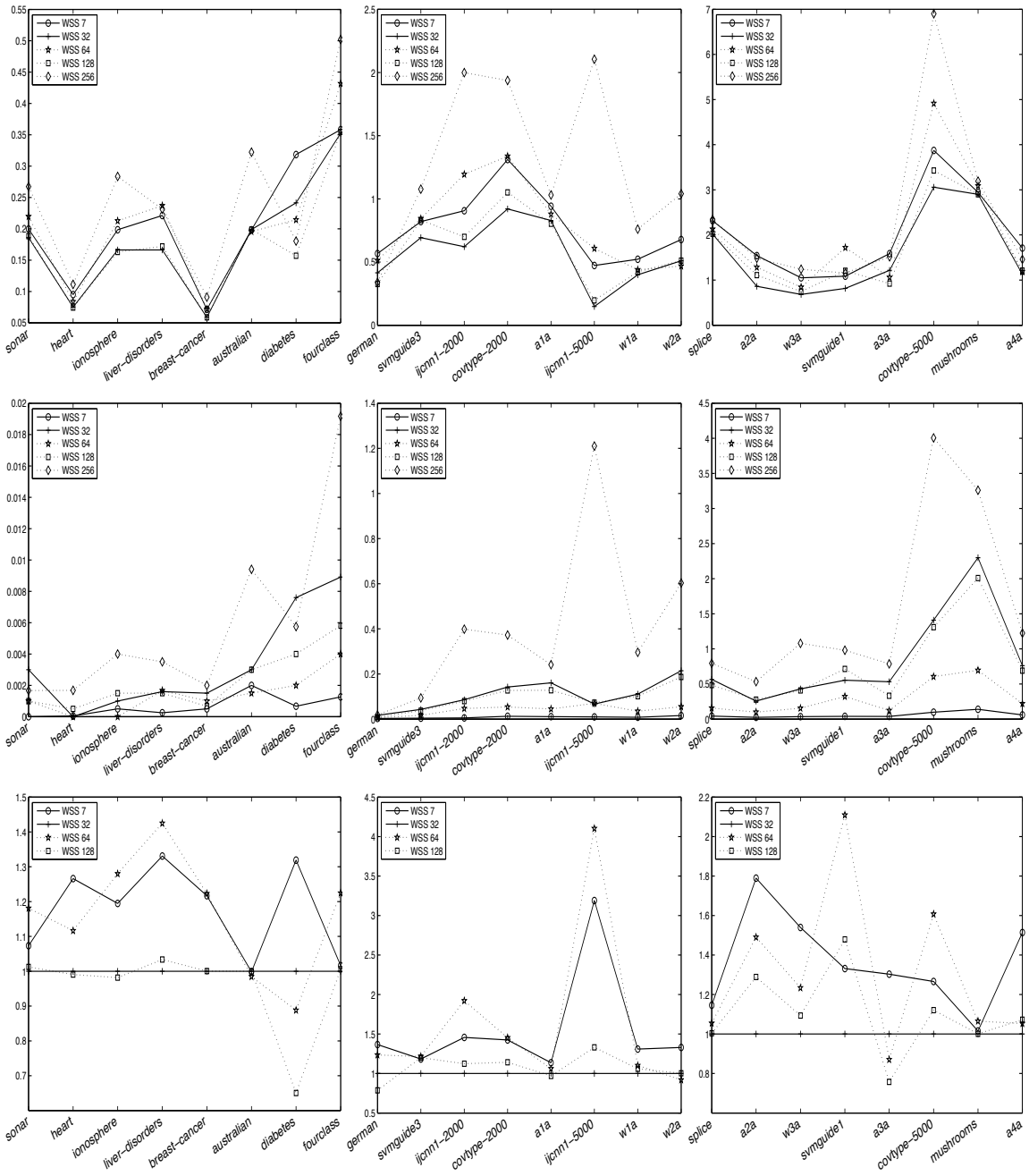


Figure 12: Computational requirements of methods based on approximations of the maximal gain strategy WSS 0 on the grid points whose cross validation error is not larger than 1.05 the minimal cross validation error. The graphics display the average number of iterations in thousands (top), the run time in seconds (middle), and the ratios WSS x/WSS 32 of the number of iterations (bottom). For the small data sets, the run time measurements are not very reliable. In addition, the set of considered grid points may slightly vary for the different methods, which in turn may influence the computational requirements.

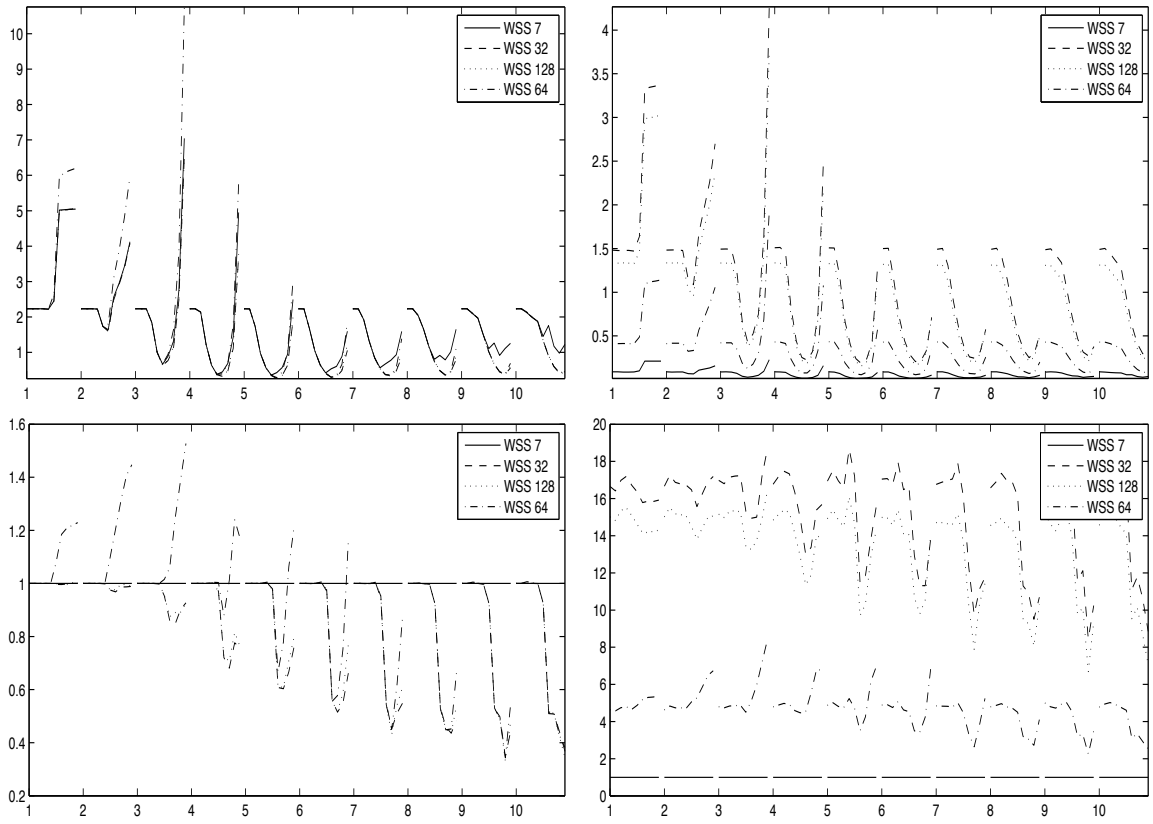


Figure 13: Computational requirements per single grid point of methods based on approximations of the maximal gain strategy WSS 0 for the SVMGUIDE1 data set. The four graphics have the same format as the ones in Figure 10. The graphics at the top display the number of iterations in thousands (left) and the run time in seconds (right), both averaged over the 10 folds, while the graphics at the bottom display the corresponding ratios WSS x/WSS 7. For some grid points, WSS 7 and WSS 32 need approximately the same number of iterations, while for some other grid points, WSS 7 needs significantly more. Nonetheless, the run times of WSS 32 are substantially worse than that of WSS 7.

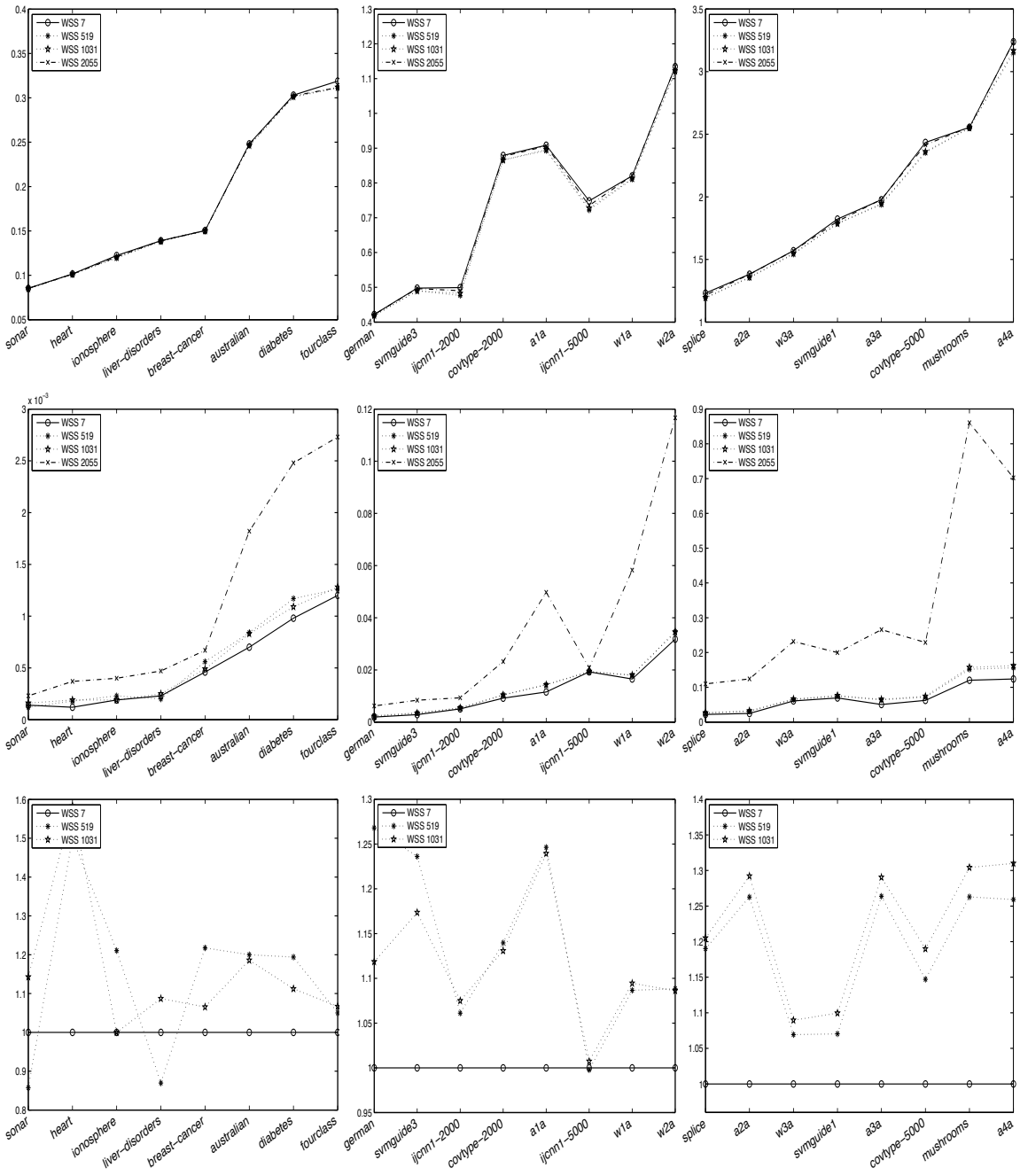


Figure 14: Average computational requirements per grid point of combining WSS 7 with some methods that use the formula for the approximate gain on inner SVs over the entire 10 by 10 grid. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios WSS x/WSS 7 of the run times (bottom). Although the combinations need a slightly smaller number of iterations, their additional overhead per iteration leads to longer run times.

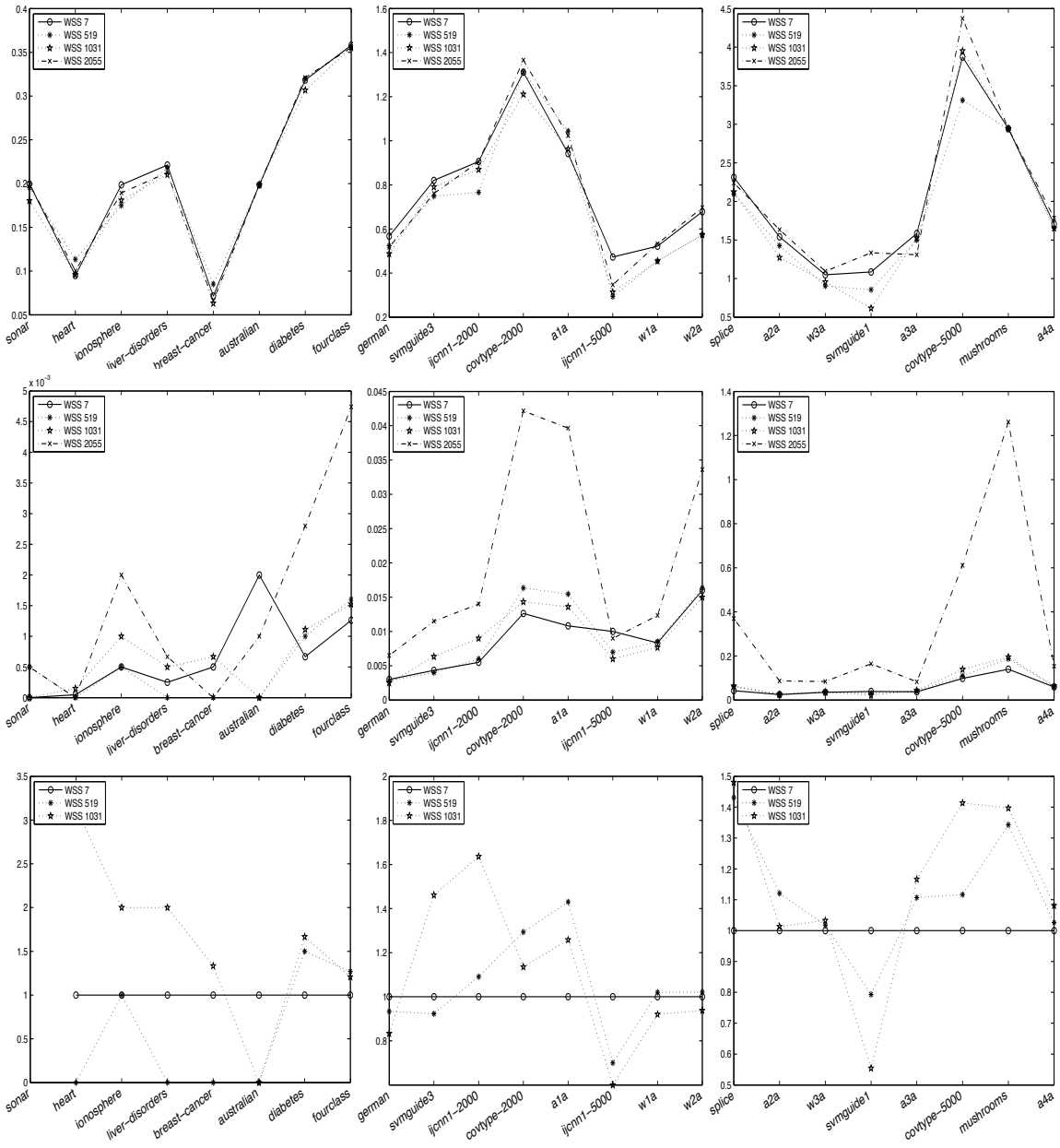


Figure 15: Computational requirements of combining WSS 7 with some methods that use the formula for the approximate gain on inner SVs on the grid points whose cross validation error is not larger than 1.05 the minimal cross validation error. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios WSS x/WSS 7 of the run times (bottom). For the small data sets, the run time measurements are not very reliable. In addition, the set of considered grid points may vary slightly for the different methods, which in turn may influence the computational requirements.

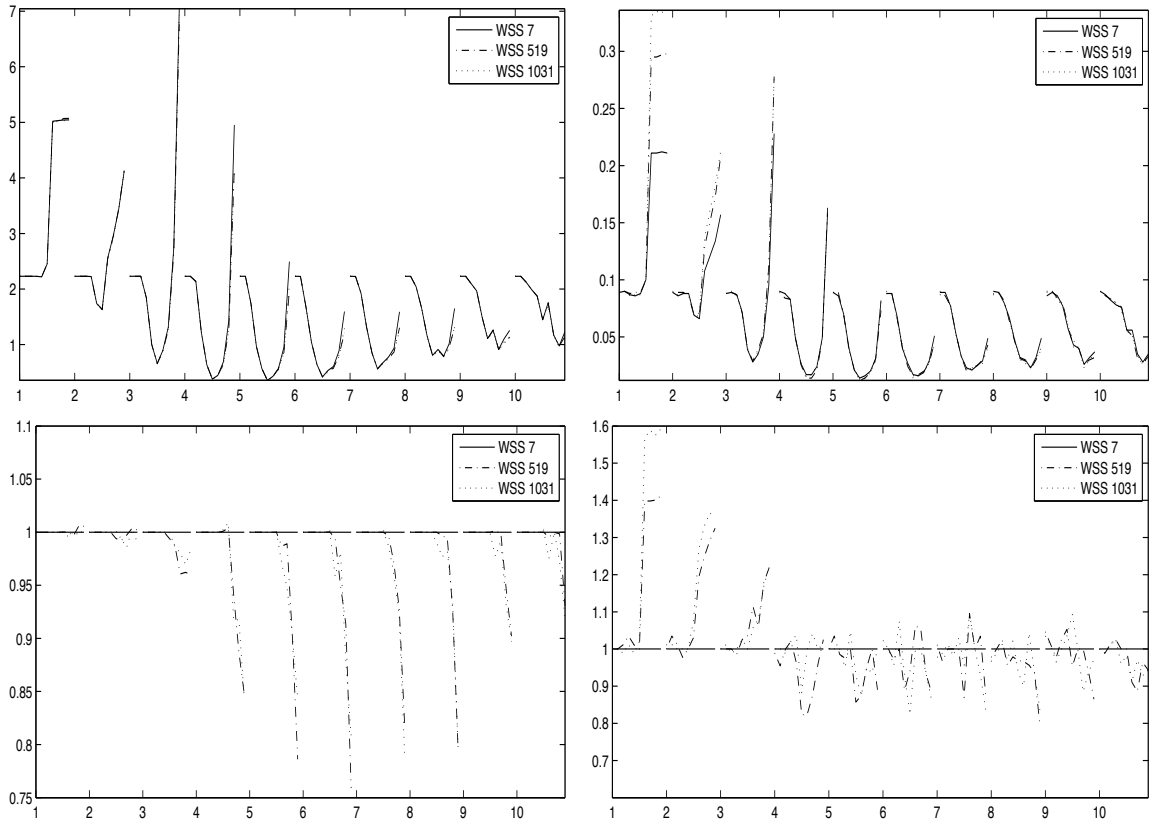


Figure 16: Computational requirements per single grid point of methods based on simple extensions of the 1D-search strategy for the SVMGUIDE1 data set. The four graphics have the same format as the ones in Figure 10. The graphics at the top display the number of iterations in thousands (left) and the run time in seconds (right), both averaged over the 10 folds, while the graphics at the bottom display the corresponding ratios WSS x/WSS 7. Note that for large λ the Boolean flag of WSS 4 is typically not set to true during the optimization, and hence all methods reduce to WSS 3. Analogously, for large λ and σ, the graphics nicely display the additional costs of WSS 512 and WSS 1024. Finally, the differences in the run time occur on a very low and hard to measure level, which explains the fluctuations in the bottom right graphics.

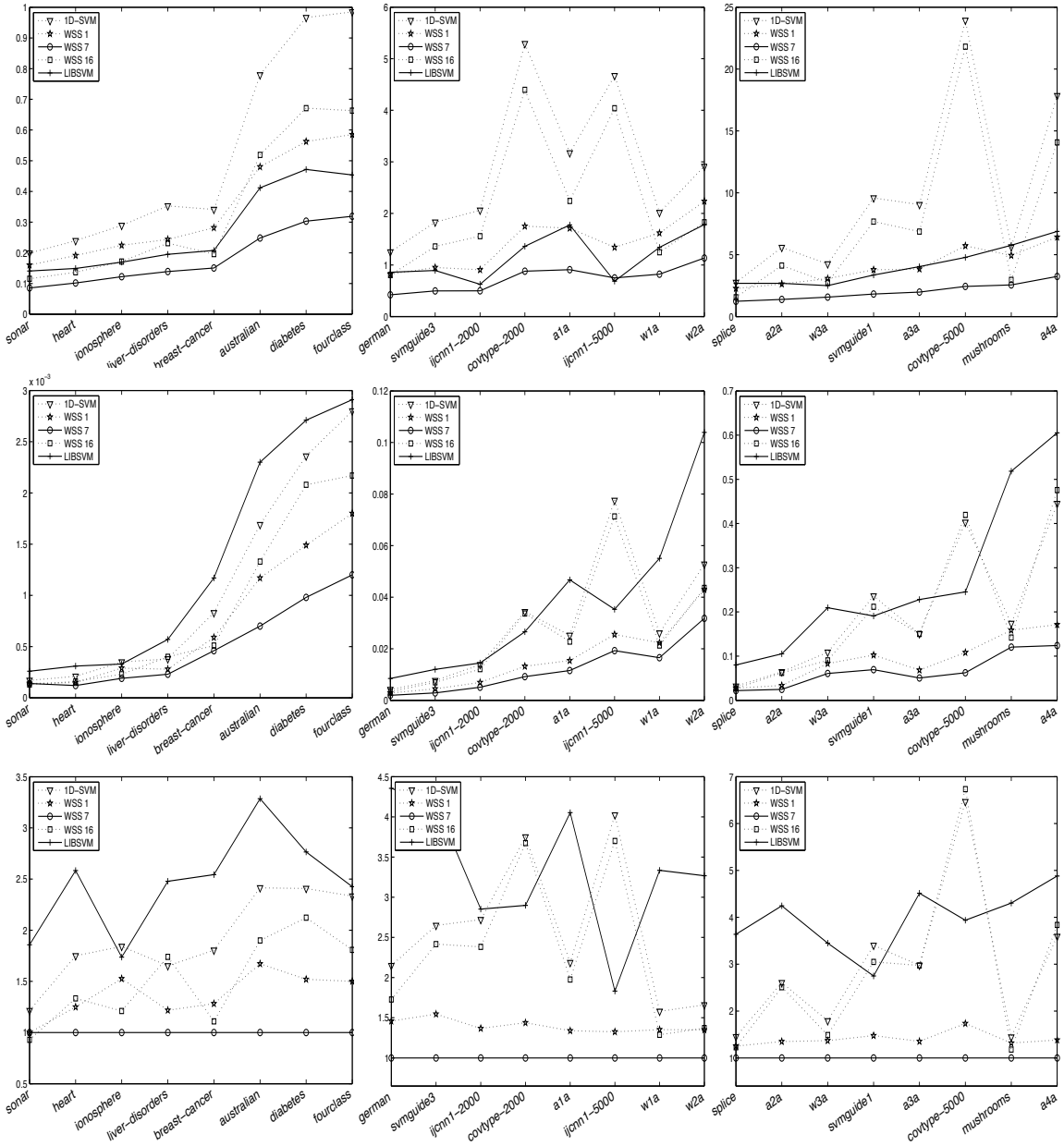


Figure 17: Average computational requirements per grid point of LIBSVM and MVP compared to some other approaches over the entire 10 by 10 grid. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios x/WSS 7 of the run times (bottom). The 2D-MVP approach of WSS 16 is not a good alternative to the 1D-SVM or even the two-dimensional WSS 7. Moreover, although WSS 1 and LIBSVM perform approximately the same number of iterations, their run time is significantly different due to the more expensive working set strategy of LIBSVM.

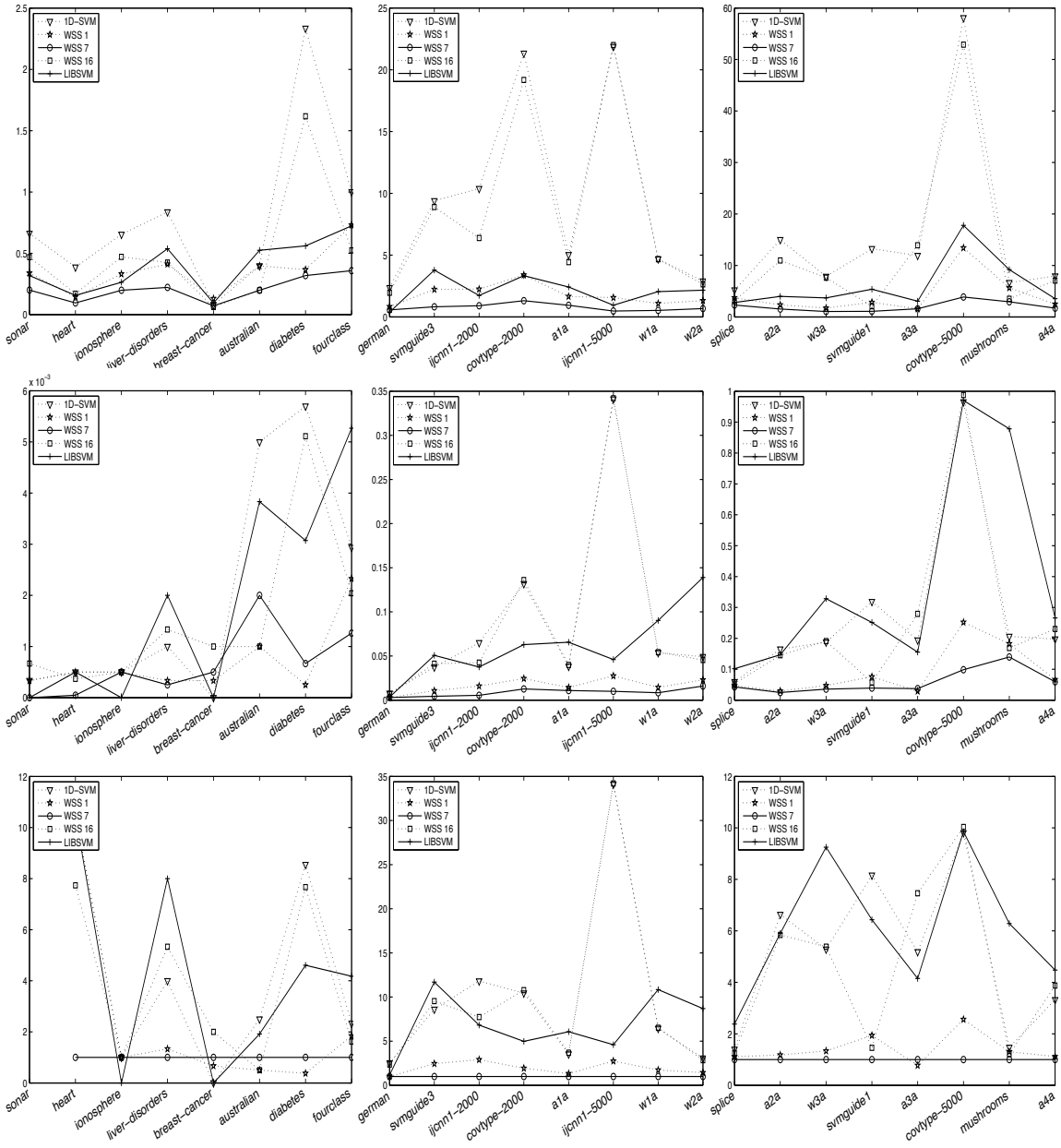


Figure 18: Computational requirements of LIBSVM and MVP compared to some other approaches on the grid points whose cross validation error is not larger than 1.05 the minimal cross validation error. The graphics display the average number of iterations in thousands (top), the run time in seconds (middle), and the ratios x/WSS 7 of the run times (bottom). Again, for the small data sets, the run time measurements are not very reliable. In particular, for the SONAR data set, the average *measured* run time for WSS 7 was 0.00 seconds, and hence the corresponding ratios could not be plotted. Besides that the conclusions of Figure 17 are confirmed.

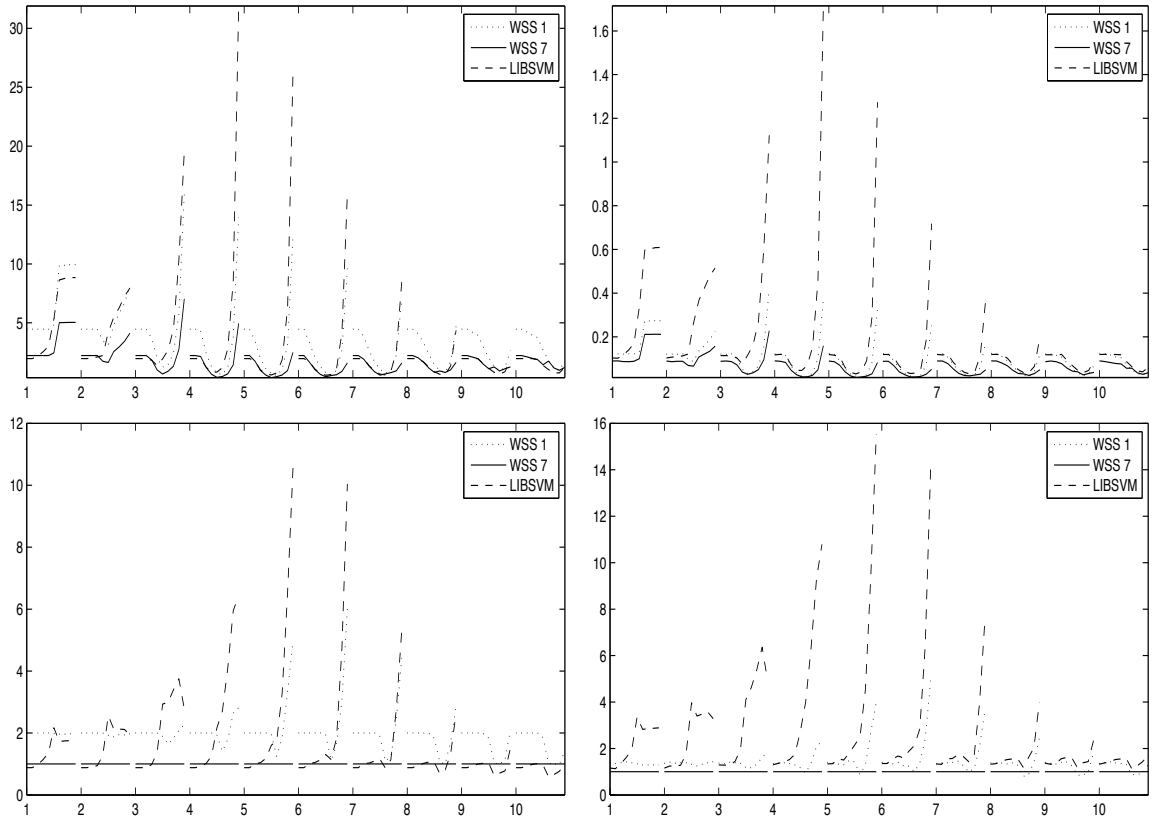


Figure 19: Computational requirements per single grid point of methods based on simple extensions of the 1D-search strategy and LIBSVM on the SVMGUIDE1 data set. The four graphics have the same format as the ones in Figure 10. For flatter kernels, LIBSVM needs less iterations than WSS 7, possibly because it solves a different optimization problem, however the improvement is small in terms of absolute numbers. On the other hand, both WSS 1 and WSS 7 are less sensitive to small λ values in regions with high computational demand.

A.2 Results for the two Different Stopping Criteria

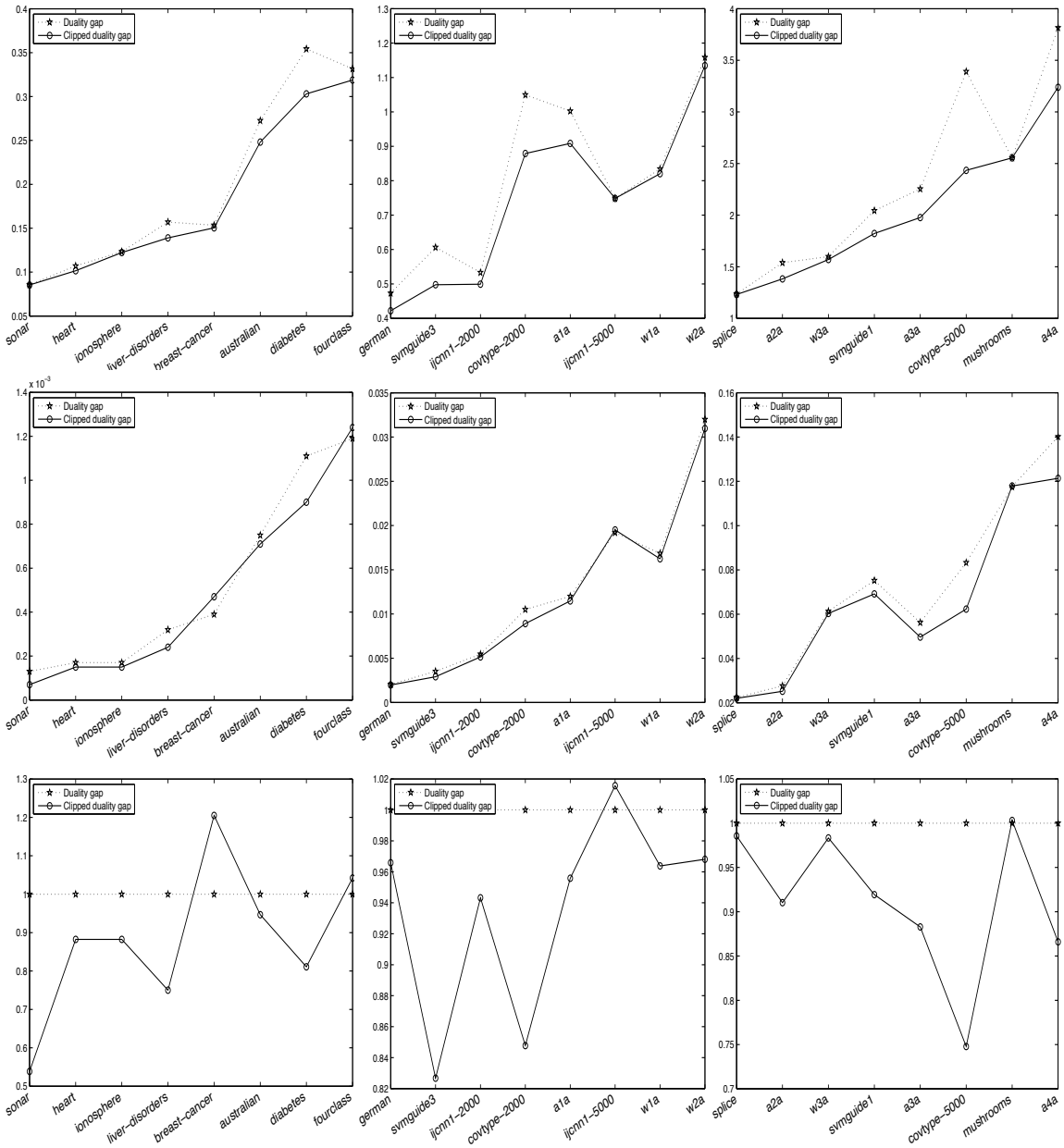


Figure 20: Average computational requirements per grid point of WSS 7 with different stopping criteria. The graphics at the top display the number of iterations in thousands for the 2D-SVM with WSS 7, while the graphics in the middle show the corresponding run time in seconds. The graphics at the bottom display the ratio of run times.

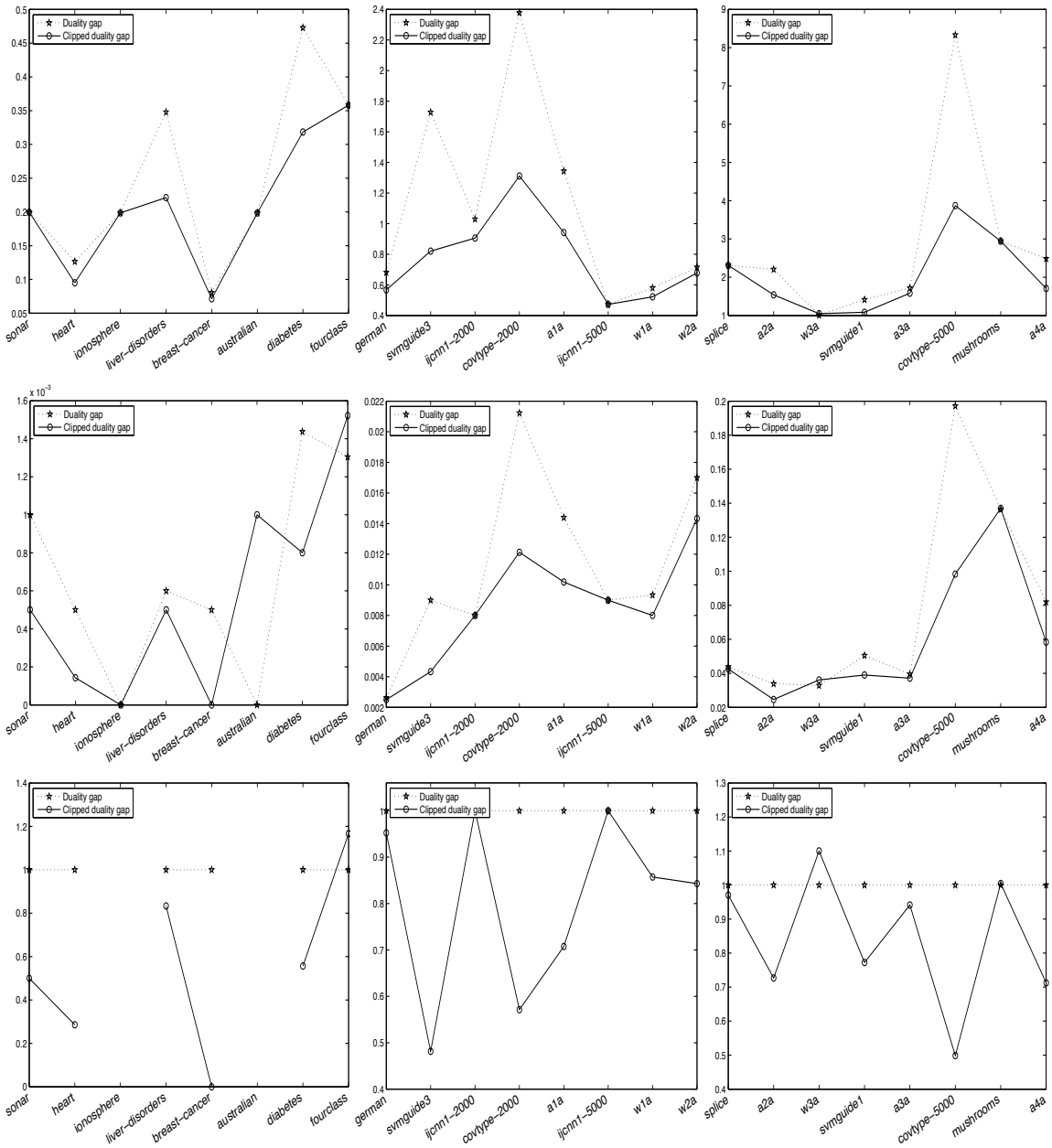


Figure 21: Computational requirements of WSS 7 with different stopping criteria on the grid points whose cross validation error is not larger than 1.05 the minimal cross validation error. Again, the graphics at the top display the number of iterations in thousands for the different stopping criteria applied to the 2D-SVM with WSS 7, while the graphics in the middle show the corresponding run time in seconds. The graphics at the bottom display the ratio of run times, where we note that for some data sets in the bottom left graphic the ratio could not be computed since the *measured* run time was zero.

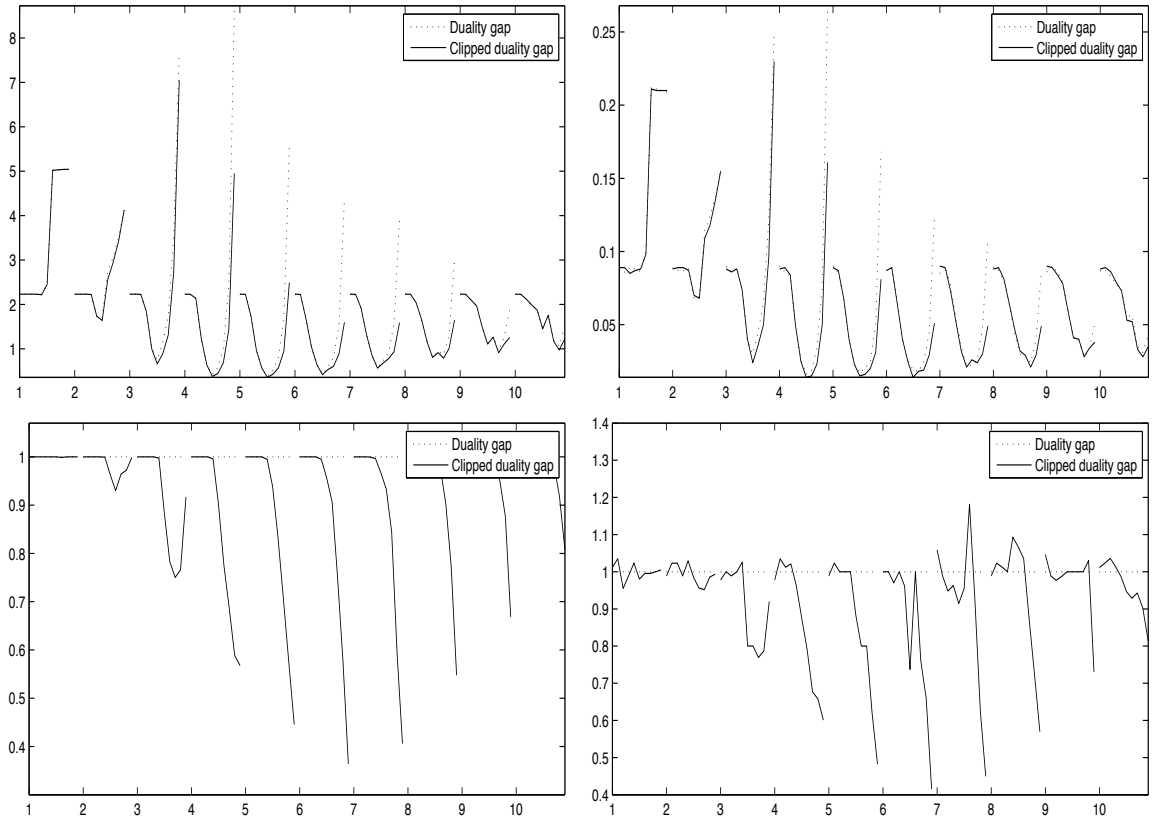


Figure 22: Computational requirements per single grid point of the two stopping criteria for the SVMGUIDE1 data set. The four graphics have the same format as the ones in Figure 10. The graphics at the top display the number of iterations in thousands (left) and the run time in seconds (right), both averaged over the 10 folds, while the graphics at the bottom display the corresponding ratios. The clipped stopping criteria (9) helps for small values of λ, whereas for larger values the behavior is basically identical. Again, some of the roughness in the bottom right graphic can be explained by the resolution of the time measurements. However, the general trend in this graphic is confirmed by the ratio of iterations displayed in the bottom left graphic.

A.3 Results for Different Numbers of Nearest Neighbors

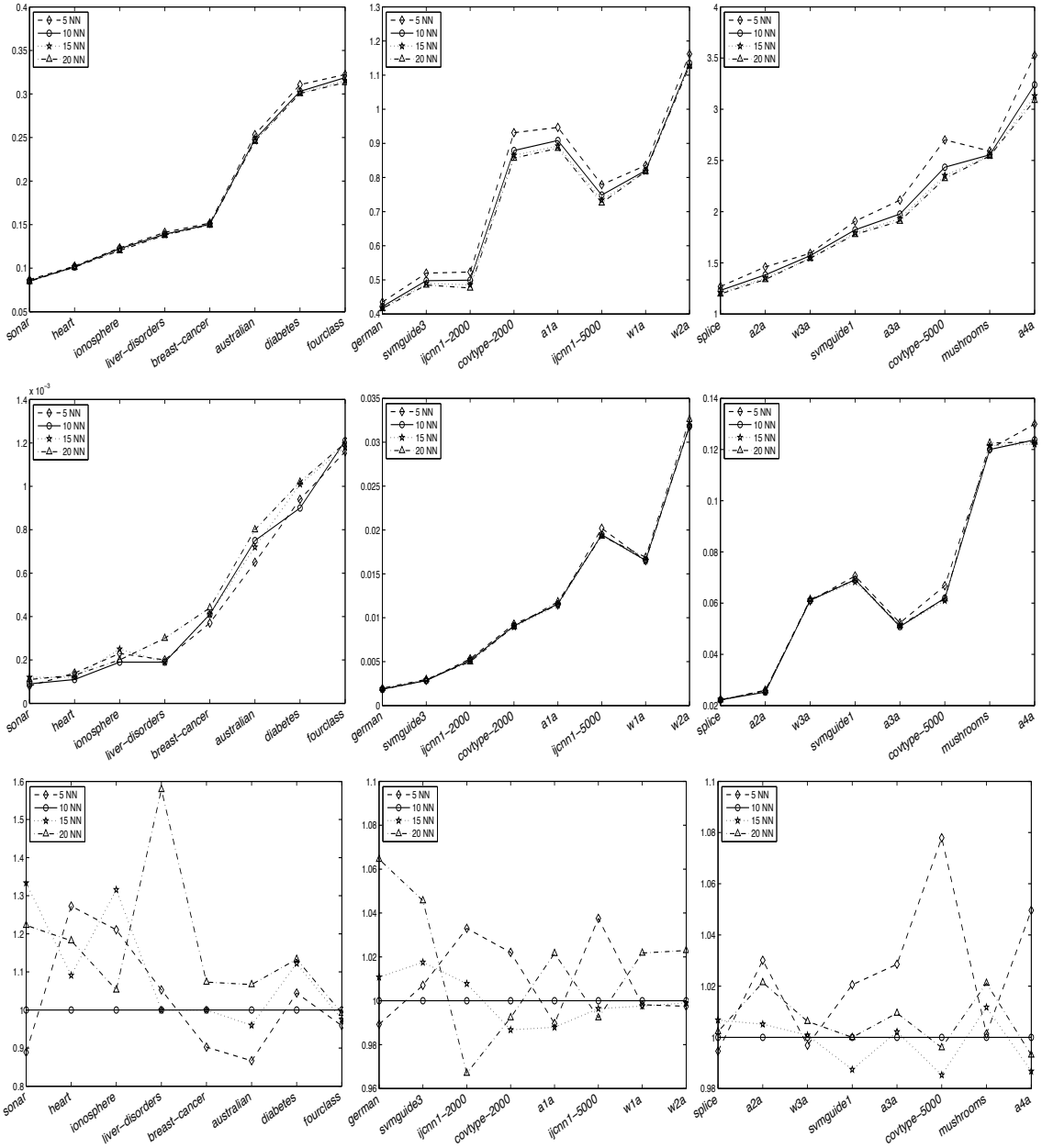


Figure 23: Average computational requirements per grid point of WSS 7 with different numbers N of nearest neighbors. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the corresponding ratios $xNN/10NN$ of the run times (bottom). For $N \geq 10$, the performance is basically identical.

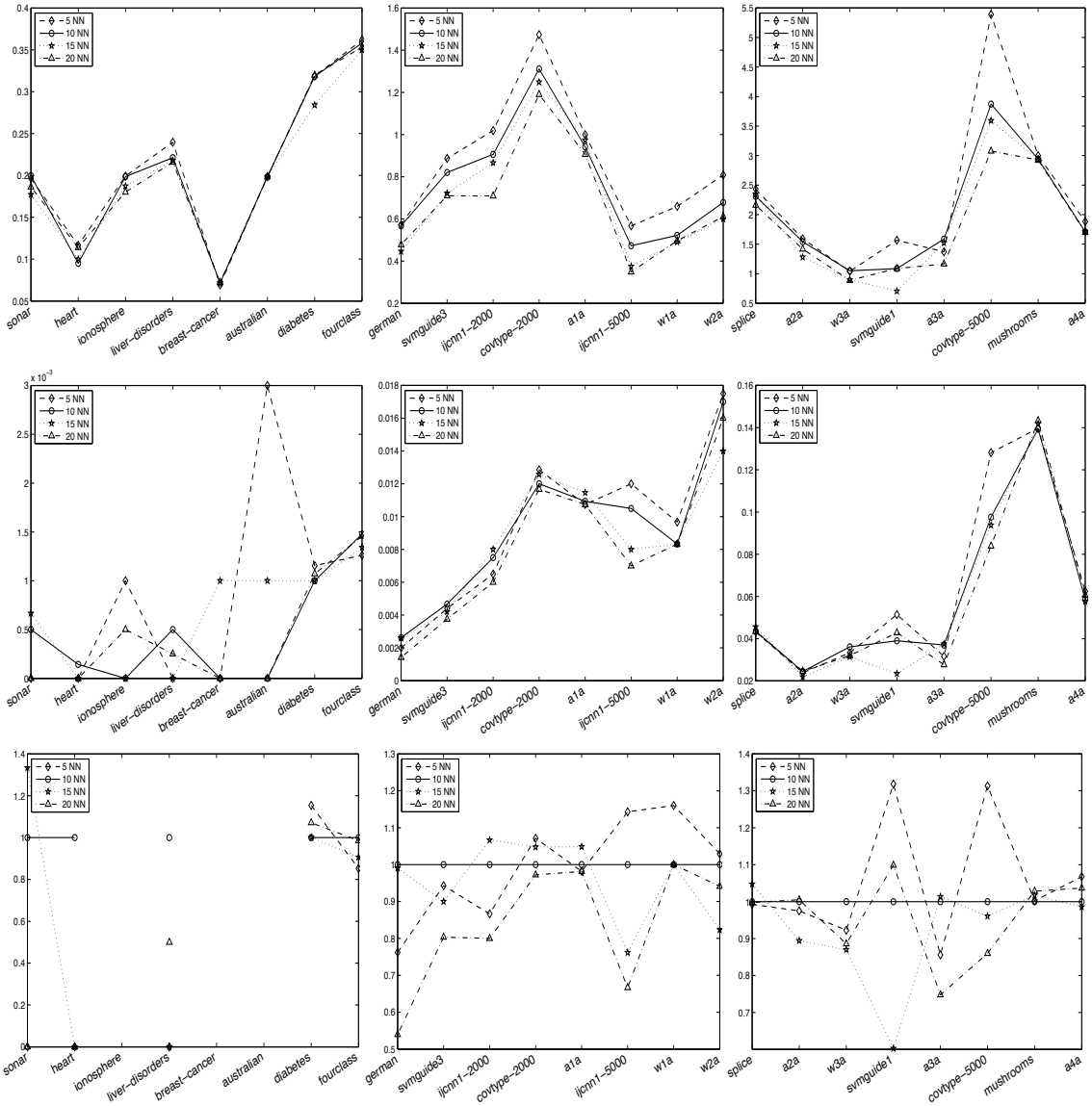


Figure 24: Computational requirements for WSS 7 with different numbers N of nearest neighbors on the grid points whose cross validation error is not larger than 1.05 the minimal cross validation error. The graphics display the average number of iterations in thousands (top), the run time in seconds (middle), and the corresponding ratios $xNN/10NN$ of the run times (bottom). The plots suggest that for grid points with good validation error the number of nearest neighbors has a stronger influence than for the average grid point, yet it is unclear to which extend this effect is caused by different hyper-parameters picked by the different methods.

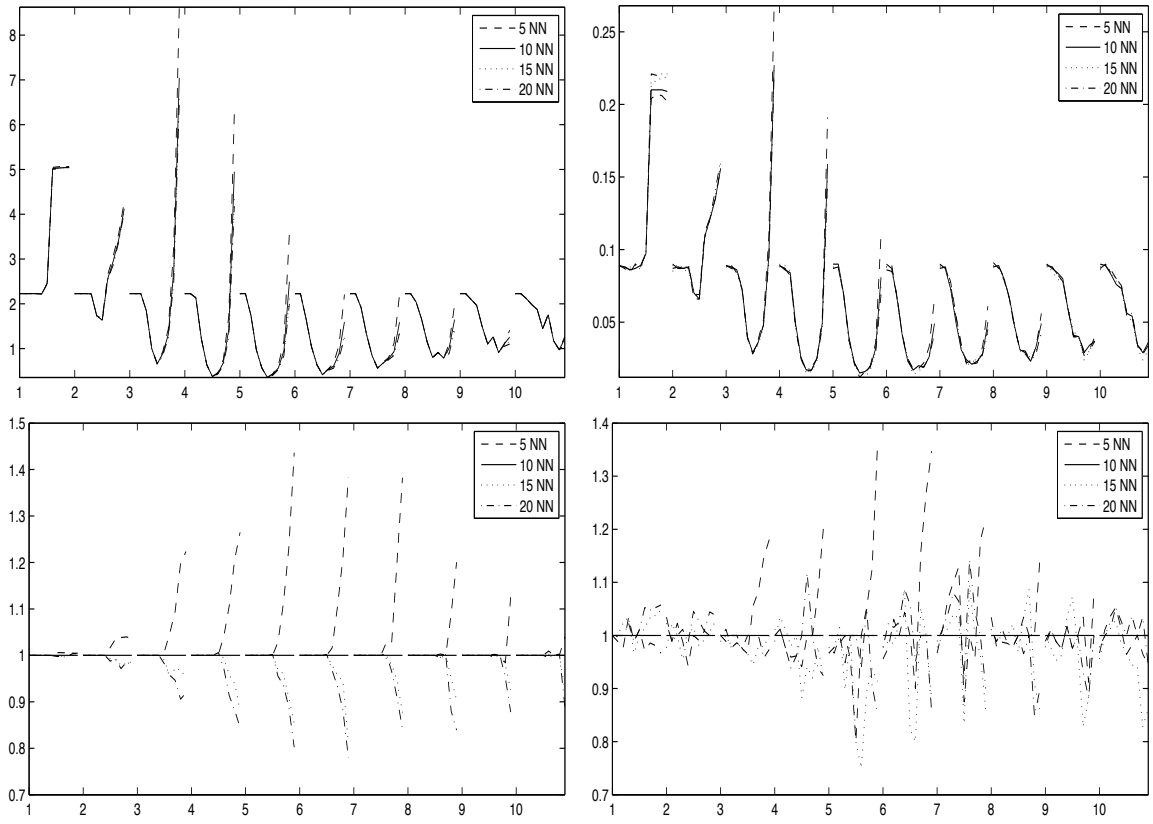


Figure 25: Average computational requirements per grid point of WSS 7 with different numbers N of nearest neighbors for the SVMGUIDE1 data set. The four graphics have the same format as the ones in Figure 10. The graphics at the top display the number of iterations in thousands (left) and the run time in seconds (right), both averaged over the 10 folds, while the graphics at the bottom display the corresponding ratios $xNN/10NN$. Using 5 nearest neighbors clearly results in a worse performance compared to using 10 nearest neighbors. Moreover, compared to $N = 10$ the number of iterations can be further reduced by using more nearest neighbors, but due to unreliable measurements of the run time, it remains somewhat unclear, if this results in significantly shorter run times.

A.4 Results for the Different Initialization Strategies

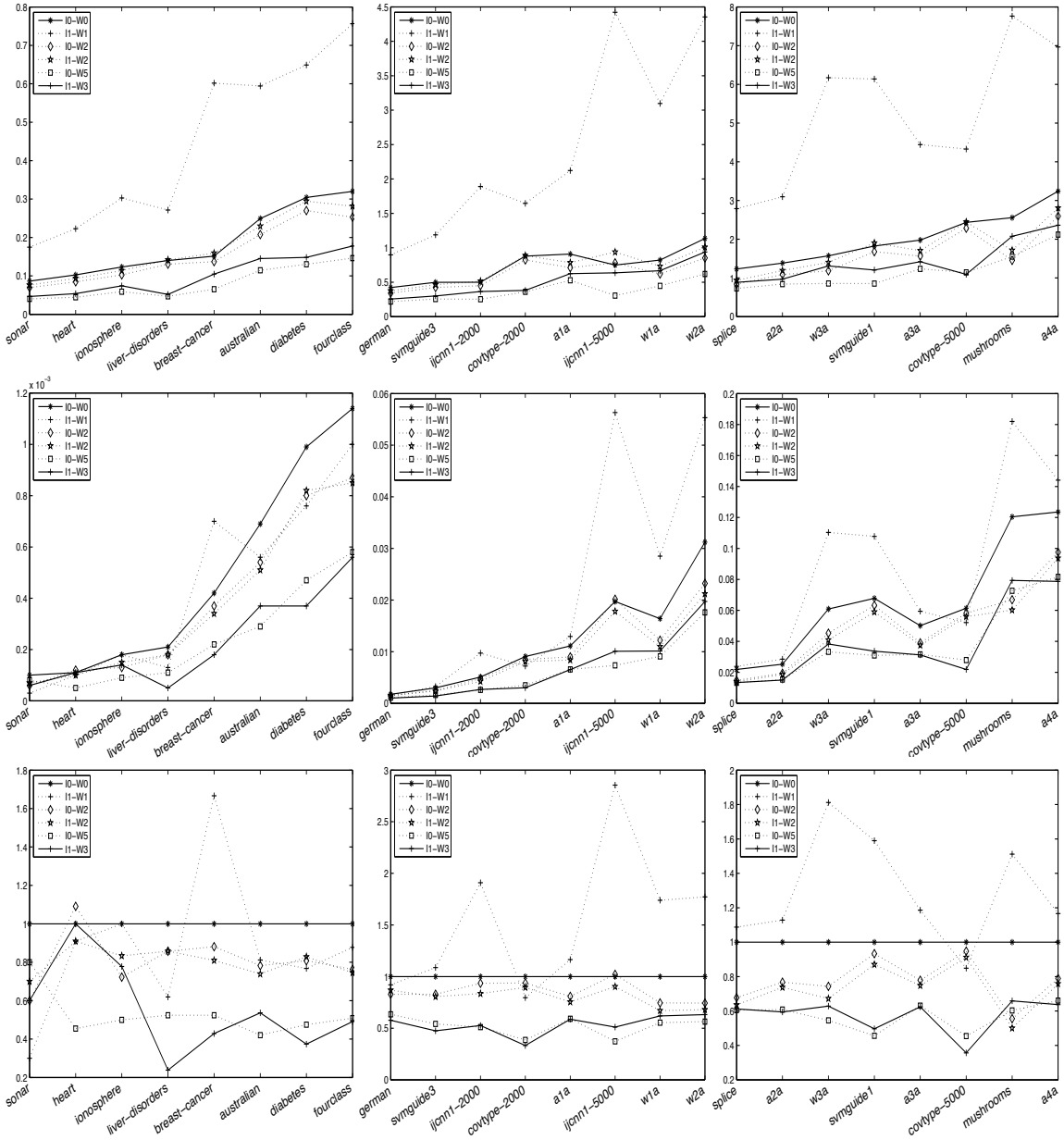


Figure 26: Average computational requirements per grid point of simple initialization strategies for the 2D-SVM with WSS 7. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios Ix-Wy/I0-W0 of the run times (bottom).

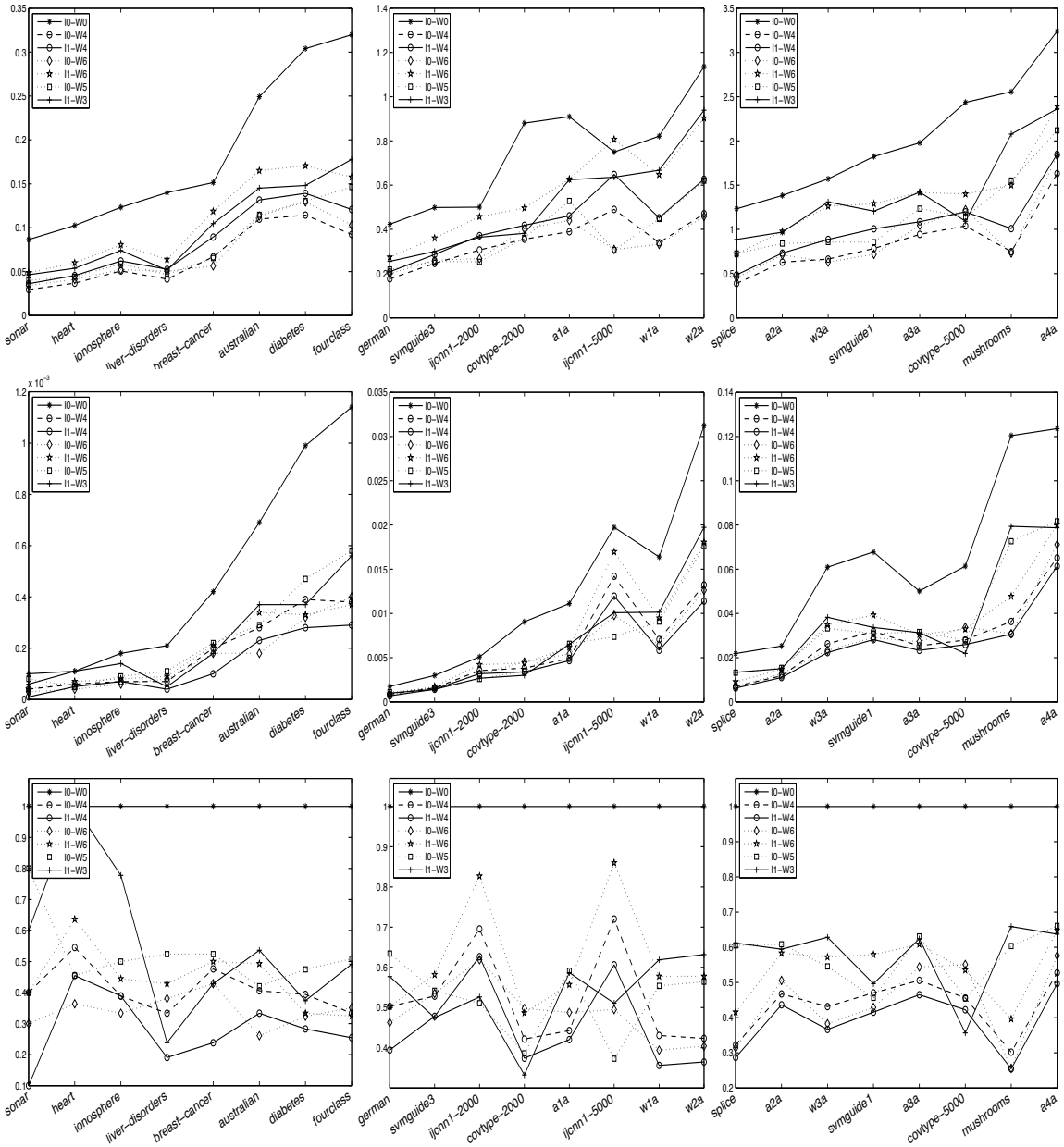


Figure 27: Average computational requirements per grid point of more complex initialization strategies for the 2D-SVM with WSS 7. The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios lx-Wy/l0-W0 of the run times (bottom). Note that, again, the cold start initializations with zeros (l0-plots) need less iterations but in most cases more run time. In almost all cases, the more complicated initialization strategies perform better than the simple warm start approaches. Overall, l0-W4, l1-W4, and l0-W6 are the most efficient methods in terms of run time.

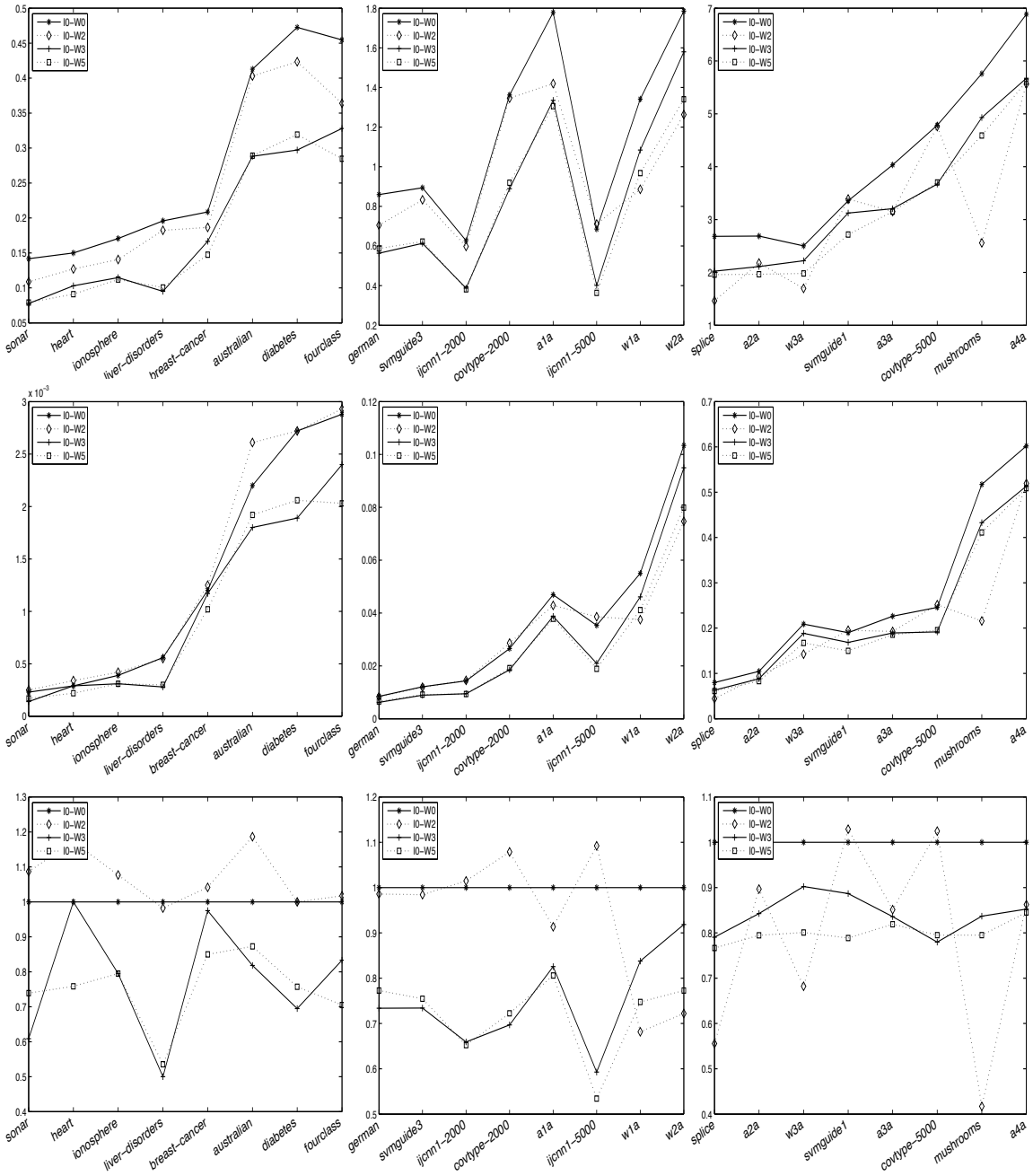


Figure 28: Average computational requirements per grid point of more complex initialization strategies for the LIBSVM for small (left), mid-sized (middle), and relatively large data sets (right). The graphics display the number of iterations in thousands (top), the run time in seconds (middle), and the ratios Ix-Wy/I0-W0 of the run times (bottom). Like for SVMs without offset, using a warm start pays off for this SVM with offset, but the gain is less pronounced.

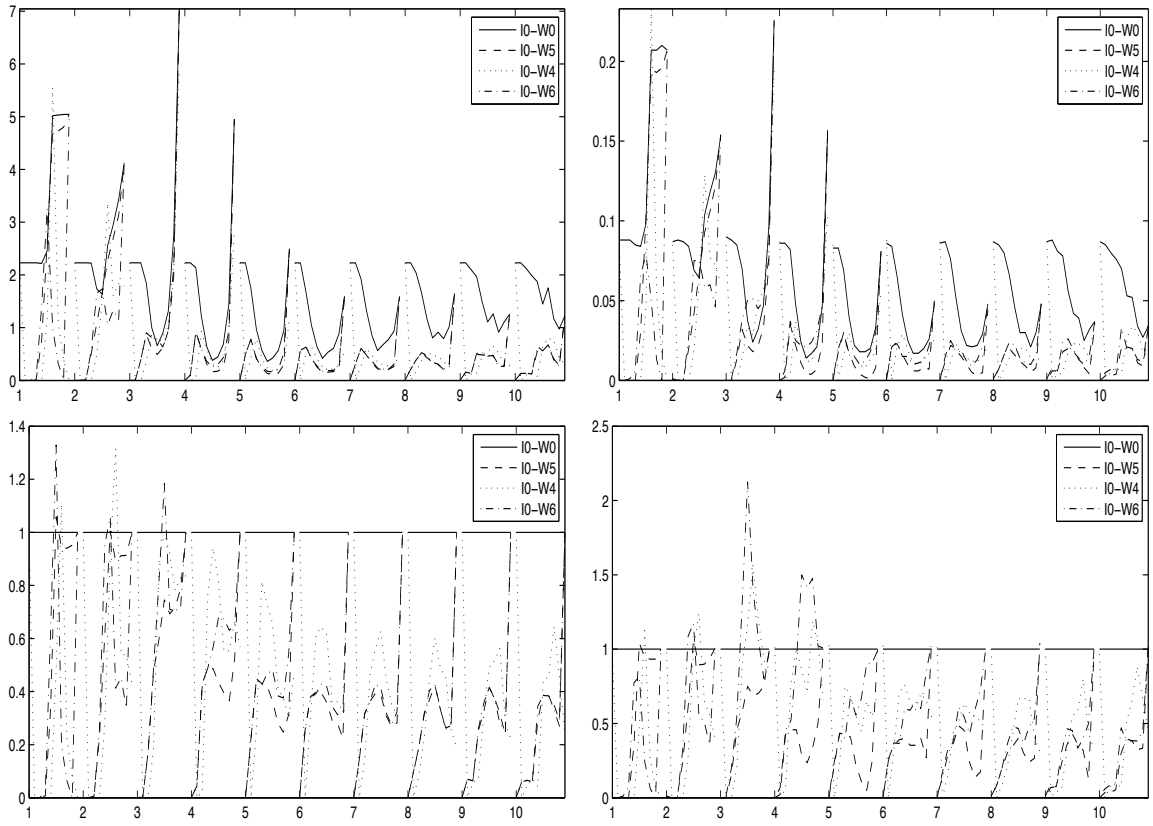


Figure 29: Computational requirements per single grid point of some initialization strategies for the SVMGUIDE1 data set. The four graphics have the same format as the ones in Figure 10. The graphics at the top display the number of iterations in thousands (left) and the run time in seconds (right), both averaged over the 10 folds, while the graphics at the bottom display the corresponding ratios l0-Wx/l0-W0. All warm start strategies perform almost uniformly better than the cold start option l0-W0. Moreover, note that the strategies l0-W5 and l0-W6 start with the smallest λ, that is, at the right hand side of each cell, whereas l0-W4 starts with the largest λ, that is, on the left hand side of each cell.

References

C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. `http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.ps.gz`, 2009.

P.-H. Chen, R.-E. Fan, and C.-J. Lin. A study on SMO-type decomposition methods for support vector machines. *IEEE Trans. Neural Networks*, 17:893–908, 2006.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, 2000.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, New York, 1996.

R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using second order information for training support vector machines. *J. Mach. Learn. Res.*, 6:1889–1918, 2005.

G. Fung and O. L. Mangasarian. Proximal support vector machine classifiers. In *KDD '01: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 77–86, New York, NY, USA, 2001. ACM.

T. Glasmachers and C. Igel. Maximum-gain working set selection for SVMs. *J. Mach. Learn. Res.*, 7:1437–1466, 2006.

C.-W. Hsu and C.-J. Lin. A simple decomposition method for support vector machines. *Mach. Learn.*, 46:291–314, 2002.

C.-W. Hsu and C.-J. Lin. BSVM. `http://www.csie.ntu.edu.tw/~cjlin/bsvm/`, 2006.

T.-M. Huang, V. Kecman, and I. Kopriva. *Kernel Based Algorithms for Mining Huge Data Sets: Supervised, Semi-supervised, and Unsupervised Learning*. Springer, Berlin, 2006.

D. Hush and C. Scovel. Polynomial-time decomposition algorithms for support vector machines. *Mach. Learn.*, 51:51–71, 2003.

D. Hush, P. Kelly, C. Scovel, and I. Steinwart. QP algorithms with guaranteed accuracy and run time for support vector machines. *J. Mach. Learn. Res.*, 7:733–769, 2006.

T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA, 1999.

V. Kecman, T.-M. Huang, and M. Vogt. Iterative single data algorithm for training kernel machines from huge data sets: Theory and performance. In L. Wang, editor, *Support Vector Machines: Theory and Applications*, pages 255–274. Springer Verlag, 2005.

S. Keerthi, V. Sindhwani, and O. Chapelle. An efficient method for gradient-based adaptation of hyperparameters in SVM models. In *Advances in Neural Information Processing Systems 19*, pages 673–680. MIT Press, Cambridge, MA, 2007.

S. S. Keerthi, S. K. Shevade, C. Battacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Comput*., 13:637–649, 2001.

C. J. Lin. On the convergence of the decomposition method for support vector machines. *IEEE Trans. Neural Networks*, 12:1288–1298, 2001.

C. J. Lin. Asymptotic convergence of an SMO algorithm without any assumptions. *IEEE Trans. Neural Networks*, 13:248–250, 2002a.

C. J. Lin. A formal analysis of stopping criteria of decomposition methods for support vector machines. *IEEE Trans. Neural Networks*, 13:248–250, 2002b.

N. List and H.-U. Simon. A general convergence theorem for the decomposition method. In *Proceedings of the 17th Annual Conference on Learning Theory*, pages 363–377. Springer, Heidelberg, 2004.

N. List and H. U. Simon. General polynomial time decomposition algorithms. In S. Ben-David, J. Case, and A. Maruko, editors, *Proceedings of the 18th Annual Conference on Learning Theory, COLT 2005*, pages 308–322. Springer, Heidelberg, 2005.

N. List and H. U. Simon. General polynomial time decomposition algorithms. *J. Mach. Learn. Res*., 8:303–321, 2007.

N. List, D. Hush, C. Scovel, and I. Steinwart. Gaps in support vector optimization. In N. Bshouty and C. Gentile, editors, *Proceedings of the 20th Conference on Learning Theory*, pages 336–348. Springer, New York, 2007.

L.Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *J. Optimization Theory Appl*., 72:7–35, 1992.

O. L. Mangasarian and D. R. Musicant. Lagrangian support vector machines. *J. Mach. Learn. Res*., 1:161–177, 2001.

I. Steinwart. Sparseness of support vector machines. *J. Mach. Learn. Res*., 4:1071–1105, 2003.

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

I. Steinwart, D. Hush, and C. Scovel. An oracle inequality for clipped regularized risk minimizers. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1321–1328. MIT Press, Cambridge, MA, 2007.

M. Vogt. SMO algorithms for support vector machines without bias. Technical report, University of Darmstadt, 2002. `http://www.rtm.tu-darmstadt.de/ehemalige_mitarbeiter/~vogt/docs/vogt_2002_smowob.pdf`.

Journal of Machine Learning Research 12 (2011) 203-239 Submitted 7/09; Revised 7/10; Published 1/11

Logistic Stick-Breaking Process

Lu Ren LR@EE.DUKE.EDU
Lan Du LD53@EE.DUKE.EDU
Lawrence Carin LCARIN@EE.DUKE.EDU
Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708, USA

David B. Dunson DUNSON@STAT.DUKE.EDU
Department of Statistical Science
Duke University
Durham, NC 27708, USA

Editor: David Blei

Abstract

A logistic stick-breaking process (LSBP) is proposed for non-parametric clustering of general spatially- or temporally-dependent data, imposing the belief that proximate data are more likely to be clustered together. The sticks in the LSBP are realized via multiple logistic regression functions, with shrinkage priors employed to favor contiguous and spatially localized segments. The LSBP is also extended for the simultaneous processing of multiple data sets, yielding a hierarchical logistic stick-breaking process (H-LSBP). The model parameters (atoms) within the H-LSBP are shared across the multiple learning tasks. Efficient variational Bayesian inference is derived, and comparisons are made to related techniques in the literature. Experimental analysis is performed for audio waveforms and images, and it is demonstrated that for segmentation applications the LSBP yields generally homogeneous segments with sharp boundaries.

Keywords: Bayesian, nonparametric, dependent, hierarchical models, segmentation

1. Introduction

One is often interested in clustering data that have associated spatial or temporal coordinates. This problem is relevant in a diverse set of applications, such as climatology, ecology, environmental health, real estate marketing, and image analysis (Banerjee et al., 2003). The available spatial or temporal information may be exploited to help infer patterns, clusters or segments in the data. To simplify the exposition, in the following discussion we focus on exploiting *spatial* information, although when presenting results we also consider *temporal* data (Fox et al., 2008).

There have been numerous techniques developed to cluster data, although most of these do not explicitly exploit appended spatial information. One class of state-of-the-art methods employs graphical techniques, such as normalized cuts (Shi and Malik, 2000; Felzenszwalb and Huttenlocher, 2004) and extensions (Zabih and Kolmogorov, 2004). These approaches regard the two-dimensional (2D) data as an undirected weighted graph, and the segmentation is equivalent to finding the minimum cut of the graph, minimizing the between-group disassociation while maximizing the within-group association (Shi and Malik, 2000). Such graph-theoretic methods have attractive computational speed, but do not provide a statistical inference (measure of confidence), and of-

©2011 Lu Ren, Lan Du, Lawrence Carin and David Dunson.

ten one must pre-define the total number of segments/clusters. Further, such graphical techniques are not readily extended to the joint analysis of multiple spatially dependent data sets, with this of interest for the simultaneous analysis of multiple images.

To consider clustering in a nonparametric Bayesian manner, the Dirichlet process (DP) (Blackwell and MacQueen, 1973) has been employed widely (Antoniak, 1974; Escobar and West, 1995; Rasmussen, 2000; Beal et al., 2002). Assume we are given N data points, $\{\mathbf{y}_n\}_{n=1}^N$, with $\mathbf{y}_n$ representing a feature vector; each feature vector is assumed drawn from a parametric distribution $F(\boldsymbol{\theta}_n)$. For each $\mathbf{y}_n$, the DP mixture model is represented as

$$\mathbf{y}_n|\boldsymbol{\theta}_n \sim F(\boldsymbol{\theta}_n), \quad \boldsymbol{\theta}_n|G \overset{iid}{\sim} G, \quad G|\alpha_0, G_0 \sim \mathrm{DP}(\alpha_0 G_0),$$

where α_0 is a non-negative precision parameter and G_0 is the base probability measure. Sethuraman (1994) developed an explicit method for constructing a draw G from a DP:

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\boldsymbol{\theta}_k^*}, \quad \pi_k = V_k \prod_{k'=1}^{k-1} (1 - V_{k'}), \quad V_k \overset{iid}{\sim} \mathrm{Beta}(1, \alpha_0), \quad \boldsymbol{\theta}_k^* \overset{iid}{\sim} G_0. \tag{1}$$

The precision parameter α_0 controls the number of sticks that have appreciable weights, with these weights defining the probability that different $\boldsymbol{\theta}_n$ share the same "atoms" $\boldsymbol{\theta}_k^*$. Since α_0 plays an important role in defining the number of significant stick weights π_k, we typically place a gamma prior on α_0 to allow the data to inform about its value.

The assumption within the DP that the data are exchangeable is generally inappropriate when one wishes to impose knowledge of spatial information (in which each $\mathbf{y}_n$ has an associated spatial location). For example, the data may be represented as $\{\mathbf{y}_n, \mathbf{s}_n\}_{n=1}^N$, in which $\mathbf{y}_n$ is again the feature vector and $\mathbf{s}_n$ represents the spatial location of $\mathbf{y}_n$. Provided with such spatial information, one may wish to explicitly impose the belief that proximate data are more likely to be clustered together.

The spatial location $\mathbf{s}_n$ may be readily considered as an *appended* feature, and the modified feature vectors (data) may then be analyzed via traditional clustering algorithms, like those discussed above. For example, the spatial coordinate has been considered explicitly in recent topic models (Cao and Li, 2007; Wang and Grimson, 2007; Gomes et al., 2008) when applied in image analysis. These previous studies seek to cluster visual words, with such clustering encouraged if the features are spatially proximate. However, these methods may produce spurious clusters that are introduced to better characterize the spatial data likelihood instead of the likelihood of the features conditionally on spatial location (Park and Dunson, 2009). In addition, such approaches require a model for the spatial locations, which is not statistically coherent as these locations are typically fixed by design, and there may be additional computational burden for this extra component.

To address these challenges, and impose spatial information more explicitly, researchers have recently modified the DP construction to manifest spatial-location dependent stick weights. The work of Duan et al. (2007) recently introduced a framework in terms of a hierarchy of Gaussian processes, in which the spatially dependent construction is obtained by thresholding K latent Gaussian processes (GPs); while this is a powerful construction, the use of GPs presents computational challenges (Sudderth and Jordan, 2008). To simplify the model structure, the Dirichlet labeling process (Petrone et al., 2009) has been proposed, in which one thresholds only one latent Gaussian process to regulate spatial dependence. However, the model inference, performed with Markov chain Monte Carlo (MCMC), is inefficient for many large-scale applications. Similar issues are also true for work that has combined the Dirichlet process with a Markov random field (MRF) constraint (Orbanz and Buhmann, 2008).

As an alternative to the above approaches, a kernel stick-breaking process (KSBP) has been proposed (Dunson and Park, 2007), imposing that clustering is more probable if two feature vectors are close in a prescribed (general) space, which may be associated explicitly with spatial position for image processing applications (An et al., 2008). With the KSBP, rather than assuming exchangeable data, the G in (1) becomes a function of spatial location:

$$\begin{aligned} G_{\mathbf{s}} &= \sum_{k=1}^{\infty} \pi_k(\mathbf{s}; V_k, \Gamma_k, \psi)\delta_{\theta_k^*}, \\ \pi_k(\mathbf{s}; V_k, \Gamma_k, \psi) &= V_k K(\mathbf{s}, \Gamma_k; \psi) \prod_{k'=1}^{k-1} \left[1 - V_{k'} K(\mathbf{s}, \Gamma_{k'}; \psi)\right], \\ V_k &\sim \text{Beta}(1, \alpha_0), \quad \theta_k^* \sim G_0, \quad \Gamma_k \sim H_0, \end{aligned} \tag{2}$$

where $K(\mathbf{s}, \Gamma_k; \psi)$ represents a kernel distance between the feature-vector spatial coordinate $\mathbf{s}$ and a local basis location Γ_k associated with the kth stick. As demonstrated when presenting results, the KSBP generally does not yield smooth segments with sharp boundaries.

Instead of thresholding K latent Gaussian processes (Duan et al., 2007) to assign a feature vector to a particular parameter, we introduce a novel non-parametric spatially dependent prior, called the logistic stick-breaking process (LSBP), to impose that it is probable that proximate feature vectors are assigned to the same parameter. The new model is constructed based on a hierarchy of spatial logistic regressions, with sparseness-promoting priors on the regression coefficients. With this relatively simple model form, inference is performed efficiently with variational Bayesian analysis (Beal, 2003), allowing consideration of large-scale problems. Further, for reasons discussed below, this model favors contiguous segments with sharp boundaries, of interest in many applications. The model developed in the paper (Chung and Dunson, 2009), based on a *probit* stick-breaking process, is most closely related to the proposed framework; the relationships between LSBP and the model (Chung and Dunson, 2009) are discussed in detail below.

In addition to exploiting spatial information when performing clustering, there has also been recent research on the simultaneous analysis of *multiple* tasks. This is motivated by the idea that multiple related tasks are likely to share the same or similar attributes (Caruana, 1997; An et al., 2008; Pantofaru et al., 2008). Exploiting the information contained in multiple data sets ("tasks"), model-parameter estimation may be improved (Teh et al., 2005; Pantofaru et al., 2008; Sudderth and Jordan, 2008). Therefore, it is desirable to employ multi-task learning when processing multiple spatially-dependent data (e.g., images), this representing a second focus of this paper.

Motivated by previous multi-task research (Teh et al., 2005; An et al., 2008), we consider the problem of simultaneously processing multiple spatially-dependent data sets. A separate LSBP prior is employed for each of the tasks, and all LSBPs share the same base measure, which is drawn from a DP. Hence, a "library" of model parameters—atoms—is shared across all tasks. This construction is related to the hierarchical Dirichlet process (HDP) (Teh et al., 2005), and is referred to here as a hierarchical logistic stick-breaking process (H-LSBP).

We present example results on two distinct problem classes, underscoring the general utility of the proposed approach. In the first example we consider segmentation of multi-person spoken audio data. In the second application we employ the H-LSBP to simultaneously segment multiple images. In addition to inferring a segmentation of each image, the framework allows sorting and searching among the images.

The remainder of the paper is organized as follows. In Section 2 we introduce the logistic stick-breaking process (LSBP) and discuss its connections with other models. We extend the model to the hierarchical LSBP (H-LSBP) in Section 3. For both the LSBP and H-LSBP, inference is performed via variational Bayesian analysis, as discussed in Section 4. Experimental results are presented in Section 5, with conclusions and future work discussed in Section 6.

2. Logistic Stick-breaking Process (LSBP)

We first consider spatially constrained clustering for a single data set (task). Assume N sample points $\{\mathbf{D}_n\}_{n=1,N}$, where $\mathbf{D}_n = (\mathbf{y}_n, \mathbf{s}_n)$, with $\mathbf{y}_n$ representing the nth feature vector and $\mathbf{s}_n$ its associated spatial location. We draw a set of candidate model parameters, and the probability that a particular space-dependent data sample employs a particular model parameter is defined by a spatially-dependent stick-breaking process, represented by a kernel-based logistic-regression.

2.1 Model Specifications

Assume an infinite set of model parameters $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$. Each observation $\mathbf{y}_n$ is drawn from a parametric distribution $F(\boldsymbol{\theta}_n)$, with $\boldsymbol{\theta}_n \in \{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$. To indicate which parameter in $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$ is associated with the nth sample, a set of indicator variables $\mathbf{Z}_n = \{z_{n1}, z_{n2}, \ldots, z_{n\infty}\}$ are introduced for each $\mathbf{D}_n$, and all the indicator variables are equal to zero or one. Given $\mathbf{Z}_n$, data $\mathbf{D}_n$ is associated with parameter $\boldsymbol{\theta}_k^*$ if $z_{nk} = 1$ and $z_{n\hat{k}} = 0$ for $\hat{k} < k$.

The $\mathbf{Z}_n$ are drawn from a spatially dependent density function, encouraging that proximate $\mathbf{D}_n$ will have similar $\mathbf{Z}_n$, thereby encouraging spatial contiguity. This may be viewed in terms of a spatially dependent stick-breaking process. Specifically, let $p_k(\mathbf{s}_n)$ define the probability that $z_{nk} = 1$, with $1 - p_k(\mathbf{s}_n)$ representing the probability that $z_{nk} = 0$; the spatial dependence of these density functions is made explicit via $\mathbf{s}_n$. The probability that the kth parameter is selected in the above model is $\pi_k(\mathbf{s}_n) = p_k(\mathbf{s}_n)\prod_{\hat{k}=1}^{k-1}[1 - p_{\hat{k}}(\mathbf{s}_n)]$, which is of the same form as a stick-breaking process (Ishwaran and James, 2001) but extends to a spatially dependent mixture model, represented as

$$G_{\mathbf{s}_n} = \sum_{k=1}^{\infty} \pi_k(\mathbf{s}_n)\delta_{\boldsymbol{\theta}_k^*}, \quad \pi_k(\mathbf{s}_n) = p_k(\mathbf{s}_n)\prod_{\hat{k}=1}^{k-1}[1 - p_{\hat{k}}(\mathbf{s}_n)].$$

Here each $p_k(\mathbf{s}_n)$ is defined in terms of a logistic link function (other link functions may also be employed, such as a probit). Specifically, we consider N_c discrete spatial locations $\{\hat{\mathbf{s}}_i\}_{i=1}^{N_c}$ within the domain of the data (e.g., the locations of the samples in $\mathbf{D}_n$). To allow the weights of the different mixture components to vary flexibly with spatial location, we propose a kernel logistic regression for each break of the stick, with

$$\log\left(\frac{p_k(\mathbf{s}_n)}{1 - p_k(\mathbf{s}_n)}\right) = g_k(\mathbf{s}_n) = \sum_{i=1}^{N_c} w_{ki}K(\mathbf{s}_n, \hat{\mathbf{s}}_i; \psi_k) + w_{k0}, \tag{3}$$

where $g_k(\mathbf{s}_n)$ is the linear predictor in the logistic regression model for the kth break and position $\mathbf{s}_n$, and

$$K(\mathbf{s}_n, \hat{\mathbf{s}}_i; \psi_k) = \exp\Big[-\frac{\|\mathbf{s}_n - \hat{\mathbf{s}}_i\|^2}{\psi_k}\Big]$$

is a Gaussian kernel measuring closeness of locations $\mathbf{s}_n$ and $\hat{\mathbf{s}}_i$, as in a radial basis function model (alternative kernel functions may be defined). The kernel basis coefficients are represented as $\mathbf{W}_k =$

$[w_{k0}, w_{k1}, \ldots, w_{kN_c}]'$. A sparseness-promoting prior is chosen for the components of $\mathbf{W}_k$, such that only a relatively small set of w_{ki} will have non-zero (or significant) amplitudes; those spatial regions for which the associated amplitudes are non-zero correspond to regions for which a particular model parameter is expected to dominate in the segmentation (this is similar to the KSBP in (2), which also has spatially localized kernels). The indicator variables controlling allocation to components are then drawn from

$$z_{nk} \sim \text{Bernoulli}[\sigma\big(g_k(\mathbf{s}_n)\big)],$$

where $\sigma(g) = 1/[1+\exp(-g)]$ is the inverse of the logit link in (3).

There are many ways that such sparseness promotion may be constituted, and we have considered two. As one choice, one may employ a hierarchical Student-t prior as applied in the relevance vector machine (Tipping, 2001; Bishop and Tipping, 2000; Bishop and Svensén, 2003):

$$w_{ki} \sim N(w_{ki}|0, \lambda_{ki}^{-1})\text{Gamma}(\lambda_{ki}|a_0, b_0),$$

where shrinkage is encouraged with $a_0 = b_0 = 10^{-6}$ (Tipping, 2001). Alternatively, one may consider a "spike-and-slab" prior (Ishwaran and Rao, 2005). Specifically,

$$w_{ki} \sim \nu_k \mathcal{N}(0, \lambda_k^{-1}) + (1-\nu_k)\delta_0, \quad \nu_k \sim \text{Beta}(\nu_k|c_0, d_0).$$

The expression δ_0 represents a unit point measure concentrated at zero. The parameters (c_0, d_0) are set such that ν_k is encouraged to be close to zero (or we simply fix $\nu_k = \frac{c_0}{c_0+d_0}$), enforcing sparseness in $\boldsymbol{w}_k$; the parameter λ_k is again drawn from a gamma prior, with hyperparameters set to allow a possibly large range in the non-zero values of w_{ki}, and therefore these are *not* set as in the Student-t representation. The advantage of the latter model is that it explicitly imposes that many of the components of $\boldsymbol{w}_k$ are exactly zero, while the Student-t construction imposes that many of the coefficients are close to zero. In our numerical experiments on waveform and image segmentation, we have employed the Student-t construction.

Note that parameter $\boldsymbol{\theta}_k^*$ is associated with an $\boldsymbol{s}$-dependent function $g_k(\boldsymbol{s})$, and there are $K-1$ such functions. The model is constructed such that within a contiguous spatial/temporal region, a particular parameter $\boldsymbol{\theta}_k^*$ is selected, with these model parameters used to generate the observed data.

There are two key components of the LSBP construction: (*i*) sparseness promotion on the w_{ki}, and (*ii*) the use of a logistic link function to define space-dependent stick weights. As discussed further in Section 2.2, these concepts are motivated by the idea of making a particular space-dependent LSBP stick weight $\pi_k(\boldsymbol{s}) = \sigma(g_k(\boldsymbol{s}))\prod_{k'<k}[1-g_k'(\boldsymbol{s})]$ near one within a localized region in space (motivating the sparseness prior on the weights), while also yielding contiguous segments with sharp boundaries (manifested via the logistic).

It is desirable to allow flexibility in the kernel parameter ψ, as this will influence the size of segments that are encouraged (discussed further below). Hence, for each k we draw

$$\psi_k = \psi_{r_k}^*, \quad r_k \sim \text{Mult}(1/\tau, \ldots, 1/\tau),$$

with $\boldsymbol{\Psi}^* = \{\psi_j^*\}_{j=1}^{\tau}$ a library of possible kernel-size parameters; r_k is an index for the one non-zero component of a *single* draw from $\text{Mult}(1/\tau, \ldots, 1/\tau)$. We employ a discrete dictionary of kernel sizes $\boldsymbol{\Psi}^*$ because there is not a conjugate prior for imposition of a continuous distribution of kernel parameters (this is discussed further in Section 4). A draw from this hierarchical prior is denoted concisely as $G_{\mathbf{s}} \sim \text{LSBP}(H, a_0, b_0, \boldsymbol{\Psi}^*)$, where it is assumed that we are using the Student-t prior for

weights $\{\boldsymbol{w}_k\}_{k=1,K-1}$, with a similar representation used for a spike-and-slab prior; note that $G_{\mathbf{s}}$ is defined simultaneously for *all* spatial locations. The model parameters $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$ are assumed drawn from the measure H.

In practice we usually truncate the LSBP to K sticks, as in a truncated stick-breaking process (Ishwaran and James, 2001). With a truncation level K specified, if $z_{nk} = 0$ for all $k = 1,\ldots,K-1$, then $z_{nK} = 1$ so that $\boldsymbol{\theta}_n = \boldsymbol{\theta}_K^*$. The VB analysis yields an approximation to the marginal likelihood of the observed data, which can be used to evaluate the effect of K on the model performance. When presenting results we consider simply setting K to a large value, and also test the model performance with K initialized to different values.

Figure 1 shows the graphical form of the model (using a Student-t sparseness prior), in which $\boldsymbol{\Psi}^*$ represents the discrete set of kernel-width candidates, ψ_k is the kernel width selected for the kth stick, and the prior H takes on different forms depending upon the application. In Figure 1 the $1/\tau$ emphasizes that the candidate kernel widths are selected with uniform probability over the τ candidates in $\boldsymbol{\Psi}^*$.

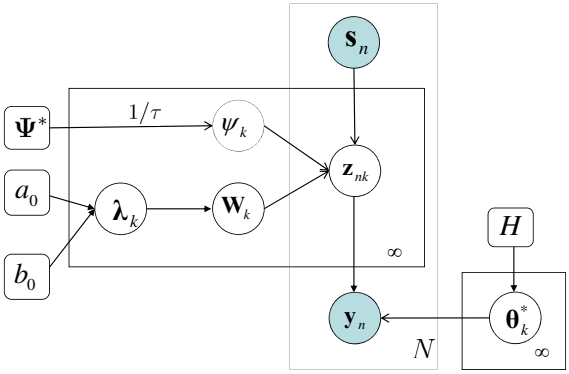


Figure 1: Graphical representation of the LSBP.

2.2 Discussion of LSBP Properties and Relationship to Other Models

The proposed model is motivated by the work (Sudderth and Jordan, 2008), in which multiple draws from a Gaussian process (GP) are employed. Candidate model parameters are associated with each GP draw, and the GP draws serve to constitute a nonparametric gating network, associating particular model parameters with a given spatial position. In the model (Sudderth and Jordan, 2008) the spatial correlation associated with the GP draws induces spatially contiguous segments (a highly spatially correlated gating network), and this may be related to a spatially-dependent stick-breaking process. However, use of the GP produces computational challenges. The proposed LSBP model also manifests multiple space-dependent functions (here $g_k(\boldsymbol{r})$), with associated candidate model parameters $\{\boldsymbol{\theta}_k^*\}_{k=1,K}$. Further, we constitute a spatially dependent gating network that has a stick-breaking interpretation. However, a different and relatively simple procedure is proposed for favoring spatially contiguous segments with sharp boundaries.

At each location $\boldsymbol{s}$ we have a stick-breaking process, with the probability of selecting model parameters $\boldsymbol{\theta}_k^*$ defined as $\pi_k(\boldsymbol{s}) = \sigma(g_k(\boldsymbol{s}))\prod_{k'<k}[1-\sigma(g_{k'}(\boldsymbol{s}))]$. Recall that $g_k(\boldsymbol{s}) = \sum_{i=1}^{N_c} w_{ki}K(\boldsymbol{s},\hat{\boldsymbol{s}}_i;\psi_k) + w_{k0}$, with sparseness favored for coefficients $\{w_{ik}\}_{i=0,N_c}$. Considering first $g_1(\boldsymbol{s})$, note that since most $\{w_{1i}\}_{i=1,N_c}$ are zero or near-zero, the bias w_{10} controls the

stick weight $\pi_1(\boldsymbol{s})$ for all $\boldsymbol{s}$ sufficiently distant from those locations $\hat{\boldsymbol{s}}_i$ with non-zero w_{1i}. Further, if $w_{1i} \gg 0$, $\sigma(g_1(\boldsymbol{s})) \approx 1$ for $\boldsymbol{s}$ in the "neighborhood" of the associated location $\hat{\boldsymbol{s}}_i$; the neighborhood *size* is defined by ψ_1. Hence, those $\{\hat{\boldsymbol{s}}_i\}_{i=1,N_c}$ with associated large $\{w_{1i}\}_{i=1,N_c}$ define localized regions as a function of $\boldsymbol{s}$ over which parameter $\boldsymbol{\theta}_1^*$ is highly probable, with locality defined by kernel scale parameter ψ_1. For those regions of $\boldsymbol{s}$ for which $\pi_1(\boldsymbol{s})$ is *not* near one, there is appreciable probability $1-\pi_1(\boldsymbol{s})$ that model parameters $\{\boldsymbol{\theta}_k^*\}_{k=2,K}$ may be used.

Continuing the generative process, model parameters $\boldsymbol{\theta}_2^*$ are probable where $\pi_2(\boldsymbol{s}) = \sigma(g_2(\boldsymbol{s}))[1-\pi_1(\boldsymbol{s})] \approx 1$. The latter occurs in the vicinity of those $\boldsymbol{s}$ that are distant from $\hat{\boldsymbol{s}}_i$ with large associated w_{1i} (i.e., where $1-\pi_1(\boldsymbol{s}) \approx 1$), while also being near $\hat{\boldsymbol{s}}_i$ with large w_{2i} (i.e., where $\sigma(g_2(\boldsymbol{s})) \approx 1$). We again underscore that w_{20} impacts $\pi_2(\boldsymbol{s})$ for all $\boldsymbol{s}$.

This process continues for increasing k, and therefore it is probable that as k gets large all or almost all $\boldsymbol{s}$ will be associated with a large stick weight, or a large *cumulative* sum of stick weights, such that parameters $\boldsymbol{\theta}_k^*$ become improbable for large k and all $\boldsymbol{s}$.

Key characteristics of this construction are the clipping property of the logistic link function, and the associated fast rise of the logistic. The former imposes that there are contiguous regions (segments) over which the same model parameter has near-unity probability of being used. This encouraging of homogeneous segments is also complemented by sharp segment boundaries, manifested by the fast rise of the logistic. The aforementioned "clipping" property is clearly not distinct to logistic regression. It would apply as well to other binary response link functions, which can be any CDF for a continuous random variable. For example, probit links (Chung and Dunson, 2009) would have the same property, though the logistic has heavier tails than the probit so may have slightly different clipping properties. We have here selected the logistic link function for computational simplicity (it is widely used, for example, in the relevance vector machine Tipping 2001, and we borrow related technology). It is interesting to see how the segmentation realizations differ with the form of link function, with this to be considered in future research.

To give a more-detailed view of the generative process, we consider a one-dimensional example, which in Section 5 will be related to a problem with real data. Specifically, consider a one-dimensional signal with 488 discrete sample points. In this illustrative example $N_c = 98$, defined by taking every fifth sample point for the underlying signal. We wish to examine the generative process of the LSBP prior, in the *absence* of data. For this illustration, it is therefore best to use the spike-and-slab construction, since without any data the Student-t construction will with high probability make all $w_{ki} \approx 0$ (when considering data, and evaluating the posterior, a small fraction of these coefficients are pulled away from zero, via the likelihood function, such that the model fits the data; we reconsider this in Section 5). Further, again for illustrative purposes, we here treat $\{w_{k0}\}_{k=1,K}$ as drawn from a separate normal distribution, *not* from the spike-and-slab prior used for all other components of $\boldsymbol{w}_k$. This distinct handling of $\{w_{k0}\}_{k=1,K}$ has been found unnecessary when processing data, as the likelihood function again imposes constraints on $\{w_{k0}\}_{k=1,K}$. Hence this form of the spike-and-slab prior on $\boldsymbol{w}_k$ is simply employed to illuminate the characteristics of LSBP, with model implementation simplifying when considering data.

In Figure 2 we plot representative draws for $\boldsymbol{w}_k$, $g_k(\boldsymbol{s})$, $\sigma(g_k(\boldsymbol{s}))$ and $\pi_k(\boldsymbol{s})$, for the one-dimensional signal of interest. In this *illustrative* example each ν_k is drawn from Beta$(1,10)$ to encourage sparseness, and those non-zero coefficients are drawn from $\mathcal{N}(0,\lambda)$, with λ fixed to correspond to a standard deviation of 15 (we could also draw each λ_k from a gamma distribution). Each bias term w_{k0} is here drawn iid from $\mathcal{N}(0,\lambda)$. We see from Figure 2 that the LSBP naturally favors localized segments that have near-unity probability of using the same model parameters. This is a

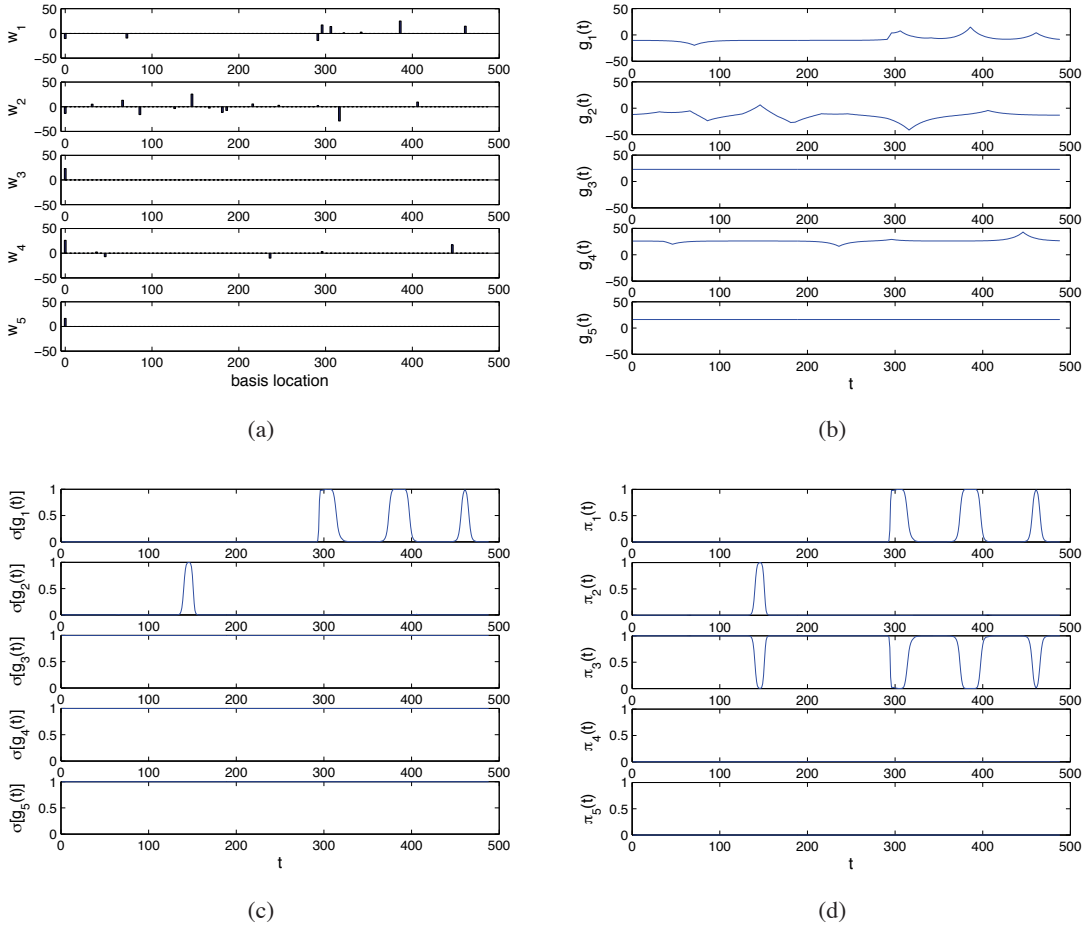


Figure 2: Example draw from a one-dimensional LSBP, using a spike-and-slab construction for model-parameter sparseness. (a) $\boldsymbol{w}_k$, (b) $g_k(t)$, (c) $\sigma_k(t)$, (d) $\pi_k(t)$

typical draw, where we note that for $k \geq 4$ the probability of θ_k^* being used is near zero. While Figure 2 represents a typical LSBP draw, one could also envision other less-desirable draws. For example, if $w_{10} \gg 0$ then $\pi_1(\boldsymbol{s}) \approx 1$ for all $\boldsymbol{s}$, implying that the parameters $\boldsymbol{\theta}_1^*$ is used for all $\boldsymbol{s}$ (essentially no segmentation). Other "pathological" draws may be envisioned. Therefore, we underscore that the data, via the likelihood function, clearly influences the posterior strongly, and the pathological draws supported by the prior in the absence of data are given negligible mass in the posterior.

As further examples, now for two-dimensional signals, Figure 3 considers example draws as a function of the kernel parameter ψ_k. These example draws were manifested via the same process used for the one-dimensional example in Figure 2, now extending $\boldsymbol{s}$ to two dimensions. Figure 3 also shows the dependence of the size of the segments on the kernel parameter ψ_k, which has motivated the learning of ψ_k in a data-dependent manner (based on a finite dictionary of kernel parameters $\boldsymbol{\Psi}^* = \{\psi_j^*\}_{j=1}^{\tau}$). The draws in Figure 3 are similar to those manifested by the GP-based construction (Sudderth and Jordan, 2008), motivating the simple model developed here.

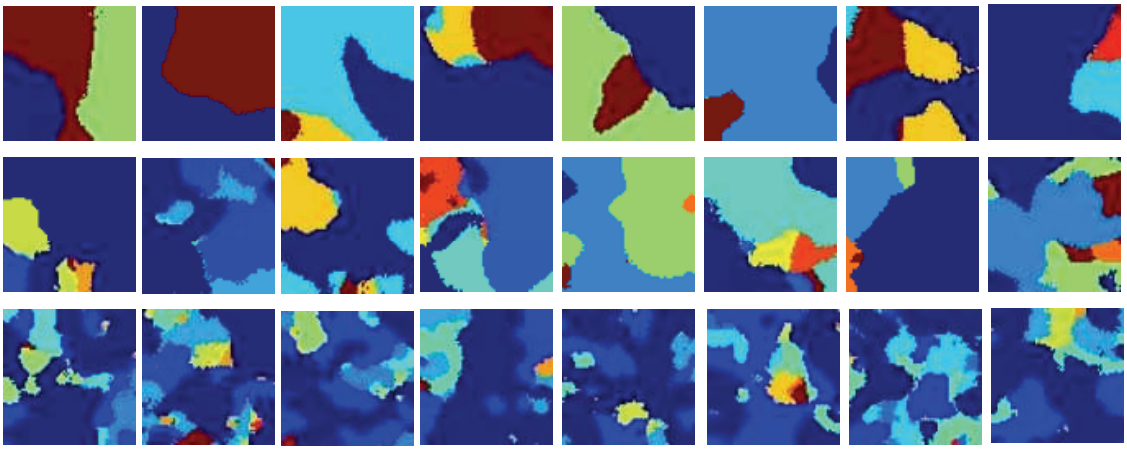


Figure 3: Samples drawn from the spatially dependent LSBP prior, for different (fixed) choices of kernel parameters ψ, applied for each k within the LSBP. In row 1 $\psi = 15$; in row 2 $\psi = 10$; and in row 3 $\psi = 5$. In these examples the spike-and-slab prior has been used to impose sparseness on the coefficients $\{\boldsymbol{w}_k\}_{k=1,K-1}$.

3. Hierarchical LSBP (H-LSBP)

Multi-task learning (MTL) is an inductive transfer framework (Caruana, 1997), with the goal of improving modeling performance by exploiting related information in multiple data sets. We here employ MTL for joint analysis of multiple spatially dependent data sets, yielding a hierarchical logistic stick-breaking process (H-LSBP). This framework models each individual data set (task) with its own LSBP draw, while sharing the same set of model parameters (atoms) across all tasks, in a manner analogous to HDP (Teh et al., 2005). The set of shared model atoms are inferred in the analysis.

The spatially-dependent probability measure for task m, G_m, is drawn from a LSBP with base measure G_0, and G_0 is shared across all M tasks. Further, G_0 is drawn from a Dirichlet process (Blackwell and MacQueen, 1973), and in this manner each task-dependent LSBP shares the same set of discrete atoms. The H-LSBP model is represented as

$$\begin{gathered}\mathbf{y}_{mn}|\boldsymbol{\theta}_{mn} \sim F(\boldsymbol{\theta}_{mn}), \quad \boldsymbol{\theta}_{mn}|G_m \sim G_m, \\ G_m|\{G_0, a_0, b_0, \boldsymbol{\Psi}^*\} \sim \text{LSBP}(G_0, a_0, b_0, \boldsymbol{\Psi}^*), \\ G_0|\gamma, H \sim \text{DP}(\gamma H).\end{gathered}$$

Note that we are assuming a Student-t construction of the sparseness prior within the LSBP, defined by hyperparameters a_0 and b_0.

Assume task $m \in \{1,\ldots,M\}$ has N_m observations, defining the data $\mathbf{D}_m = \{\mathbf{D}_{m1},\ldots,\mathbf{D}_{m(N_m)}\}$. We introduce a set of latent indicator variables $\mathbf{t}_m = \{t_{m1},\ldots,t_{m\infty}\}$ for each task, with

$$t_{mk} \overset{iid}{\sim} \sum_{l=1}^{\infty} \beta_l \delta_l, \quad k = 1,\ldots,\infty, \quad m = 1,\ldots,M, \tag{4}$$

where β_l corresponds to the lth stick weight of the stick-breaking construction of the DP draw $G_0 = \sum_{l=1}^{\infty} \beta_l \delta_{\theta_l^*}$. The indicator variables t_{mk} establish an association between the observations from each task and the atoms $\{\theta_l^*\}_{l=1}^{\infty}$ shared globally; hence the atom $\theta_{t_{mk}}^*$ is associated with LSBP g_k for task m. Accordingly, we may write the probability measure G_m, for position $\mathbf{s}_{mn}$, in the form

$$G_{\mathbf{s}_{mn}} = \sum_{k=1}^{\infty} \pi_{mk}(\mathbf{s}_{mn}) \delta_{\theta_{t_{mk}}^*}.$$

Note that it is possible that in such a draw we may have the same atom used for two different LSBP g_k. This doesn't pose a problem in practice, as the same type of segment (atom) may reside in multiple distinct spatial positions (e.g., of an image), and the different k with the same atom may account for these different regions of the data.

A graphical representation of the proposed hierarchical model is depicted in Figure 4. As in the single-task LSBP discussed in Section 2, a uniform prior is placed on the discrete elements of Ψ^*, and the precision parameter γ for the Dirichlet process is assumed drawn from a gamma distribution $\text{Ga}(e_0, f_0)$. In practice we truncate the number of sticks used to represent G_0, employing $L-1$ draws from the beta distribution, and the length of the Lth stick is $\beta_L = 1 - \sum_{l=1}^{L-1} \beta_l$ (Ishwaran and James, 2001). We also set a truncation level K for each G_m, analogous to truncation of a traditional stick-breaking process.

We note that one may suggest drawing L atoms $\theta_l^* \sim H$, for $l = 1, \ldots, L$, and then simply assigning each of these atoms in the same way to each of $K = L$ g_k in the M LSBPs associated with the M images under test. Although there are K functions g_k in the LSBP, as a consequence of the stick-breaking construction, those with small index k are more probable to be used in the generative process. Therefore, the process reflected by (4) serves to re-order the atoms in an task-dependent manner, such that the important atoms for a given task occur with small index k. In our experiments, we make $K < L$, since the number of different segments/atoms anticipated for any given task is expected to be small relative to the library of possible atoms $\{\theta_l^*\}_{l=1}^{L}$ available across all tasks.

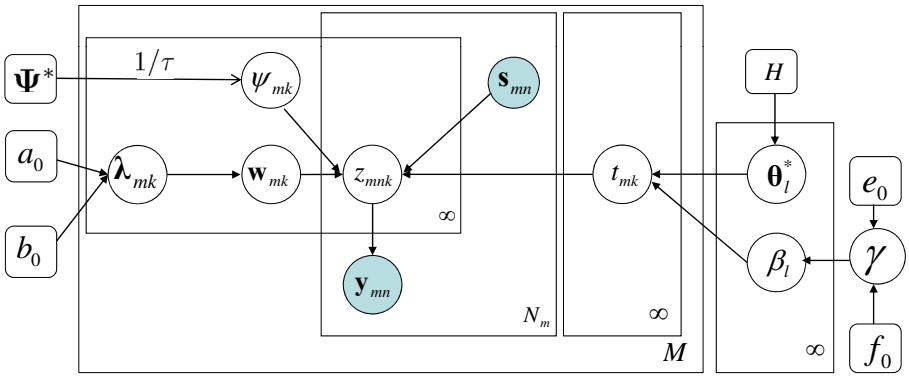


Figure 4: Graphical representation of H-LSBP.

One may view the H-LSBP model as a hierarchy of multiple layers, in terms of a hierarchical tree structure as depicted in Figure 5. In this figure $G_{m1}, \ldots, G_{m(K-1)}$ represent the $K-1$ "gating nodes" within the mth task, and each gating node controls how the data are assigned to the K layers. Thus, the H-LSBP may be viewed as a mixture-of-experts model (Bishop and Svensén, 2003) with

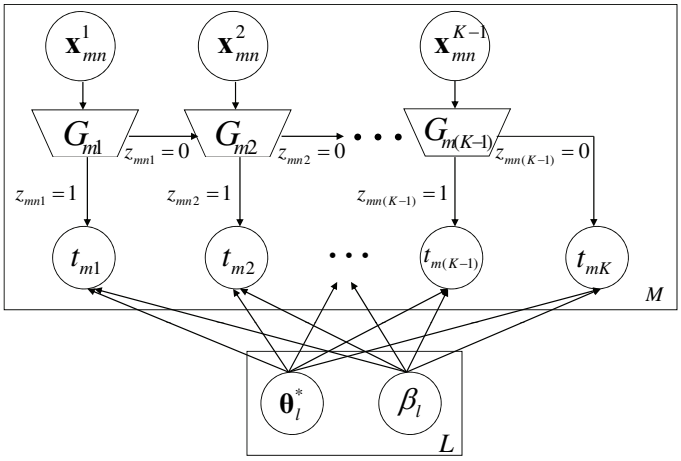


Figure 5: Hierarchical tree structure representation of the H-LSBP, with spatially dependent gating nodes. The parameters $\mathbf{x}_{mn}^k$ are defined as $\mathbf{x}_{mn}^k = \{1, \{K(\mathbf{s}_{mn}, \hat{\mathbf{s}}_{mi}; \psi_{mk})\}_{i=1}^{N_c}\}$.

spatially dependent gating nodes. Given the assigned layer k indicated by $\mathbf{z}_{mn}$, the appearance feature $\mathbf{y}_{mn}$ is drawn from the associated atom $\boldsymbol{\theta}_{t_{mk}}^*$.

3.1 Setting Model Parameters

To implement LSBP, one must set several parameters. As discussed above, the hyperparameters associated with the Student-t prior on w_{ki} are set as $a_0 = b_0 = 10^{-6}$, this corresponding to the settings of the related RVM (Bishop and Tipping, 2000). The number of kernel centers N_c is generally set in a natural manner, depending upon the application. For example, in the audio example considered in Section 5.2, N_c is set to the number of total temporal subsequences used to sample the signal. For the image-processing application, N_c may be set to the number of superpixels used to define space-dependent image features (discussed in more detail when presenting image-segmentation results in Section 5.3). The truncation level K on the LSBP may be set to any large value that exceeds the number of anticipated segments in the image, and the model automatically infers the number of segments in the end. The details are discussed and examined in Section 5 when presenting results. For the H-LSBP results one must also set L, which defines the total library size of model atoms/parameters shared across the multiple data sets. Again, we have found any relatively large setting for L to yield good results, as the nonparametric nature of LSBP manifests a selection of which subset of the L library elements are actually needed for the data under test. This is also examined when presenting experimental results in Section 5.

We must also define a set of possible kernel scales, $\{\psi_j^*\}_{j=1}^{\tau}$. These again are set naturally to define the relative range of scales in the data under test. For example, in the image-segmentation application, we select τ scale levels to cover a range of resolutions characteristic of the images of interest (e.g., defined by the size of the expected segment sizes relative to the overall image size). In the specific audio and image segmentation applications discussed below we explicitly define these parameters, and note that no tuning of these parameters was performed. Our experience is that any "reasonable" set of kernel scales yields very similar results.

The final thing that must be set within the model is the base measure H. For the audio-signal example the data observed at each time point is a real vector, and therefore it is convenient to use a multivariate Gaussian distribution to represent $F(\cdot)$ in (1). Therefore, in that example the observation-model parameters correspond to the mean and covariance of a Gaussian, implying that the measure H should be a Gaussian-Wishart prior (or a Gaussian-Gamma prior, if a diagonal covariance matrix is assumed in the prior). For the image processing application the observed image feature vectors are quantized, and consequently the observation at any point in the image corresponds to a code index. In this case $F(\cdot)$ is represented by a multinomial distribution, and hence H is made to correspond to a Dirichlet distribution. Therefore, one may naturally define H based upon the form of the model $F(\cdot)$, in ways typically employed within such Bayesian models.

4. Model Inference

Markov chain Monte Carlo (MCMC) (Gilks et al., 1998) is widely used for performing inference with hierarchical models like LSBP. For example, many of the previous spatially-dependent mixtures have been analyzed using MCMC (Duan et al., 2007; Dunson and Park, 2007; Nguyen and Gelfand, 2008; Orbanz and Buhmann, 2008). The H-KSBP (An et al., 2008) model is developed based on a hybrid variational inference inference algorithm; however, nearly half of the model parameters still need to be estimated via a sampling technique. Although MCMC is an attractive method for such inference, the computational requirements may lead to implementation challenges for large-scale problems, and algorithm convergence is often difficult to diagnose.

The LSBP model proposed here may be readily implemented via MCMC sampling. However, motivated by the goal of fast and relatively accurate inference for large-scale problems, we consider variational Bayesian (VB) inference (Beal, 2003).

4.1 Variational Bayesian Analysis

Bayesian inference seeks to estimate the posterior distribution of the latent variables $\mathbf{\Phi}$, given the observed data $\mathbf{D}$:

$$p(\mathbf{\Phi}|\mathbf{D},\mathbf{\Upsilon}) = \frac{p(\mathbf{D}|\mathbf{\Phi},\mathbf{\Upsilon})p(\mathbf{\Phi}|\mathbf{\Upsilon})}{\int p(\mathbf{D}|\mathbf{\Phi},\mathbf{\Upsilon})p(\mathbf{\Phi}|\mathbf{\Upsilon})d\mathbf{\Phi}},$$

where the denominator $\int p(\mathbf{D}|\mathbf{\Phi},\mathbf{\Upsilon})p(\mathbf{\Phi}|\mathbf{\Upsilon})d\mathbf{\Phi} = p(\mathbf{D}|\mathbf{\Upsilon})$ is the model evidence (marginal likelihood); the vector $\mathbf{\Upsilon}$ denotes hyper-parameters within the prior for $\mathbf{\Phi}$. Variational Bayesian (VB) inference (Beal, 2003) seeks a variational distribution $q(\mathbf{\Phi})$ to approximate the true posterior distribution of the latent variables $p(\mathbf{\Phi})$. The expression

$$\log p(\mathbf{D}|\mathbf{\Upsilon}) = L(q(\mathbf{\Phi})) + \mathrm{KL}(q(\mathbf{\Phi}) \parallel p(\mathbf{\Phi}|\mathbf{D},\mathbf{\Upsilon}))$$

with

$$L(q(\mathbf{\Phi})) = \int q(\mathbf{\Phi}) \log \frac{p(\mathbf{D}|\mathbf{\Phi},\mathbf{\Upsilon})p(\mathbf{\Phi}|\mathbf{\Upsilon})}{q(\mathbf{\Phi})} d\mathbf{\Phi}, \tag{5}$$

yielding a lower bound for $\log p(\mathbf{D}|\mathbf{\Upsilon})$ so that $\log p(\mathbf{D}|\mathbf{\Upsilon}) \geq L(q(\mathbf{\Phi}))$, since $\mathrm{KL}(q(\mathbf{\Phi}) \parallel p(\mathbf{\Phi}|\mathbf{D},\mathbf{\Upsilon})) \geq 0$. Accordingly, the goal of minimizing the KL divergence between the variational distribution and the true posterior reduces to adjusting $q(\mathbf{\Phi})$ to maximize (5).

Variational Bayesian inference (Beal, 2003) assumes a factorized $q(\mathbf{\Phi})$, typically with the same form as employed in $p(\mathbf{\Phi}|\mathbf{D},\mathbf{\Upsilon})$. With such an assumption, the variational distributions can be

updated iteratively to increase the lower bound. For the LSBP model applied to a single task, as introduced in Section 2.1, we assume

$$q(\mathbf{\Phi}) = \prod_{k=1}^{K} q(\boldsymbol{\theta}_k) \prod_{k'=1}^{K-1} \big[q(\mathbf{w}_{k'}) q(\boldsymbol{\lambda}_{k'}) \prod_{n=1}^{N} q(z_{nk'}) \big],$$

where $q(\boldsymbol{\theta}_k)$ is defined by the specific application. In the audio-segmentation example considered below, the feature vector $\mathbf{y}_n$ may be assumed drawn from a multivariate normal distribution, and the K model parameters are means and precision matrices $\{\boldsymbol{\mu}_k^*, \boldsymbol{\Omega}_k^*\}_{k=1}^K$; accordingly $q(\boldsymbol{\theta}_k)$ is specified as a Normal-Wishart distribution (as is H), $N(\boldsymbol{\mu}_k|\tilde{\boldsymbol{\mu}}_k, \tilde{t}_k^{-1}\boldsymbol{\Omega}_k{}^{-1})\mathrm{Wi}(\boldsymbol{\Omega}_k|\tilde{\boldsymbol{V}}_k, \tilde{d}_k)$. For the rest of the model, $q(\mathbf{w}_{k'}) = \prod_{i=0}^{N_c} N(w_{k'i}|\tilde{m}_{k'i}, \tilde{\Gamma}_{k'i})$, $q(\boldsymbol{\lambda}_{k'}) = \prod_{i=0}^{N_c} Ga(\lambda_{k'i}|\tilde{a}_{k'i}, \tilde{b}_{k'i})$, and $q(z_{nk'})$ has a Bernoulli form $\rho_{nk'}^{z_{nk'}}(1-\rho_{nk'})^{1-z_{nk'}}$ with $\rho_{nk'} = \sigma(g_{k'}(n))$. The factorized representation for $q(\mathbf{\Phi})$ is a function of the hyper-parameters on each of the factors, with these hyper-parameters adjusted to minimize the aforementioned KL divergence.

By integrating over all the hidden variables and model parameters, the lower bound for the log model evidence

$$\begin{aligned} \log p(\mathbf{D}|\boldsymbol{\Upsilon}) &= \log \int p\big(\mathbf{y}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{W}, \boldsymbol{\lambda}, \mathbf{z}\big) \mathrm{d}\mathbf{\Phi} \\ &\geq \int q(\boldsymbol{\theta}, \mathbf{W}, \boldsymbol{\lambda}, \mathbf{z}) \log \frac{p\big(\mathbf{y}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{W}, \boldsymbol{\lambda}, \mathbf{z}\big)}{q(\boldsymbol{\theta}, \mathbf{W}, \boldsymbol{\lambda}, \mathbf{z})} \mathrm{d}\mathbf{\Phi} \\ &= \int q(\boldsymbol{\theta}) q(\mathbf{W}) q(\boldsymbol{\lambda}) q(\mathbf{z}) \log \frac{p\big(\mathbf{y}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{W}, \boldsymbol{\lambda}, \mathbf{z}\big)}{q(\boldsymbol{\theta}) q(\mathbf{W}) q(\boldsymbol{\lambda}) q(\mathbf{z})} \mathrm{d}\mathbf{\Phi} \\ &\equiv LB(q(\mathbf{\Phi})), \end{aligned} \tag{6}$$

is a function of variational distributions $q(\mathbf{\Phi})$. The variational lower bound is optimized by iteratively taking derivatives with respect to the hyper-parameters in each $q(\cdot)$, and setting the result to zero while fixing the hyper-parameters of the other terms. Within each iteration, the lower bound is increased until the model converges.

The difficulty of applying VB inference for this model lies with the logistic-link function, which is not within the conjugate-exponential family. Based on bounding log convex functions, we use a variational bound for the logistic sigmoid function in the form (Bishop and Svensén, 2003)

$$\sigma(x) \geq \sigma(\eta) \exp\big(\frac{x-\eta}{2} - f(\eta)(x^2 - \eta^2)\big), \tag{7}$$

where $f(\eta) = \frac{\tanh(\eta/2)}{4\eta}$ and η is a variational parameter. An exact bound is achieved as $\eta = x$ or $\eta = -x$.

The detailed update equations are omitted for brevity, but are of the form employed in the work (Beal, 2003; Bishop and Svensén, 2003). Like other optimization algorithms, VB inference may converge to a local-optimal solution. However, such a problem can be alleviated by running the algorithm multiple times from different initializations (including varying the truncation level K, and for each case the atom parameters are initialized with k-mean clustering method (Gersho and Gray, 1991) for a fast model convergence) and then using the solution that maximizes the variational model evidence.

4.2 Sampling the Kernel Width

As introduced in Section 2.1, the kernel width ψ_k is inferred for each k. Due to the non-conjugacy of the sigmoid function, we cannot acquire a variational distribution for ψ_k. However, we can sample

it from its posterior distribution or find a maximum a posterior (MAP) solution by establishing a discrete set of potential kernel widths $\boldsymbol{\Psi}^* = \{\psi_j^*\}_{j=1}^{\tau}$, as discussed above. This resulting hybrid variational inference algorithm combines both sampling technique and VB inference, motivated by the Monte Carlo Expectation Maximization (MCEM) algorithm (Wei and Tanner, 1990) and developed by An et al. (2008). The intractable nodes within the graphical model are approximated with Monte Carlo samples from their conditional posterior distributions, and the lower bound of the log model evidence generally has small fluctuations after the model converges (An et al., 2008). A detail on related treatments within variational Bayesian (VB) analysis has been discussed (Winn and Bishop, 2005) (see Section 6.3 of that paper).

Based on the variables $\mathbf{z}_n$, the cluster membership of each data $\mathbf{D}_n$ corresponding to different mixture components $\{\boldsymbol{\theta}_k^*\}_{k=1}^K$ can be specified as

$$\xi_{nk} = \prod_{k'=1}^{k-1} (1 - z_{nk'}) \cdot z_{nk}.$$

Based on the above assumptions, we observe that if $\xi_{nk} = 1$ and the other entries in $\xi_n = [\xi_{n1}, \ldots, \xi_{nK}]$ are equal to zero, then $\mathbf{y}_n$ is assigned to be drawn from $F(\boldsymbol{\theta}_k^*)$.

With the variables $\boldsymbol{\xi}$ introduced and a uniform prior $\mathbf{U}$ assumed on the kernel width $\{\psi_j^*\}_{j=1}^{\tau}$, the posterior distribution for each ψ_k is represented as

$$\begin{aligned} p(\psi_k = \psi_j^* | \cdots) \propto \quad & U_j \cdot \exp\Big\{ \textstyle\sum_n < \xi_{nk} > \Big[< \log\sigma\big(g_k^j(\mathbf{s}_n)\big) > \Big] \Big\} \\ & \cdot \exp\Big\{ \textstyle\sum_n \sum_{l>k} < \xi_{nl} > \Big[< \log\Big(1 - \sigma\big(g_k^j(\mathbf{s}_n)\big)\Big) > \Big] \Big\}, \end{aligned} \tag{8}$$

where U_j is the jth component of $\mathbf{U}$, $< \cdot >$ represents the expectation with the associated random variables, $g_k^j(\mathbf{s}_n) = \sum_{i=1}^{N_c} w_{ki} K(\mathbf{s}_n, \hat{\mathbf{s}}_i; \psi_j^*) + w_{k0}$ with $j = 1, \ldots, \tau$.

With the definition $\mathbf{x}_n^j = \Big[1, K(\mathbf{s}_n, \hat{\mathbf{s}}_1; \psi_j), \ldots, K(\mathbf{s}_n, \hat{\mathbf{s}}_{N_c}; \psi_j)\Big]$, it can be verified that

$$\log\big(1 - \sigma(g_k^j(\mathbf{s}_n))\big) = -\mathbf{W}_k^T \mathbf{x}_n^j + \log\sigma(g_k^j(\mathbf{s}_n)). \tag{9}$$

Inserting (9) into the kernel width's posterior distribution, (8) can be reduced to

$$\begin{aligned} p(\psi_k = \psi_j^* | \cdots) \propto \quad & U_j \cdot \exp\Big\{ \textstyle\sum_n < \xi_{nk} > \Big[< \log\sigma\big(g_k^j(\mathbf{s}_n)\big) > \Big] \Big\} \\ & \cdot \exp\Big\{ \textstyle\sum_n \sum_{l>k} < \xi_{nl} > \Big[- < \mathbf{W}_k >^T \mathbf{x}_n^j + < \log\sigma\big(g_k^j(\mathbf{s}_n)\big) > \Big] \Big\}, \end{aligned}$$

in which $< \log\sigma\big(g_k^j(\mathbf{s}_n)\big) >$ is calculated via the variational bound of the logistic sigmoid function in (7):

$$< \log\sigma\big(g_k^j(\mathbf{s}_n)\big) > \geq \log\sigma(\eta_{nk}) + \frac{1}{2}(< g_k^j(\mathbf{s}_n) > - \eta_{nk}) + f(\eta_{nk})(< \{g_k^j(\mathbf{s}_n)\}^2 > - \eta_{nk}^2),$$

in which

$$\begin{aligned} & < g_k^j(\mathbf{s}_n) > = < \mathbf{W}_k >^T \mathbf{x}_n^j, \quad < \{g_k^j(\mathbf{s}_n)\}^2 > = {\mathbf{x}_n^j}^T < \mathbf{W}_k \mathbf{W}_k^T > \mathbf{x}_n^j \\ & \mathbf{x}_n^j = \big[1, K(\mathbf{s}_n, \hat{\mathbf{s}}_1; \psi_j^*), \ldots, K(\mathbf{s}_n, \hat{\mathbf{s}}_{N_c}; \psi_j^*)\big] \end{aligned} \tag{10}$$

As $\eta_{nk} = \sqrt{{\mathbf{x}_n^j}^T < \mathbf{W}_k \mathbf{W}_k^T > \mathbf{x}_n^j}$, the bound holds and the Equation (10) is reduced to:

$$< \log\sigma\big(g_k^j(\mathbf{s}_n)\big) > = \log\sigma(\eta_{nk}) + \frac{1}{2}(< \mathbf{W}_k >^T \mathbf{x}_n^j - \eta_{nk}).$$

From the above discussion, we have the following update equation for the kernel widths. For each specific k from $k = 1, \ldots, K$:

$$\psi_k = \psi^*_{r_k}, \quad r_k \sim \text{Mult}(p_{k1}, \ldots, p_{k\tau}),$$
$$p_{kj} = \frac{p(\psi_k = \psi^*_j)}{\sum_{l=1}^{\tau} p(\psi_k = \psi^*_l)}.$$

We sample the kernel width based on the multinomial distribution from a given discrete set in each iteration, or we can set the kernel width by choosing one with the largest probability component. The latter one can be regarded as a MAP solution by specifying a discrete prior. Both of the two methods get similar results in our experiments. Therefore, we only present the result by sampling the kernel widths in our experimental examples.

Because of the sampling of the kernel width within the VB iterations, the lower bound shown in (6) does not monotonically increase in general. Until the model converges, the lower bound generally has small fluctuations, as shown when presenting experimental results.

For the hierarchical logistic stick-breaking process (H-LSBP), we adopt a similar inference technique to that employed for LSBP, with the addition of updating the parameters of the Dirichlet process. We omit those details here, but summarize the model update equations in the Appendix.

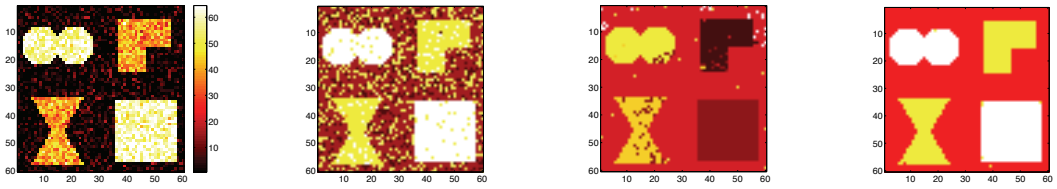
5. Experimental Results

The LSBP model proposed here may be employed in many problems for which one has spatially-dependent data that must be clustered or segmented. Since the spatial relationships are encoded via a kernel distance measure, the model can also be used to segment time-series data. Below we consider three examples: (*i*) a simple "toy" problem that allows us to compare with related approaches in an easily understood setting, (*ii*) segmentation of multiple speakers in an audio signal, and (*iii*) segmentation of images. When presenting (*iii*), we first consider processing single images, to demonstrate the quality of the segmentations, and to provide more details on the model. We then consider joint segmentation of multiple images, with the goal of inferring relationships between images (of interest for image sorting and search). In all examples the Student-t construction is used to impose the model sparseness, and all model coefficients (including the bias terms) are drawn from the same prior.

5.1 Simulation Example

In this example the feature vector $\mathbf{y}_n$ is the intensity value of each pixel, and the pixel location is the spatial information $\mathbf{s}_n$. Each observation is assumed to be drawn from a spatially dependent Gaussian mixture (i.e., $F(\cdot)$ is a Gaussian). A comparison is made between the proposed LSBP, the Dirichlet process (DP), and the kernel stick-breaking process (KSBP), and in all cases VB inference is performed; for the KSBP, we use the same model as considered by An et al. (2008), and this simple example was also taken from that paper. The data are shown in Figure 6(a), in which four distinct contiguous sub-regions reside in a background, with a color bar encoding the pixel amplitudes. Each pixel is drawn from a Gaussian distribution with a standard deviation of 10; the two pairs of contiguous regions are generated respectively from the Gaussian distributions with mean intensities equal to 40 and 60, and the background has a mean of 5 (An et al., 2008). In the LSBP, DP, and KSBP analyses, we do not set the number of clusters *a priori* and the models infer the number of clusters automatically from the data. Therefore, we fixed the truncation level to $K = 10$

for all models, and the clustering results are shown in Figure 6, with different colors representing the cluster index (mixture component to which a data sample is assigned).

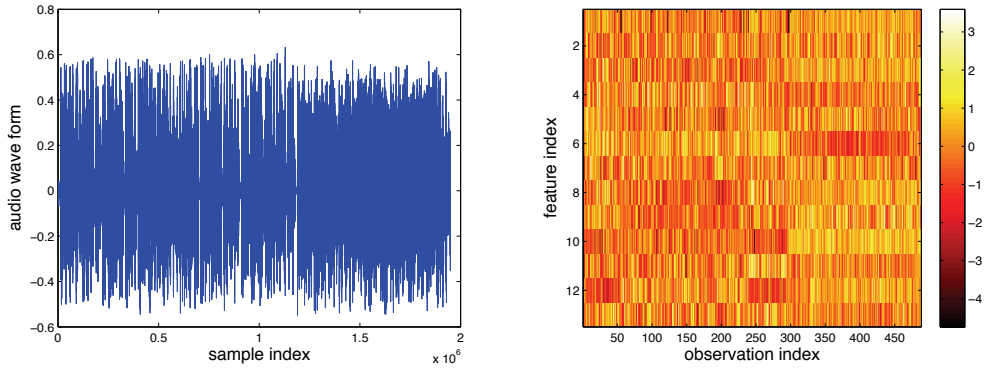


(a) (b) (c) (d)

Figure 6: Segmentation results for the simulation example. (a) original image, (b) DP, (c) KSBP, (d) LSBP

Compared with DP and KSBP, the proposed LSBP shows a much cleaner segmentation in Figure 6(d), as a consequence of the imposed favoring of contiguous segments. We also note that the proposed model inferred that there were only three important k (three dominant sticks) within the observed data, consistent with the representation in Figure 6(a).

5.2 Segmentation of Audio Waveforms



(a) (b)

Figure 7: Original audio waveform, (a), and representation in terms of MFCC features, (b).

With the kernel in (2.1) specified in a temporal (one-dimensional) space, the proposed LSBP is naturally extended to segmentation of sequential data, such as for speaker diarization (Ben et al., 2004; Tranter and Reynolds, 2006; Fox et al., 2008). Provided with a spoken document consisting of multiple speakers, speaker diarization is the process of segmenting the audio signal into contiguous temporal regions, and within a given region a particular individual is speaking. Further, one also wishes to group all temporal regions in which a specific individual is speaking.

We assume the acoustic observations at different times are drawn from a Gaussian mixture model (each generating Gaussian ideally corresponds to a speaker ID). Within LSBP and KSBP, the observations of adjacent temporal points are encouraged to be drawn from the same Gaussian, since they are with high probability assumed to be generated from the same source (speaker). The total number of speakers is unknown in advance, and is inferred from the data. An alternative approach,

to which we compare, is a sticky HMM (Fox et al., 2008), in which the speech is represented by an HMM with Gaussian state-dependent emissions; to associate a given speaker with a particular state, the states are made to be persistent, or "sticky', with the state-dependent degree of stickiness also inferred.

We consider identification of different speakers from a recording of broadcast news, which may be downloaded with its ground truth.[1] The spoken document has a length of 122.05 seconds, and consists of three speakers. Figure 7(a) presents the audio waveform with a sampling rate of 16000 Hz. The ground truth indicates that Speaker 1 talked within the first 13.77 seconds, followed by Speaker 2 until the 59.66 second, then Speaker 1 began to talk again until 74.15 seconds, and Speaker 3 followed and speaks until the end.

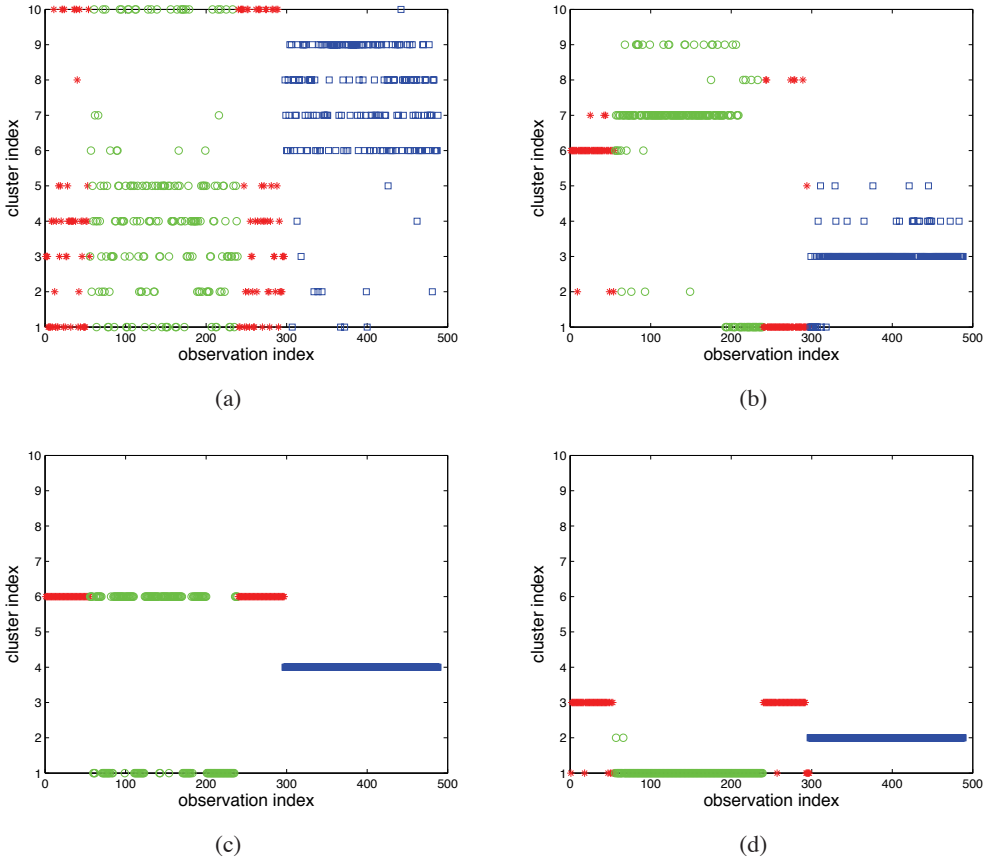


Figure 8: Segmentation results for the audio recording. The colored symbols denote the ground truth: red represents Speaker 1, green represents Speaker 2, blue represents Speaker 3. Each MFCC feature vector is assigned to a cluster index ($K = 10$), with the index shown along the vertical axis. (a) DP, (b) KSBP, (c) sticky HMM using VB inference, (d) LSBP

For the feature vector, we computed the first 13 Mel Frequency Cepstral Coefficients (MFCCs) (Ganchev et al., 2005) over a 30 ms window every 10 ms, and defined the observations as averages over every 250 ms block, without overlap. We used the first 13 MFCCs because the high frequency

1. Recording can be downloaded from `http://www.itl.nist.gov/iad/mig//tests/rt/2002/index.html`.

content of these features contained little discriminative information (Fox et al., 2008). The software that we used to extract the MFCCs feature can be downloaded online.[2] There are 488 feature vectors in total, shown in Figure 7(b); the features are normalized to zero mean and the standard deviation is made equal to one.

To apply the DP, KSBP and LSBP Gaussian mixture models on this data, we set the truncation level as $K = 10$. To calculate the temporal distance between each pair of observations, we take the observation index from 1 to 488 as the location coordinates in (2.1) for $\mathbf{s}$. The potential kernel-width set is $\Psi^* = \{50, 100, \ldots, 1000\}$ for LSBP and KSBP; note that these are the same range of parameters used to present the generative model in Figure 2. The experiment shows that all the models converge after 20 VB iterations.

For the sticky HMM, we employed two distinct forms of posterior computation: (*i*) a VB analysis, which is consistent with the methods employed for the other models; and (*ii*) a Gibbs sampler, analogous to that employed in the original sticky-HMM paper (Fox et al., 2008). For both the VB and Gibbs sampler, a truncated stick-breaking representation was used for the DP draws from the hierarchical Dirichlet process (HDP); see Fox et al. (2008) for a discussion of how the HDP is employed in this model.

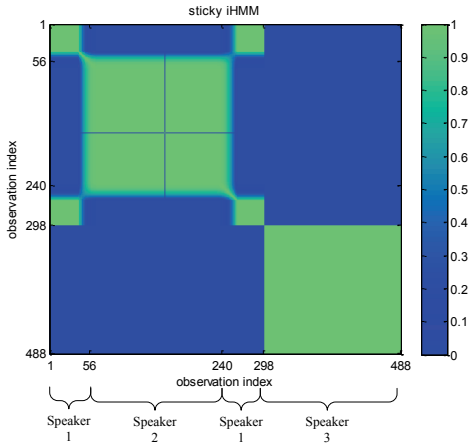


Figure 9: Sticky HMM results for the data in Figure 7(a), based on a Gibbs sampler. The figure denotes the fraction of times within the collection samples that a given portion of the waveform shares the same underlying state.

To segment the audio data, we labeled each observation to the index of the cluster with the largest probability value, and the results are shown in Figure 8 (here the sticky-HMM results were computed via VB analysis). To indicate the ground truth, different symbols and colors are used to represent different speakers.

From the results in Figure 8, the proposed LSBP yields the best segmentation performance, with results in close agreement with ground truth. We found the sticky-HMM results to be very sensitive to VB initialization, and the results in Figure 8 were the best we could achieve.

2. Software can be downloaded from `http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/`.

While the sticky HMM did not yield reliable VB-computed results, it performed well when a Gibbs sampler was employed (as in the work Fox et al., 2008). In Figure 9 are shown the fraction of times within the collection Gibbs samples that a given portion of the signal share the same underlying state; note that the results are in very close agreement with "truth". We cannot plot the Gibbs results in the same form as the VB results in Figure 8 due to label switching within the Gibbs sampler. The Gibbs-sampler results were computed using 5000 burn iterations and 5000 collection iterations.

These results demonstrate that the proposed LSBP, based on a fast VB solution, yields results commensurate with a state-of-the-art method (the sticky HMM based on a Gibbs sampler). On the same PC, the VB LSBP results required approximately 45 seconds of CPU time, while the Gibbs sticky-HMM results required 3 hours; in both cases the code was written in non-optimized Matlab, and these numbers should be viewed as providing a *relative* view of computational expense. The accuracy and speed of the VB LSBP is of interest for large-scale problems, like those considered in the next section. Further, the LSBP is a general-purpose algorithm, applicable to time- and spatially-dependent data (images), while the sticky HMM is explicitly designed for time-dependent data.

In the LSBP, DP, and KSBP analyses, we do not set the number of clusters *a priori* and the models infer the number of clusters automatically from the data. Therefore, we fixed the truncation level to $K = 10$ for all models, and the clustering results are shown in Figure 6, with different colors representing the cluster index (mixture component to which a data sample is assigned).

In Figure 2 we illustrated a draw from the LSBP prior, in the absence of any data. The parameters of that example (number of samples, the definition of N_c, and the library Ψ^*) were selected as to correspond to this audio example. To generate the draws in Figure 2, a spike-and-slab prior was employed, since the Student-t prior would prefer (in the absence of data) to set all coefficients to zero (or near zero), with high probability. Further, for related reasons we treated the bias terms w_{k0} distinct from the other coefficients. We now consider a draw from the LSBP *posterior*, based on the audio data considered above. This gives further insight into the machinery of the LSBP. We also emphasize that, in this example based on real data, as in all examples shown in this section, we impose sparseness via the Student-t prior. Therefore, when looking at the posterior, we may see which coefficients w_{ki} have been "pulled" away from zero such that the model fits the observed data. A representative draw from the LSBP posterior is shown in Figure 10, using the same presentation format as applied to the draw from the prior in Figure 2. Note that only three sticks have appreciable probability for any time t, and the segments tend to be localized, with near-unity probability of using a corresponding model parameter within a given segment. While the spike-slab prior was needed to manifest desirable draws from the prior alone, the presence of data simplifies the form of the LSBP prior, based only on a relatively standard use of the hierarchical Student-t construction.

5.3 Image Segmentation with LSBP

The images considered first are from Microsoft Research Cambridge[3] and each image has 320×213 pixels. To apply the hierarchical model to image segmentation, we first over-segment each image into $1,000$ "superpixels", which are local, coherent and preserve most of the structure necessary for segmentation at the scale of interest (Ren and Malik, 2003). The software used for this

3. Images can be downloaded from `http://research.microsoft.com/en-us/projects/objectclassrecognition/`.

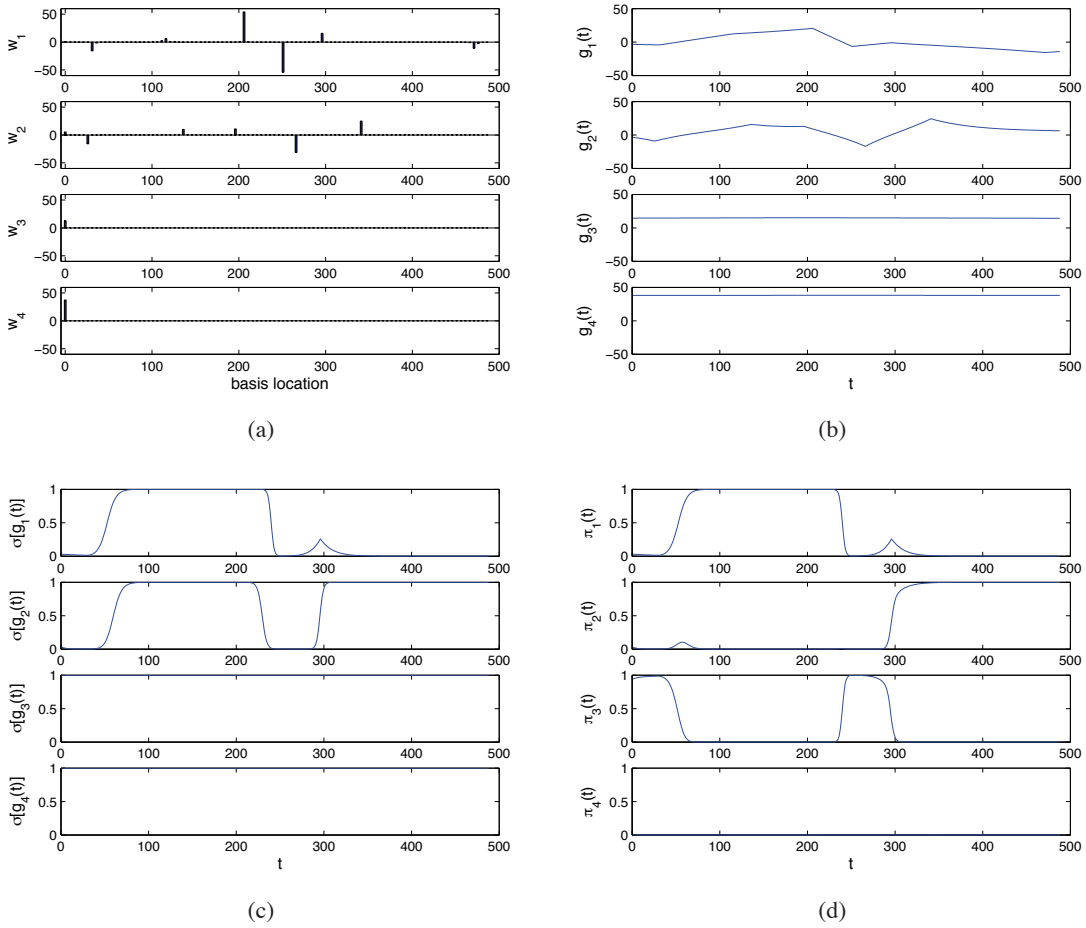


Figure 10: Example draw from the LSBP posterior, for the audio data under test. (a) $\boldsymbol{w}_k$, (b) $g_k(t)$, (c) $\sigma_k(t)$, (d) $\pi_k(t)$

is described in Mori (2005), and can be downloaded at `http://fas.sfu.ca/~mori/research/superpixels/`. Each superpixel is represented by both color and texture descriptors, based on the local RGB, hue feature vectors (Weijer and Schmid, 2006), and also the values of Maximum Response (MR) filter banks (Varma and Zisserman, 2002). We discretize these features using a codebook of size 32, and then calculate the distributions (Ahonen and Pietikäinen, 2009) for each feature within each superpixel as visual words (Cao and Li, 2007; Wang and Grimson, 2007).

Since each superpixel is represented by three visual words, the mixture components $\boldsymbol{\theta}_k^*$ are three multinomial distributions as $\{\text{Mult}(\mathbf{p}^{1*}_k) \otimes \text{Mult}(\mathbf{p}^{2*}_k) \otimes \text{Mult}(\mathbf{p}^{3*}_k)\}$ for $k = 1, \ldots, K$. The variational distribution $q(\boldsymbol{\theta}_k^*)$ is $\text{Dir}(\mathbf{p}^{1*}_k|\tilde{\boldsymbol{\beta}}^1_k) \otimes \text{Dir}(\mathbf{p}^{2*}_k|\tilde{\boldsymbol{\beta}}^2_k) \otimes \text{Dir}(\mathbf{p}^{3*}_k|\tilde{\boldsymbol{\beta}}^3_k)$, and within VB inference we optimize the parameters $\tilde{\boldsymbol{\beta}}^1_k$, $\tilde{\boldsymbol{\beta}}^2_k$, and $\tilde{\boldsymbol{\beta}}^3_k$.

To perform segmentation at the patch level (each superpixel corresponds to one patch), the center of each superpixel is recorded as the location coordinate $\mathbf{s}_n$. The discrete kernel-width set $\boldsymbol{\Psi}^*$ is composed of $30, 35, \ldots, 160$, which are scaled empirically based on the image and object average size. Typically we may choose the $\boldsymbol{\Psi}^*$ as a subset between the minimum and maximum Euclidean

distance associated with any two data points' spatial locations within this image. To save computational resources, we chose as basis locations $\{\hat{\mathbf{s}}_i\}_{i=1}^{N_c}$ the spatial centers of every tenth superpixel in a given image, after sequentially indexing the superpixels (we found that if we do not perform this subsampling, very similar segmentation results are achieved, but at greater computational expense).

Three representative example images are shown in Figures 11(a), (b) and (c); the superpixels are generated by over-segmentation (Mori, 2005) on each image, with associated over-segmentation results shown in Figures 11(d), (e) and (f). The segmentation task now reduces to grouping/clustering the superpixels based on the associated image feature vector and associated spatial information. To examine the effect of the truncation level K, we considered K from 2 to 10 and quantified the VB approximation to the model evidence (marginal likelihood). The segmentation performance for each of these images is shown in Figure 11(g), (h) and (i), using respectively $K = 4$, 3 and 6, based on the model evidence (discussed further below). These (typical) results are characterized by homogeneous segments with sharp boundaries. In Figure 11(j), (k) and (l), the segmentation results are shown with K fixed at $K = 10$. In this case the LSBP has ten sticks; however, based on the segmentation there are a subset of sticks (5, 8 and 7, respectively) inferred to have appreciable amplitude.

Based upon these representative example results, which are consistent with a large number of tests on related images, we make the following observations. Considering first the "chimney" results in Figure 11(a), (g) and (j), for example, we note that there are portions of the brick that have textural differences. However, the prior tends to favor contiguous segments, and one solid texture is manifested for the bricks. We also note the sharp boundaries manifested in the segments, despite the fact that the logistic-regression construction is only using simple Gaussian kernels (not particularly optimized for near-linear boundaries). For the relatively simple "chimney" image, the segmentation results are very similar with different initializations of K (Figure 11(g)) and simply truncating the sticks at a "large" value (Figure 11(j) with $K = 10$).

The "cow" example is more complex, pointing out further characteristics of LSBP. We again observe homogeneous contiguous segments with sharp boundaries. In this case a smaller K yields (as expected) a simpler segmentation (Figure 11(h)). All of the relatively dark cows are segmented together. By contrast, with the initialization of $K = 10$, the results in Figure 11(k) capture more details in the cows. However, we also note that in Figure 11(k) the clouds are properly assigned to a distinctive type of segment, while in Figure 11(h) the clouds are just included in the sky cluster/segment. Similar observations are also obtained from the "flower" example for Figure 11(c), with more flower texture details kept with a large truncation level setting in Figure 11(l) than the result with a smaller K shown in Figure 11(i).

Because of the sampling of the kernel width, the lower bound of the log model evidence did not increase monotonically in general. For the "chimney" example considered in Figure 11(a), the log model evidence was found to sequentially increase approximately within the first 20 iterations and then converge to the local optimal solution with small fluctuations, as shown in Figure 12(a) with a model of $K = 4$. To test the model performance with different initializations of K, we calculate the mean and standard deviation of the lower bound after 25 iterations when K equals from 2 to 10, as plotted in Figure 12(b); from this figure one clearly observes that the data favor the model with $K = 4$, for at this point the VB lower bound (approximation to the evidence) has its largest value. Hence, one may stop examining increasing K once it is evident that the model evidence is falling with increasing K (as compared with simply setting K to a large value).

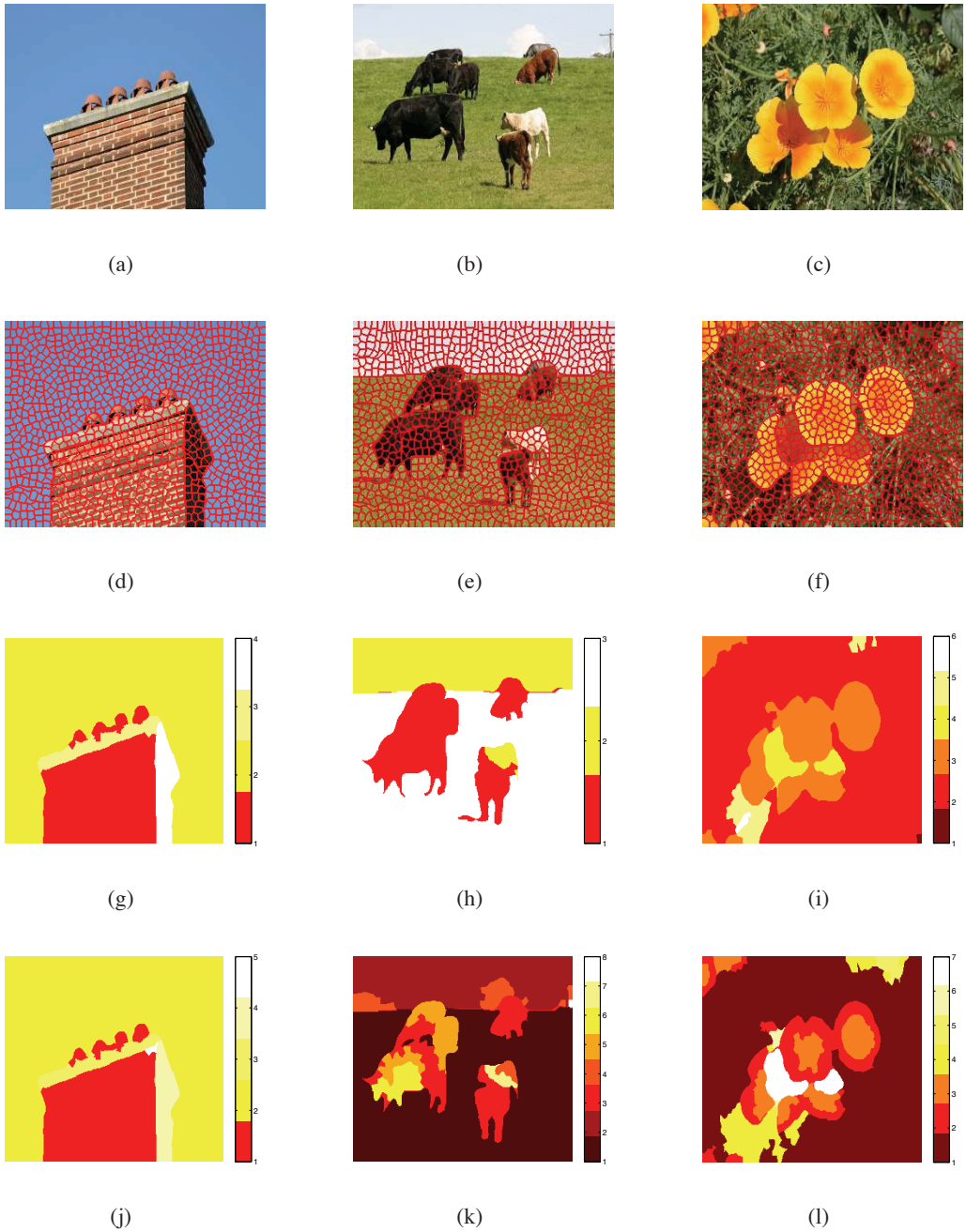
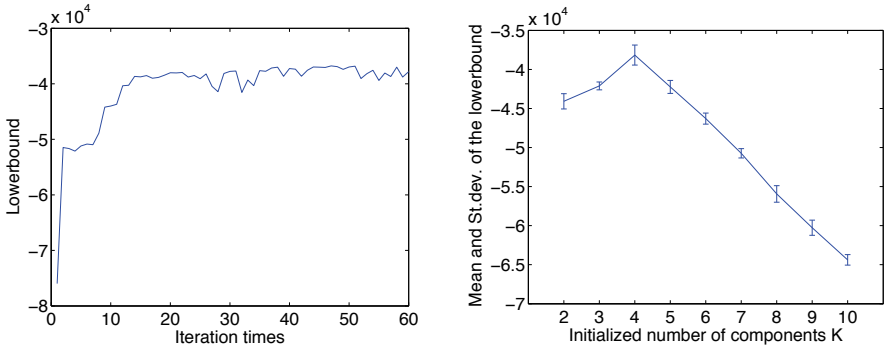


Figure 11: LSBP Segmentation for three image examples. (a)∼(c) image examples of "chimney","cows" and "flowers"; (d)∼(f) image examples represented with "superpixels"; (g)∼(i) segmentation results with largest values of model evidence ($K = 4$ for "chimney", $K = 3$ for "cows" and $K = 6$ for "flowers"); (j)∼(l) segmentation results with a initialization of $K = 10$ for the image examples.



(a) (b)

Figure 12: LSBP Segmentation for three image examples. (a)VB iteration lowerbound for image "chimney" with $K = 4$; (b) Approximating the model evidence as a function of K for image "chimney".

To further evaluate the performance of LSBP for image segmentation, we also consider several other state-of-art methods, including two other non-parametric statistical models: the Dirichlet process (DP) (Sethuraman, 1994) and the kernel stick-breaking process (KSBP) (An et al., 2008). We also consider two graph-based spectral decomposition methods: normalized cuts (Ncuts) (Shi and Malik, 2000) and multi-scale Ncut with long-range graph connections (Cour et al., 2005). Further, we consider the Student-t distribution mixture model (Stu.-t MM) (Sfikas et al., 2007), and also spatially varying mixture segmentation with edge preservation (St.-svgm) (Sfikas et al., 2008). We consider the same data source as in the previous examples, but for the next set of results segmentation "ground truth" was provided with the data. The data are divided into eight categories: trees, houses, cows, faces, sheep, flowers, lake and street; each category has thirty images. All models were initialized with a segment number of $K = 10$.

Figure 13 shows typical segmentation results for the different algorithms. Given a segment count number, both the normalized cuts and the multi-scale Ncut produced very smooth segmentations, while certain textured regions might be split into several pieces. The Student-t distribution mixture model (Stu.-t MM) yields a relatively robust segmentation, but it is sensitive to the texture appearance. Compared with Stu.-t MM, the spatially varying mixtures (St.-svgm) favors a more contiguous segmentation for the texture region, preserving edges; this may make a good tradeoff between keeping coherence and capturing details, but the segmentation performance is degraded by redundant boundaries, such as those within the goose body. Compared with these state-of-art algorithms, the LSBP results appear to be very competitive. Among the Bayesian methods (DP, KSBP and LSBP), LSBP tends to yield better segmentation, characterized by homogeneous segmentation regions and sharp segment boundaries.

To quantify segmentation results, we also calculated the Rand Index (RI) (Unnikrishnan et al., 2007) and the Variation of Information (VoI) (Meilă, 2003), using segmentation "truth" provided with the data. RI measures consistency between two segmentation labels via an overlapping fraction, and VoI roughly calculates the amount of randomness that exists in one segmentation that is not explained by the other. Accordingly, for the RI measure, larger values represent better performance, and for VoI smaller values are preferred. We calculated the average RI and VoI values of the

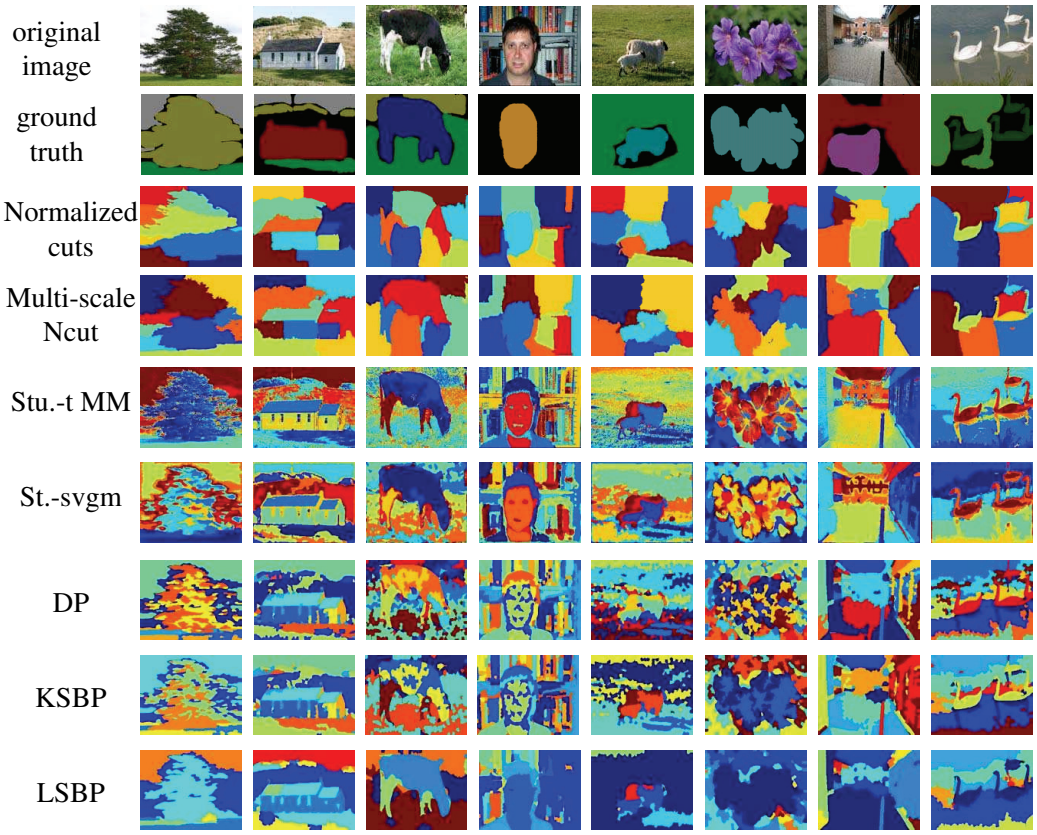


Figure 13: Segmentation examples of different methods with an initialization of $K = 10$. From top to down, each row shows: the original image, the image ground truth, normalized cuts, multiscale Ncut, Student-t distributions mixture model (Stu.-t MM), spatially varying mixtures (St.-svgm), DP mixture, KSBP mixture, and the LSBP mixture model.

thirty images for each category; the statistics for the two measures are depicted in Tables 1 and 2, considering all 240 images and various K.

Compared with other state-of-the-art methods, the LSBP yields relatively larger mean and median values for average RI, and relatively small average VoI, for most K. For $K = 2$ and 4 the spatially varying mixtures (St.-svgm) shows the largest RI values, while it does not yield similar effectiveness as K increases. In contrast, the LSBP yields a relatively stable RI and VoI from $K = 4$ to 10. This property is more easily observed in Figure 14, which shows the averaged RI and VoI evaluated as a function of K, for categories "houses" and "cows". The Stu.-t MM, St.-svgm, DP and KSBP have similar performances for most K; LSBP generates a competitive result with a smaller K, and also yields robust performance with a large K.

We also considered the Berkeley 300 data set.[4] These images have size 481×321 pixels, and we also over-segmented each image into 1000 superpixels. Both the RI and VoI measures are calculated on average, with the multiple labels (human labeled) provided with the data. Each individual image

4. Data set can be downloaded from `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/`.

K		2	4	6	8	10
Ncuts	mean	0.5552	0.6169	0.6269	0.6180	0.6093
	median	0.5259	0.6098	0.6376	0.6286	0.6235
	st. dev.	0.0953	0.1145	0.1317	0.1402	0.1461
Multi-scale Ncuts	mean	0.6102	0.6491	0.6387	0.6306	0.6228
	median	0.5903	0.6548	0.6515	0.6465	0.6396
	st. dev.	0.0979	0.1361	0.1462	0.1523	0.1584
Stu.-t MM	mean	0.6522	0.6663	0.6409	0.6244	0.6110
	median	0.6341	0.6858	0.6631	0.6429	0.6360
	st. dev.	0.1253	0.1248	0.1384	0.1455	0.1509
St.-svgm	mean	**0.6881**	**0.6861**	0.6596	0.6393	0.6280
	median	**0.6781**	**0.7026**	0.6825	0.6575	0.6516
	st. dev.	0.1249	0.1262	0.1427	0.1532	0.1599
DP	mean	0.6335	0.6527	0.6389	0.6270	0.6187
	median	0.6067	0.6669	0.6431	0.6321	0.6232
	st. dev.	0.1272	0.1283	0.1384	0.1464	0.1507
KSBP	mean	0.6306	0.6530	0.6396	0.6290	0.6229
	median	0.5963	0.6693	0.6448	0.6371	0.6272
	st. dev.	0.1237	0.1303	0.1397	0.1464	0.1523
LSBP	mean	0.6516	0.6791	**0.6804**	**0.6704**	**0.6777**
	median	0.6384	0.6921	**0.6900**	**0.6835**	**0.6885**
	st. dev.	0.1310	0.1202	0.1263	0.1294	0.1319

Table 1: Statistics on the averaged Rand Index (RI) over 240 images as a function of K (Microsoft Research Cambridge images).

typically has roughly ten segments within the ground truth. We calculated the evaluation measures for $K = 5, 10$ and 15. Table 3 presents results, demonstrating that all methods produced competitive results for both the RI and VoI measures. By a visual evaluation of the segmentation results (see Figure 15), multi-scale Ncut is not as good as the other methods when the segments are of irregular shape and unequal size.

The purpose of this section was to demonstrate that LSBP yields competitive segmentation performance, compared with many state-of-the-art algorithms. It should be emphasized that there is no perfect way of quantifying segmentation performance, especially since the underlying "truth" is itself subjective. An important advantage of the Bayesian methods (DP, KSBP and LSBP) is that they may be readily extended to joint segmentation of multiple images, considered in the next section.

5.4 Joint Image Segmentation with H-LSBP

In this section we consider H-LSBP for joint segmentation of multiple images. Experiments are performed on the Microsoft data, with another two unlabeled categories: "cloud" and "office". Each category is composed of 30 images, and therefore there are 300 images in total, analyzed simultaneously. The same feature and image processing techniques are employed as above.

K		2	4	6	8	10
Ncuts	mean	1.7911	2.2034	2.4344	2.6885	2.8828
	median	1.8201	2.1990	2.4392	2.7134	2.8956
	st. dev.	0.4402	0.4213	0.4003	0.3673	0.3615
Multi-scale Ncuts	mean	1.7017	2.0538	2.3535	2.5548	2.7397
	median	1.7322	2.0238	2.3746	2.5912	2.7471
	st. dev.	0.4253	0.4276	0.4030	0.4056	0.4215
Stu.-t MM	mean	1.4903	2.0078	2.4258	2.7421	3.0085
	median	1.5312	2.0283	2.4653	2.7495	3.0341
	st. dev.	0.5161	0.4544	0.4120	0.3941	0.3798
St.-svgm	mean	1.4031	1.8957	2.2667	2.5764	2.7999
	median	1.4000	1.8957	2.2673	2.5919	2.8123
	st. dev.	0.5094	0.4176	0.4113	0.3956	0.4001
DP	mean	1.4810	1.9522	2.2961	2.5808	2.7740
	median	1.5145	1.9522	2.3541	2.6321	2.8432
	st. dev.	0.4952	0.3923	0.4186	0.4164	0.4573
KSBP	mean	1.4806	1.9383	2.3063	2.5888	2.7873
	median	1.4980	1.9811	2.3403	2.6304	2.8338
	st. dev.	0.4811	0.3919	0.4150	0.4128	0.4457
LSBP	mean	**1.4484**	**1.8142**	**1.9811**	**2.1050**	**2.0861**
	median	**1.4631**	**1.8288**	**1.9825**	**2.1528**	**2.1178**
	st. dev.	0.4835	0.4478	0.4979	0.5101	0.5254

Table 2: Statistics on the Variation of Information (VoI) over 240 images as a function of K (Microsoft Research Cambridge images).

	Normalized cuts	Multiscale Ncut	Stu.-t MM	St.-svgm	DP mixture	KSBP mixture	LSBP mixture
RI	0.7220	0.7404	0.7093	0.7188	0.7228	0.7237	0.7241
VoI	2.7857	2.5541	3.7772	3.5682	2.8573	2.7027	2.6591

Table 3: Different segmentation methods compared on Berkeley 300 images data set.

The H-LSBP automatically generates a set of indicator variables $\mathbf{z}_{mn}$ for each superpixel. The probability that the nth superpixel within image m is associated with the kth hidden indicator variable t_{mk}, is represented as $p_k(\mathbf{s}_{mn}) = \sigma(g_k(\mathbf{s}_{mn}))\prod_{l<k}(1-\sigma(g_l(\mathbf{s}_{mn})))$. By integrating out the distribution for each hidden indicator variable t_{mk} drawn from the global set of atoms $\boldsymbol{\theta}^*_k$, we approximate the membership for each superpixel by assigning it to the cluster with largest probability. This "hard" segmentation decision is employed to provide labels for each data point (the Bayesian analysis yields a "soft" segmentation in terms of a full posterior distribution), as employed above when considering one image at a time.

Our goal is to segment all the images simultaneously, sharing model parameters (atoms) across all images. The results of this analysis are used to infer the inter-relationship between different images, of interest for image sorting and search. We set truncation levels $L = 40$ and $K = 10$ (similar results were found for larger truncations, and these parameters have not been optimized). As demonstrated below, the model automatically infers the total number of principal atoms shared

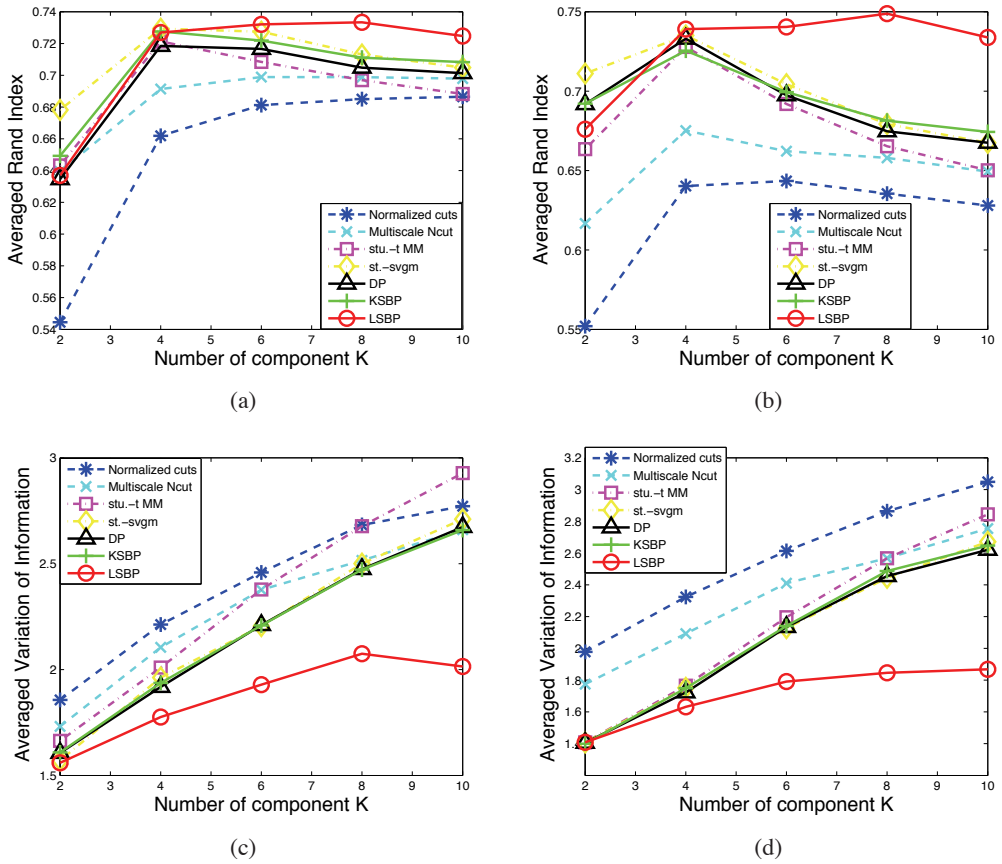


Figure 14: Average Rand Index (RI) and Variation of Information (VoI) as functions of K with image categories. (a) RI for "houses" , (b) RI for "cows", (c) VoI for "houses", (d) VoI for "cows".

across all images, and the number of atoms that dominate the segmentation of each individual image. The learning of these principal atoms, across the multiple images, is an important aspect of the model, so that the associated mixture weights with these atoms for each image can be regarded as a measurable quantity of inter-relationship between images (Blei et al., 2003; An et al., 2008). Specifically, similar images should have similar distributions over the model atoms. With the same inter-relationship measure generated from the HDP (Teh et al., 2005), H-KSBP (An et al., 2008) and the proposed H-LSBP, we may compare model utility as an image sorting or organizing engine.

To depict how the atoms are shared across multiple images with H-LSBP, we display an atom-usage count matrix in Figure 16, in which the size of each square size is proportional to the relative counts of that atom in a given image. Similar atom usage was revealed for HDP and H-KSBP (omitted for brevity), but the H-LSBP generally was more parsimonious in its use of atoms. This is attributed to the fact that the spatial continuity constraint within LSBP encourages a parsimonious representation (a relatively small number of contiguous clusters).

Each inferred image atom is in principle associated with one class of features within the images. To get a feel for how the model operates, we examine the types of image segments associated with representative atoms. Specifically, in Figure 17 we consider how eight representative atoms

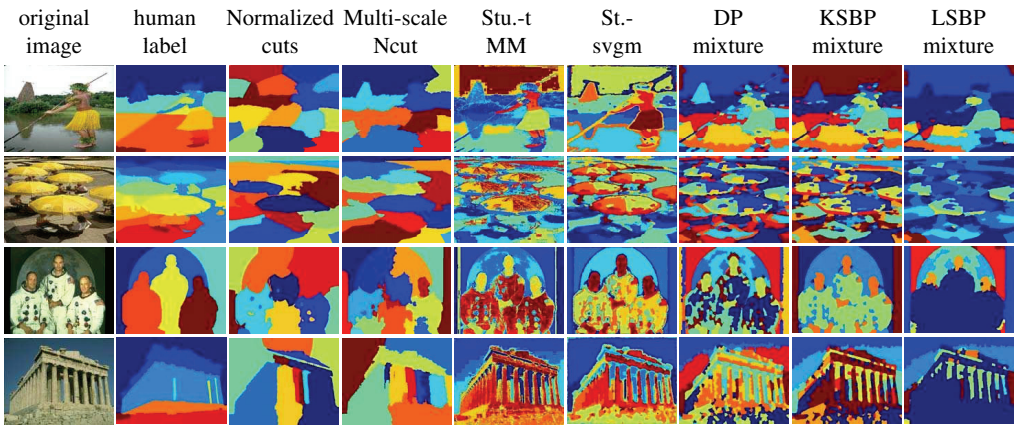


Figure 15: Segmentation examples of different methods with $K = 10$, for Berkeley image data set. From left to right, each column shows: the original image, the image ground truth, normalized cuts, multiscale Ncut, the Student-t distribution mixture model (Stu.-t MM), spatially varying mixtures (St.-svgm), DP mixture, KSBP mixture, and the LSBP mixture model.

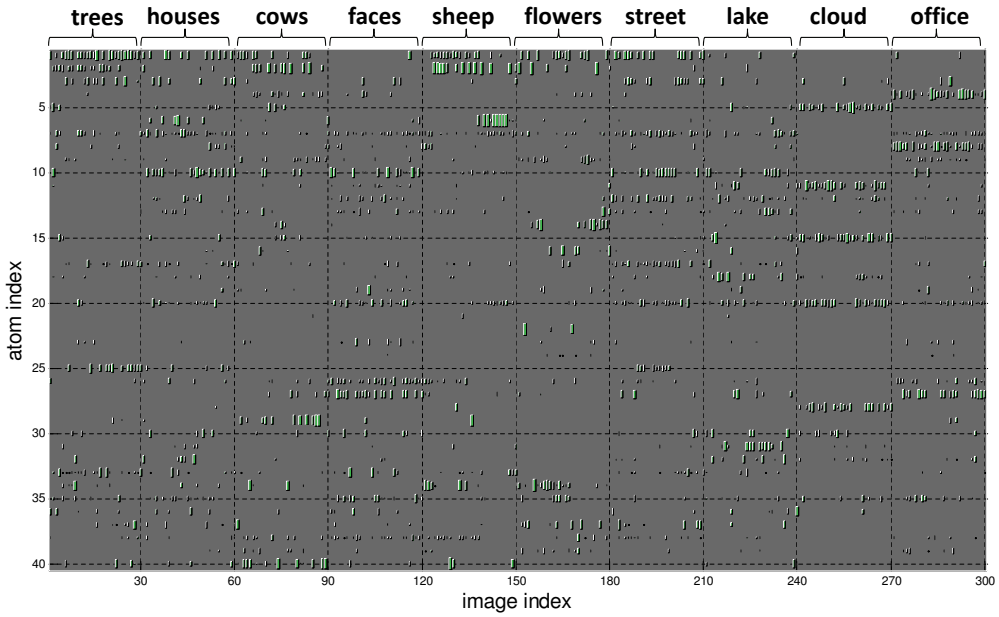


Figure 16: Atom usage-count matrix for H-LSBP.

are distributed within example images. In this figure we show the original image, and also the same image with all portions *not* associated with a given atom blacked out. From Figure 17 we observe that atom 1 is principally associated with trees, atom 2 is associated with grass, atom 4 principally models offices, and atom 10 is mainly attributed to the surface of buildings. Figure 18 shows atom examples inferred from the H-KSBP and HDP, and the representative "cloud", "grass",

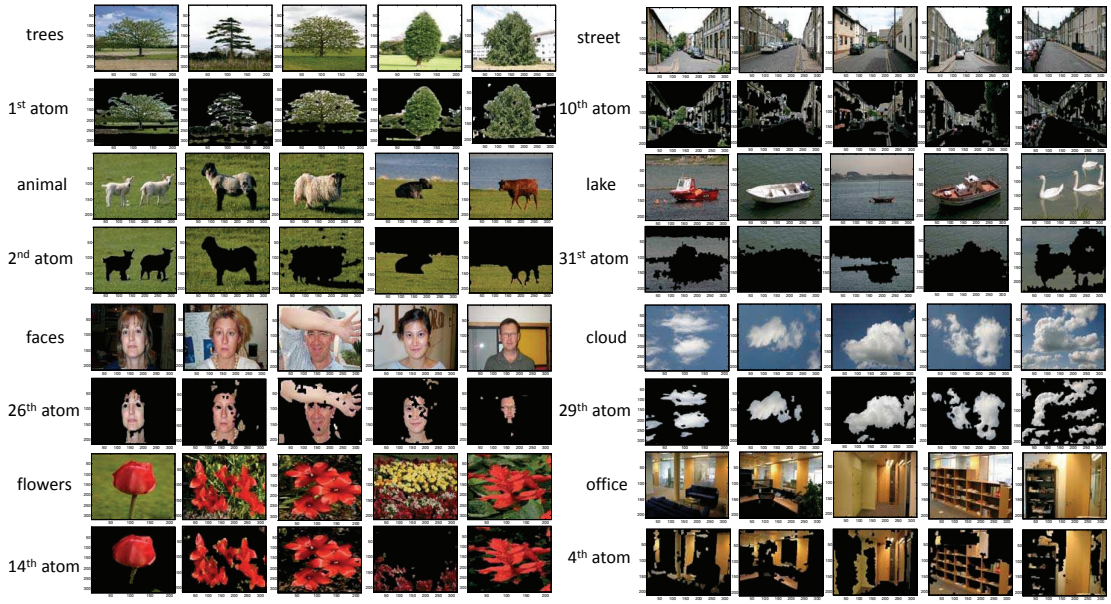


Figure 17: Demonstration of different atoms inferred by the H-LSBP model. The original images and associated connection to model-parameter atoms are shown on consecutive rows. All regions *not* associated with a respective atom are blacked out.

"tree" and "street" atoms do not do as well in maintaining spatial contiguity. This property is especially important to locate certain objects or scenes. For example, for an image annotation task, it is usually expensive to acquire training data set by manually annotating image by image. Therefore, the H-LSBP might be used as an automatic annotation tool to save redundant manual work for the preprocessing the images with no words given.

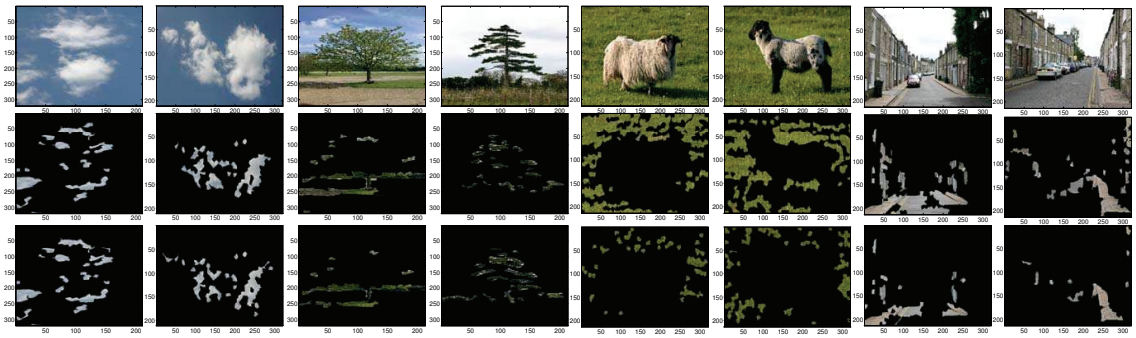


Figure 18: Examples of different atoms inferred by the H-KSBP and HDP model: The first row is the original images; the second row is the atoms inferred by H-KSBP; the third row is the atoms inferred by HDP.

Based on the atoms inferred from Figure 16, we can jointly segment the 300 images with H-LSBP. Each atom represents a label, and the superpixels that shared the same atom are grouped together. Some representative segmentation examples are shown in Figure 19, in which each column shows one segmentation example with its "ground truth" (the second row), and the color bar encodes the labels/indexes of the results in the third row (the labels are re-ordered to be different from the atom index).

Another interesting problem is to infer the inter-relationship between different images, and this may be achieved by quantifying the degree to which they share atoms (the sharing of the same set of atoms across all images plays an important role in inferring inter-image relationships). Since we know which atoms $\{\boldsymbol{\theta}_l^*\}_{l=1}^L$ the superpixels within each image are drawn from, we may calculate the Kullback-Leibler (KL) divergence based on the histogram over atoms between each pair of images (a small value is added to the probability of each atom, to avoid numerical problems when computing the KL divergence, when the actual usage of particular atoms may be zero). The KL divergence between different categories, computed by averaging across all of the sub-class images, are shown in Figure 20. To make the figure easier to read, the KL divergence D_{KL} is re-scaled as $\exp(-D_{KL})$. In Figure 20(a) results are shown based on the proposed H-LSBP, in (b) based upon an H-KSBP analysis, and in (c) based upon an HDP analysis. The H-LSBP, H-KSBP and HDP each yield good results, but Figure 20 indicates that the H-LSBP produces smaller cross-class similarity (additionally, the H-KSBP results are better than those of HDP).

To demonstrate the utility of the proposed method in the context of an image sorting/search engine, we show image sorting examples in Figure 21. The left-most column is the original image, and columns 2-6 are the ordered five most similar images in the database, ordered according to the value of the KL divergence between the original image and the remaining 299 images. The five most similar images are shown in Figure 21, with generally good sorting performance manifested.

5.5 Computational Complexity

All the experiments in this paper were performed in Matlab on a Pentium PC with 1.73 GHz CPU and 4G RAM. For the audio-waveform example, 80 VB iterations for LSBP required 40 seconds.

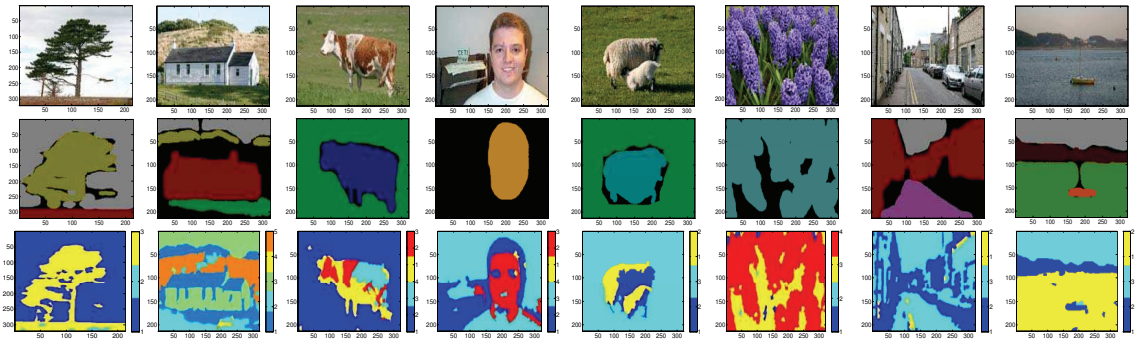


Figure 19: Representative set of segmentation results of H-LSBP. The top row gives example images, the second row defines "truth" as given by the data set, and the third row represents the respective H-LSBP results.

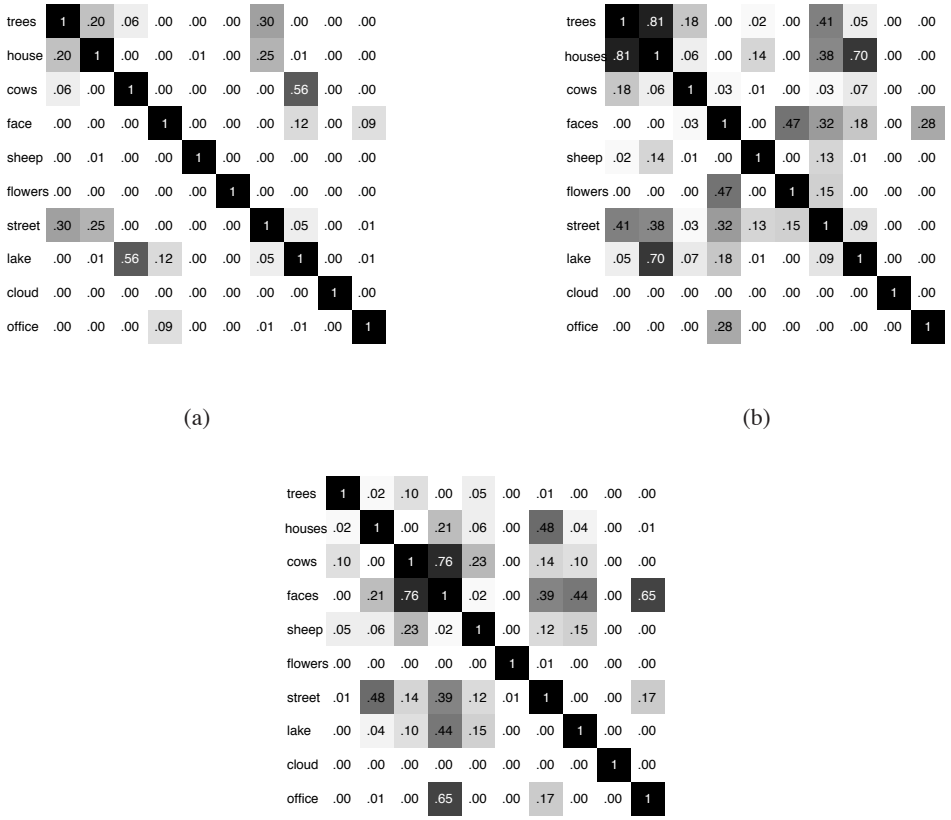


Figure 20: Similarity matrix associated with the ten image categories. (a) H-LSBP, (b) H-KSBP, (c) HDP

For the multi-task image segmentation, H-LSBP required nearly 7 hours of CPU to jointly segment 300 images, using 60 VB iterations (this CPU time may be cut in half if we only use 30 VB iterations, with minor degradation in performance). With both experiments, KSBP/H-KSBP typically required comparable CPU time, while DP/HDP required less than half the CPU time.

6. Conclusions

The logistic stick-breaking process (LSBP) is proposed for clustering spatially- or temporally-dependent data, imposing the belief that proximate data are more likely to be clustered together. The sticks in the LSBP are realized via multiple kernel-based logistic regression functions, with a shrinkage prior employed for favoring contiguous and spatially localized partitions. Competitive segmentation performance has been manifested in several examples. Relative to other related approaches, the proposed LSBP yields sharp segmentations, and is able to automatically infer an appropriate number of segments.

We also propose the *hierarchical* logistic stick-breaking process, H-LSBP, to segment multiple data sets simultaneously, with example results presented for images. The model parameters (atoms)

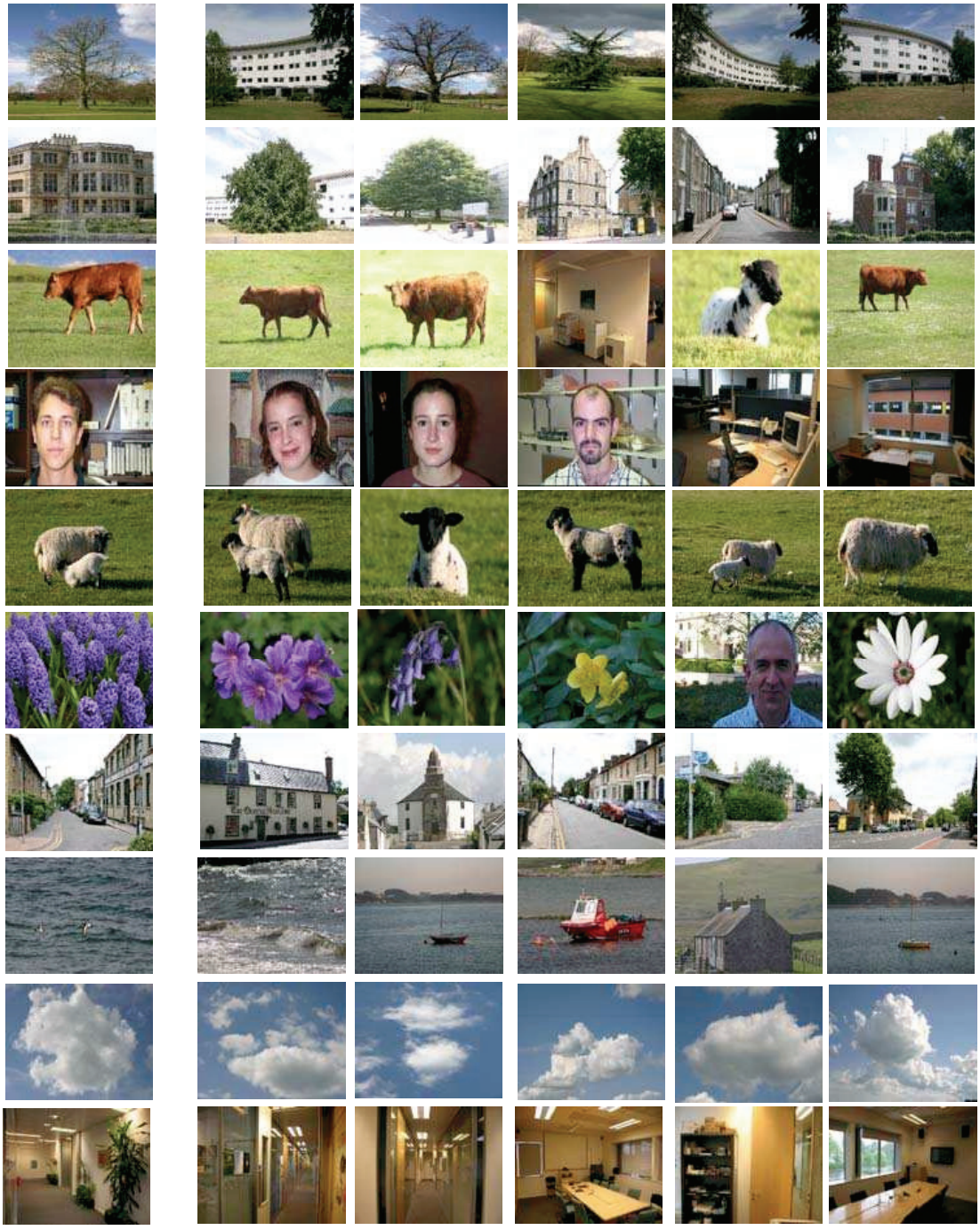


Figure 21: Sample image sorting result, as generated by H-LSBP. The first left column shows the images inquired, followed by the five most similar images from the second to sixth column.

are shared across all images, using a shared draw from a global DP prior. The total number of important atoms across all images, as well as the particular important atoms for a specific image, are inferred with an efficient variational Bayesian (VB) solution. Compared with the hierarchical Dirichlet process (HDP) and the hierarchical KSBP, the proposed method yields superior segmentation performance, based on studies with natural images. Further, we have investigated the ability of HDP, H-KSBP and H-LSBP to infer inter-relationship between different images, based on the underlying sharing of model atoms. The improved segmentation quality of the H-LSBP, relative to HDP and H-KSBP, also yields an improved ability to infer inter-image relationships.

Concerning future research, the results in Figure 17 indicate that the inferred atoms have connections to physical entities in images. This suggests that the model may be extended to the joint modeling of images and text (Barnard et al., 2003), with the text associated with aspects of the image. In addition, in the H-LSBP modeling of multiple images, the employed DP prior assumes that the order of the images is exchangeable (although LSBP imposes that spatial location within a particular image is not exchangeable). There are many applications (e.g., video) for which the multiple images may have a prescribed time index, that should be exploited. The results on the time-dependent audio data demonstrate how LSBP may also be employed to exploit temporal information.

The LSBP software is posted at `www.ece.duke.edu\~lcarin\LSBP_code.rar`

Appendix A. VB Update Equations for H-LSBP

For the model introduced in Section 3, we assume

$$q(\mathbf{\Phi}) = q(\gamma)\prod_{l=1}^{L} q(\boldsymbol{\theta}_l)\prod_{l'=1}^{L-1} q(\tilde{\beta}_{l'})\prod_{m=1}^{M}\Big[\prod_{k'=1}^{K} q(t_{mk'})\prod_{k=1}^{K-1}\big[q(\mathbf{w}_{mk})q(\boldsymbol{\lambda}_{mk})\prod_{n=1}^{N_m} q(z_{mnk})\big]\Big],$$

where $q(\boldsymbol{\theta}_l)$ is the Dirichlet distribution, the same form as its prior $p(\boldsymbol{\theta}_l|\boldsymbol{\alpha}_0)$. Then $q(\boldsymbol{\theta}_l|\tilde{\boldsymbol{\alpha}}_l)$ is updated with a uniform prior specified for $\boldsymbol{\alpha}_0$ as follows:

$$\tilde{\alpha}_{li} = \alpha_{0i} + \sum_{m=1}^{M}\sum_{n=1}^{N_m}\sum_{k'=1}^{K} < \xi_{mn,k'} >_{q(\mathbf{z}_{mn})} < t_{mk',l} > y_{mni},$$

where $\alpha_{0i} = 1/I$ for $i = 1,\ldots,I$, and I is the feature dimension; $< \xi_{mn,k'} >_{q(\mathbf{z}_{mn})} = \prod_{k=1}^{k'-1}(1-q(z_{mn}=k))\cdot q(z_{mn}=k')$ represents the approximated posterior probability that data D_{mn} is associated with the hidden "atom" $t_{mk'}$. For $k'=K$, $\xi_{mn,k'} = \prod_{k=1}^{k'-1}(1-q(z_{mn}=k))$. Finally, $< t_{mk',l} >= q(t_{mk'}=l)$ represents the approximated posterior probability that $t_{mk'}$ takes the atom $\boldsymbol{\theta}_l$.

For updating $q(\tilde{\boldsymbol{\beta}})$ and $q(\gamma)$ given the prior $p(\gamma) = Ga(\gamma|e_0,f_0)$, assume $q(\tilde{\beta}_l) = Be(\tilde{\beta}_l|\pi_{l1},\pi_{l2})$ with $l = 1,\ldots,L$, and $q(\gamma) = Ga(\gamma|\tilde{e},\tilde{f})$. Then the update equations are as follows:

$$\begin{aligned}
&\pi_{l1} = 1 + \textstyle\sum_{m=1}^{M}\sum_{k'=1}^{K} < t_{mk',l} >,\\
&\pi_{l2} = \tilde{e}/\tilde{f} + \textstyle\sum_{m=1}^{M}\sum_{k'=1}^{K}\sum_{l'=l+1}^{L} < t_{mk',l'} >,\\
&\tilde{e} = e_0 + L - 1,\\
&\tilde{f} = f_0 - \textstyle\sum_{l=1}^{L-1}\big[\psi(\pi_{l2}) - \psi(\pi_{l1}+\pi_{l2})\big],
\end{aligned}$$

in which $\psi(\cdot)$ is the Digamma function.

Given the approximate distribution of the other variables,

$$q(t_{mk'}=l) \propto \exp\Big[< \log p(t_{mk'}|\boldsymbol{\beta}) >_{q(\boldsymbol{\beta})} + < \log p(\mathbf{y}_m|t_{mk'},\mathbf{z}_m,\boldsymbol{\theta}_l) >_{q(\mathbf{z}_m),q(\boldsymbol{\theta}_l)} \Big],$$

where $< \cdot >_{q(\cdot)}$ represents the expectation of the associated variable's distribution. One may readily derive that

$$q(t_{mk'}=l) \quad \propto \quad \exp\Big[\sum_{l'=1}^{l-1}\big[\psi(\pi_{l'2})-\psi(\pi_{l'1}+\pi_{l'2})\big]+\big[\psi(\pi_{l1})-\psi(\pi_{l1}+\pi_{l2})\big]+ \sum_{n=1}^{N_m} < \xi_{mn,k'} >_{q(\mathbf{z}_{mn})} < \log p(y|\boldsymbol{\theta}_l) >_{q(\boldsymbol{\theta}_l)} \Big],$$

where $< \log p(y|\boldsymbol{\theta}_l) >_{q(\boldsymbol{\theta}_l)}$ is the data likelihood, with expectation performed with respect to the distribution of atoms $\boldsymbol{\theta}_l$ (which may be derived readily). Then $q(t_{mk'}) = \text{Mult}(u_{mk'1},\ldots,u_{mk'L})$, in which $u_{mk'l} = \frac{q(t_{mk'}=l)}{\sum_{l'=1}^{L} q(t_{mk'}=l')}$.

Similarly, assume $q(\mathbf{W}_{mk}) = N(\tilde{\mathbf{m}}_{mk}, \tilde{\boldsymbol{\Gamma}}_{mk})$ and $q(z_{mnk}=1) = \rho_{mn,k} = \sigma(h_{mnk})$ for $k = 1,\ldots,K-1$, then

$$h_{mnk} = \sum_{k'=k}^{K} (-1)^{\nu_{kk'}} < \xi_{mn,k'}^{-k} >_{q(\mathbf{z}_{mn}^{-k})} \sum_{l'=1}^{L} q(t_{mk'}=l') \big[< \log p(y|\boldsymbol{\theta}_l) >_{q(\boldsymbol{\theta}_l)} \big] + \tilde{\mathbf{m}}_{mk}^T \mathbf{x}_{mn}^k,$$

where $< \xi_{mn,k'}^{-k} >_{\mathbf{z}_{mn}^{-k}} = \prod_{j=1,j\neq k}^{K-1} \big[\rho_{mn,j}(-1)^{\nu_{jk'}} + \nu_{jk'}\big]$ is the expectation associated the gating variables $\{\mathbf{z}_{mn1},\ldots,\mathbf{z}_{mn(k-1)},\mathbf{z}_{mn(k+1)},\ldots,\mathbf{z}_{mn(K-1)}\}$ except $\mathbf{z}_{mnk}$, with the following definition for $\nu_{kk'}$:

$$\nu_{kk'} = \begin{cases} 0 & \text{if } t_{mk'} \text{ is in the left subtree of } G_{mk} \text{ (see Fig. 5),} \\ 1 & \text{otherwise.} \end{cases}$$

Assuming $q(\lambda_{mki}) = Ga(\tilde{a}_{mki}, \tilde{b}_{mki})$, with $i = 0,1,\ldots,N_c$, the update equations for $q(\mathbf{W}_{mk})$ are as follows:

$$\tilde{\boldsymbol{\Gamma}}_{mk} = \big[2\textstyle\sum_{n=1}^{N_m} f(\eta_{mnk}) \mathbf{x}_{mn}^k \mathbf{x}_{mn}^{k\,T} + \text{diag}(\frac{\tilde{\mathbf{a}}_{mk}}{\tilde{\mathbf{b}}_{mk}})\big]^{-1},$$
$$\tilde{\mathbf{m}}_{mk} = \tilde{\boldsymbol{\Gamma}}_{mk} \textstyle\sum_{n=1}^{N_m} \big[(\rho_{mn,k} - 1/2)\mathbf{x}_{mn}^k\big],$$

where the variational parameter

$$\eta_{mnk} = \sqrt{\mathbf{x}_{mn}^{k\,T} (\tilde{m}_{mk}\tilde{m}_{mk}^T + \tilde{\boldsymbol{\Gamma}}_{mk}) \mathbf{x}_{mn}^k},$$

and $f(\eta_{mnk}) = \frac{\tanh(\eta_{mnk}/2)}{4\eta_{mnk}}$ (Bishop and Svensén, 2003; Bishop and Tipping, 2000). The parameters $\mathbf{x}_{mn}^k$ are defined as $\mathbf{x}_{mn}^k = \{1, \{K(\mathbf{s}_{mn}, \hat{\mathbf{s}}_{mi}; \psi_{mk}\}_{i=1}^{N_c}\}$.

Given $q(\mathbf{W}_{mk})$, the update equations for $q(\boldsymbol{\lambda}_{mk})$ are

$$\tilde{a}_{mki} = a_0 + 1/2,$$
$$\tilde{b}_{mki} = \tfrac{1}{2}(\tilde{\boldsymbol{\Gamma}}_{mk}(i,i) + \tilde{m}_{mki}^2) + b_0.$$

References

T. Ahonen and M. Pietikäinen. Image discription using joint distribution of filter bank responses. *Pattern Recognition Letters*, 30:368–376, 2009.

Q. An, C. Wang, I. Shterev, E. Wang, L. Carin, and D. B. Dunson. Hierarchical kernel Stick-breaking process for multi-task image analysis. In *Proceedings of the International Conference of Machine Learning*, 2008.

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

S. Banerjee, B. P. Carlin, and A. E. Gelfand. *Hierarchical Modeling and Analysis for Spatial Data*. Chapman and Hall/CRC, 1st edition, 2003.

K. Barnard, P. Duygulu, D. Forsyth, N. D. Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

M. J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, The Gatsby Computational Neuroscience Unit, University College London, 2003.

M. J. Beal, Z. Ghahramani, and C. E. Rasmussen. The infinite hidden Markov model. In *Proceedings of the Advances in Neural Information Processing Systems 14*, 2002.

M. Ben, M. Betser, F. Bimbot, and G. Gravier. Speaker diarization using bottom-up clustering based on a parameter-derived distance between adapted GMMs. In *Proceedings of the International Conference on Spoken Language Processing*, 2004.

C. M. Bishop and M. Svensén. Bayesian hierarchical mixture of experts. In *Proceedings of the Ninetheenth Conference of Uncertainty in Artificial Intelligence*, 2003.

C. M. Bishop and M. E. Tipping. Variational relevance vector machines. In *Proceedings of the sixth Conference of Uncertainty in Artificial Intelligence*, 2000.

D. Blackwell and J. B. MacQueen. Ferguson distributions via Polya urn schemes. *Ann. Statist.*, 1 (2):353–355, 1973.

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

L. Cao and F. Li. Spatially coherent latent topic model for concurrent object segmentation and classification. In *Proceedings of ICCV*, 2007.

R. Caruana. *Multitask learning*. Kluwer Academic Publishers, 1997.

Y. Chung and D. B. Dunson. Nonparametric Bayes conditional distribution modeling with variable selection. *Journal of the American Statistical Association*, 2009.

T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.

J. A. Duan, M. Guindani, and A. E. Gelfand. Generalized spatial Dirichlet process models. *Biometrika*, 94(4):809–825, 2007.

D. B. Dunson and J. Park. Kernel Stick-breaking processes. *Biometrika*, 95:307–323, 2007.

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

P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59:167–181, 2004.

E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky. The sticky HDP-HMM for systems with state persistence. In *Proceedings of the International Conference on Machine Learning*, 2008.

T. Ganchev, N. Fakotakis, and G. Kokkinakis. Comparative evaluation of various MFCC implementations on the speaker verification task. In *Proceedings of the International Conference on Speech and Computer*, 2005.

A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Springer, 1991.

W. R. Gilks, S. Richardson, and D. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman and Hall/CRC, 1st edition, 1998.

R. Gomes, M. Welling, and P. Perona. Memory bounded inference in topic models. In *Proceedings of the International Conference of Machine Learning*, 2008.

H. Ishwaran and L. F. James. Gibbs sampling methods for Stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, 2001.

H. Ishwaran and J.S. Rao. Spike and slab variable selection: Frequentist and Bayesian strategies. *Annals of Statistics*, 2005.

M. Meilă. Comparing clusterings by the variation of information. In *Proceedings of the Sixteenth Annual Conference ofn Computational Learning Theory*, 2003.

G. Mori. Guiding model search using segmentation. In *Proceedings of the International Conference on Computer Vision*, 2005.

X. Nguyen and A. E. Gelfand. The Dirichlet labeling process for functional data analysis. Technical Report T.R. 08-37, Dept. of Statistical Science, Duke University, 2008.

P. Orbanz and J. M. Buhmann. Nonparametric Bayesian image segmentation. *International Journal of Computer Vision*, 77:25–45, 2008.

C. Pantofaru, C. Schmid, and M. Hebert. Object recognition by integrating multiple image segmentations. *In ECCV*, 2008.

J. Park and D. B. Dunson. Bayesian generalized product partition model. *Statistica Sinica*, 2009.

S. Petrone, M. Guindani, and A. E. Gelfand. Hybrid Dirichlet mixture models for functional data. *Journal Royal Statistical Society, Ser. B*, 2009.

C. E. Rasmussen. The infinite Gaussian mixture model. In *Proceedings of the Advances in Neural Information Processing Systems 12*, 2000.

X. Ren and J. Malik. Learning a classification model for segmentation. In *Proceedings of the International Conference on Computer Vision*, 2003.

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 2:639–650, 1994.

G. Sfikas, C. Nikou, and N. P.Galatsanos. Robust image segmentation with mixtures of student's t-distributions. In *Proceedings of the IEEE International conference on Image Processing*, 2007.

G. Sfikas, C. Nikou, and N. P.Galatsanos. Edge preserving spatially varying mixtures for image segmentaiton. In *Proceedings of the International conference on Computer Vision and Pattern recognition*, 2008.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

E. B. Sudderth and M. I. Jordan. Shared segmentation of natural scenes using dependent Pitman-Yor processes. In *Proceedings of the Neural Information Processing Systems*, 2008.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1582, 2005.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

S. Tranter and D. Reynolds. An overview of automatic speaker diarisation systems. *IEEE Trans. on Audio, Speech, and Language Processing*, 14:1557–1565, 2006.

R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward objective evaluation of image segmentation algorithms. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(6):929–944, 2007.

M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. *In Proceedings of the 7th European Conference on Computer Vision (ECCV)*, 2002.

X. Wang and E. Grimson. Spatial latent Dirichlet allocation. In *Proceedings of the Neural Information Processing Systems*, 2007.

G. C. G. Wei and M. A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor mans data augmentation algorithms. *Journal of the American Statistical Association*, 85(411): 699–704, 1990.

J. V. D. Weijer and C. Schmid. Coloring local feature extraction. In *Proceedings of the 9th European Conference on Computer Vision*, 2006.

J. Winn and C. M. Bishop. Variational message passing. *J. Machine Learning Research*, 6:661–694, 2005.

R. Zabih and V. Kolmogorov. Spatially coherent clustering using Graph Cuts. In *Proceedings of CVPR'04*, 2004.

Journal of Machine Learning Research 12 (2011) 241-266 Submitted 8/10; Revised 12/10; Published 1/11

Online Learning in Case of Unbounded Losses Using Follow the Perturbed Leader Algorithm

Vladimir V. V'yugin VYUGIN@IITP.RU
Institute for Information Transmission Problems
Russian Academy of Sciences
Bol'shoi Karetnyi per. 19
Moscow GSP-4, 127994, Russia

Editor: Manfred Warmuth

Abstract

In this paper the sequential prediction problem with expert advice is considered for the case where losses of experts suffered at each step cannot be bounded in advance. We present some modification of Kalai and Vempala algorithm of following the perturbed leader where weights depend on past losses of the experts. New notions of a volume and a scaled fluctuation of a game are introduced. We present a probabilistic algorithm protected from unrestrictedly large one-step losses. This algorithm has the optimal performance in the case when the scaled fluctuations of one-step losses of experts of the pool tend to zero.

Keywords: prediction with expert advice, follow the perturbed leader, unbounded losses, adaptive learning rate, expected bounds, Hannan consistency, online sequential prediction

1. Introduction

Experts algorithms are used for online prediction or repeated decision making or repeated game playing. Starting with the Weighted Majority Algorithm (WM) of Littlestone and Warmuth (1994) and Aggregating Algorithm of Vovk (1990), the theory of Prediction with Expert Advice has rapidly developed in the recent times. Also, most authors have concentrated on predicting binary sequences and have used specific (usually convex) loss functions, like absolute loss, square and logarithmic loss. A survey can be found in the book of Lugosi and Cesa-Bianchi (2006). Arbitrary losses are less common, and, as a rule, they are supposed to be bounded in advance (see well known Hedge Algorithm of Freund and Schapire 1997, Normal Hedge of Chaudhuri et al. 2009 and other algorithms).

In this paper we consider a different general approach—"Follow the Perturbed Leader – FPL" algorithm, now called Hannan's algorithm, see Hannan (1957), Kalai and Vempala (2003) and Lugosi and Cesa-Bianchi (2006). Under this approach we only choose the decision that has fared the best in the past—the leader. In order to cope with adversary some randomization is implemented by adding a perturbation to the total loss prior to selecting the leader. The goal of the learner's algorithm is to perform almost as well as the best expert in hindsight in the long run. The resulting FPL algorithm has the same performance guarantees as WM-type algorithms for fixed learning rate and bounded one-step losses, save for a factor $\sqrt{2}$. A major advantage of the FPL algorithm is that its analysis remains easy for an adaptive learning rate, in contrast to WM-derivatives (see remark in Hutter and Poland 2004).

©2011 Vladimir V'yugin.

Prediction with Expert Advice considered in this paper proceeds as follows. We are asked to perform sequential actions at times $t = 1,2,\ldots,T$. At each time step t, experts $i = 1,\ldots N$ receive results of their actions in form of their losses s_t^i—arbitrary real numbers.

At the beginning of the step t *Learner*, observing cumulating losses $s_{1:t-1}^i = s_1^i + \ldots + s_{t-1}^i$ of all experts $i = 1,\ldots N$, makes a decision to follow one of these experts, say Expert i. At the end of step t *Learner* receives the same loss s_t^i as Expert i at step t and suffers *Learner's* cumulative loss $s_{1:t} = s_{1:t-1} + s_t^i$.

In the traditional framework, we suppose that one-step losses of all experts are bounded, for example, $0 \le s_t^i \le 1$ for all i and t.

Well known simple example of a game with two experts shows that Learner can perform much worse than each expert: let the current losses of two experts on steps $t = 0,1,\ldots,6$ be $s^1_{0,1,2,3,4,5,6} = (\frac{1}{2},0,1,0,1,0,1)$ and $s^2_{0.1,2,3,4,5,6} = (0,1,0,1,0,1,0)$. Evidently, "Follow the Leader" algorithm always chooses the wrong prediction.

When the experts one-step losses are bounded, this problem has been solved using randomization of the experts cumulative losses. The method of following the perturbed leader was discovered by Hannan (1957). Kalai and Vempala (2003) rediscovered this method and published a simple proof of the main result of Hannan. They called an algorithm of this type FPL (Following the Perturbed Leader).

The FPL algorithm outputs prediction of an expert i which minimizes

$$s_{1:t-1}^i - \frac{1}{\varepsilon}\xi^i,$$

where ξ^i, $i = 1,\ldots N$, $t = 1,2,\ldots$, is a sequence of i.i.d random variables distributed according to the exponential distribution with the density $p(x) = \exp\{-x\}$, and ε is *a learning rate*.

Kalai and Vempala (2003) show that the expected cumulative loss of the FPL algorithm has the upper bound

$$E(s_{1:t}) \le (1+\varepsilon) \min_{i=1,\ldots,N} s_{1:t}^i + \frac{\log N}{\varepsilon},$$

where ε is a positive real number such that $0 < \varepsilon < 1$ is a learning rate, N is the number of experts.

Hutter and Poland (2004, 2005) presented a further developments of the FPL algorithm for countable class of experts, arbitrary weights and adaptive learning rate. Also, FPL algorithm is usually considered for bounded one-step losses: $0 \le s_t^i \le 1$ for all i and t. Using a variable learning rate, an optimal upper bound was obtained in Hutter and Poland (2005):

$$E(s_{1:t}) \le \min_{i=1,\ldots,N} s_{1:t}^i + 2\sqrt{2T \ln N}.$$

Most papers on prediction with expert advice either consider bounded losses or assume the existence of a specific loss function (see Lugosi and Cesa-Bianchi 2006). We allow losses at any step to be unbounded. The notion of a specific loss function is not used.

The setting allowing unbounded one-step losses do not have wide coverage in literature; we can only refer reader to Allenberg et al. (2006), Cesa-Bianchi et al. (2007) and Poland and Hutter (2005).

Poland and Hutter (2005) have studied the games where one-step losses of all experts at each step t are bounded from above by an increasing sequence B_t given in advance. They presented a learning algorithm which is asymptotically consistent for $B_t = t^{1/16}$.

Allenberg et al. (2006) have considered polynomially bounded one-step losses for a modified version of the Littlestone and Warmuth (1994) algorithm under partial monitoring. In full information case, their algorithm has the expected regret $2\sqrt{N\ln N}(T+1)^{\frac{1}{2}(1+a+\beta)}$ in the case where one-step losses of all experts $i=1,2,\ldots N$ at each step t have the bound $(s_t^i)^2 \le t^a$, where $a>0$, and $\beta>0$ is a parameter of the algorithm. They have proved that this algorithm is Hannan consistent if

$$\max_{1\le i\le N}\frac{1}{T}\sum_{t=1}^{T}(s_t^i)^2 < cT^a$$

for all T, where $c>0$ and $0<a<1$.

Cesa-Bianchi et al. (2007) derived a new forecasting strategy for the Weighted Majority algorithm in unbounded setting with regret

$$\sqrt{Q_T\ln N}+M_T\ln N,$$

where $M_T=\max_{1\le i\le N}\max_{1\le t\le T}|s_t^i|$ is the largest absolute value of any loss s_t^i of an expert i at time step T, and $Q_T=\sum_{t=1}^{T}(s_t^{i_{\min}})^2$ is the sum of squared losses of the best at first T steps expert $i^{\min}$. These bounds were improved using cumulative variances of losses (under distributions used in the Weighted Majority algorithm). Cesa-Bianchi et al. (2007) do not study asymptotic consistency of their algorithm.

In this paper we present a sufficient condition for the FPL algorithm to be asymptotically consistent in case where losses are unbounded. In particular, this setting covers a case where loss grows "faster than polynomial, but slower than exponential". We present some modification of Kalai and Vempala (2003) algorithm of following the perturbed leader (FPL) for the case of unrestrictedly large one-step expert losses s_t^i not bounded in advance: $s_t^i\in(-\infty,+\infty)$. This algorithm uses adaptive weights depending on past cumulative losses of the experts.

A motivating example, where losses of the experts cannot be bounded in advance, is given in Section 5.

The full information case is considered in this paper. We analyze the asymptotic consistency of our algorithms using nonstandard scaling. We introduce new notions of *the volume of a game* $v_t=v_0+\sum_{j=1}^{t}\max_i|s_j^i|$ and *the scaled fluctuation* of the game $\mathrm{fluc}(t)=\Delta v_t/v_t$, where $\Delta v_t=v_t-v_{t-1}$ and v_0 is a nonnegative constant.

We show in Theorem 2 that the algorithm of following the perturbed leader with adaptive weights constructed in Section 3 is asymptotically consistent in the mean in the case where $v_t\to\infty$ and $\Delta v_t=o(v_t)$ as $t\to\infty$ with a computable bound. Specifically, if $\mathrm{fluc}(t)\le\gamma(t)$ for all t, where $\gamma(t)$ is a computable function such that $\gamma(t)=o(1)$ as $t\to\infty$, our algorithm has the expected regret

$$2\sqrt{(8+\varepsilon)(1+\ln N)}\sum_{t=1}^{T}(\gamma(t))^{1/2}\Delta v_t,$$

where $\varepsilon>0$ is a parameter of the algorithm.

In case where all losses are nonnegative: $s_t^i\in[0,+\infty)$, we obtain the regret

$$2\sqrt{(2+\varepsilon)(1+\ln N)}\sum_{t=1}^{T}(\gamma(t))^{1/2}\Delta v_t.$$

In particular, this algorithm is asymptotically consistent (in the mean) in a modified sense

$$\limsup_{T\to\infty}\frac{1}{v_T}E(s_{1:T}-\min_{i=1,\dots N}s^i_{1:T})\le 0, \tag{1}$$

where $s_{1:T}$ is the total loss of our algorithm on steps $1,2,\dots T$, and $E(s_{1:T})$ is its expectation.

Proposition 1 of Section 2 shows that if the condition $\Delta v_t = o(v_t)$ is violated the cumulative loss of any probabilistic prediction algorithm can be much more than the loss of the best expert of the pool.

In Section 3 we present some sufficient conditions under which our learning algorithm is Hannan consistent.[1]

In Section 4 we consider some special cases of our algorithm and applications for the case of standard time-scaling.

In particular, Corollary 8 of Theorem 2 says that our algorithm is asymptotically consistent (in the modified sense) in the case when one-step losses of all experts at each step t are bounded by t^a, where a is a positive real number. We prove this result under an extra assumption that the volume of the game grows slowly, $\liminf_{t\to\infty} v_t/t^{a+\delta} > 0$, where $\delta > 0$ is arbitrary. Corollary 8 shows that our algorithm is also Hannan consistent when $\delta > \frac{1}{2}$.

In Section 5 we consider an application of our algorithm for constructing an arbitrage strategy in some game of buying and selling shares of some stock on financial market. We analyze this game in the decision theoretic online learning (DTOL) framework (see Freund and Schapire 1997). We introduce *Learner* that computes weighted average of different strategies with unbounded gains and losses. To change from the follow leader framework to DTOL we derandomize our FPL algorithm.

This paper is an extended version of the ALT 2009 conference paper V'yugin (2009).

2. Games of Prediction with Expert Advice with Unbounded One-step Losses

We consider a game of prediction with expert advice with arbitrary unbounded one-step losses. Following Cesa-Bianchi et al. (2007) we call a game with such losses "signed game" and call these losses "signed losses".

At each step t of the game, all N experts receive one-step losses $s^i_t \in (-\infty,+\infty)$, $i=1,\dots N$, and the cumulative loss of the ith expert after step t is equal to

$$s^i_{1:t} = s^i_{1:t-1} + s^i_t.$$

A probabilistic learning algorithm of choosing an expert outputs at any step t the probabilities $P\{I_t = i\}$ of following the ith expert given the cumulative losses $s^i_{1:t-1}$ of the experts $i=1,\dots N$ in hindsight (see Figure 1).

The performance of this probabilistic algorithm is measured in its *expected regret*

$$E(s_{1:T}-\min_{i=1,\dots N}s^i_{1:T}),$$

where the random variable $s_{1:T}$ is the cumulative loss of the master algorithm, $s^i_{1:T}$, $i=1,\dots N$, are the cumulative losses of the experts algorithms and E is the mathematical expectation (with respect

1. This means that (1) holds with probability 1, where E is omitted.

Probabilistic algorithm of choosing an expert.
FOR $t = 1, \dots T$
Given past cumulative losses of the experts $s^i_{1:t-1}, i = 1, \dots N$, choose an expert i with probability $P\{I_t = i\}$.
Receive the one-step losses at step t of the expert s^i_t and suffer one-step loss $s_t = s^i_t$ of the master algorithm.
ENDFOR

Figure 1: Probabilistic algorithm of choosing an expert

to the probability distribution generated by probabilities $P\{I_t = i\}$, $i = 1, \dots N$, on the first T steps of the game).

In the case of bounded one-step expert losses, $s^i_t \in [0,1]$, and a convex loss function, the well-known learning algorithms have expected regret $O(\sqrt{T \log N})$ (see Lugosi and Cesa-Bianchi 2006).

A probabilistic algorithm is called *asymptotically consistent* in the mean if

$$\limsup_{T\to\infty} \frac{1}{T} E(s_{1:T} - \min_{i=1,\dots N} s^i_{1:T}) \le 0. \tag{2}$$

A probabilistic learning algorithm is called *Hannan consistent* if

$$\limsup_{T\to\infty} \frac{1}{T} \left(s_{1:T} - \min_{i=1,\dots N} s^i_{1:T} \right) \le 0 \tag{3}$$

almost surely, where $s_{1:T}$ is its random cumulative loss.

In this section we study the asymptotical consistency of probabilistic learning algorithms in the case of unbounded one-step losses.

Notice that when $0 \le s^i_t \le 1$ all expert algorithms have total loss $\le T$ on first T steps. This is not true for the unbounded case, and there are no reasons to divide the expected regret (2) by T. We change the standard time scaling (2) and (3) on a new scaling based on a new notion of volume of a game. We modify the definition (2) of the normalized expected regret as follows. Define *the volume* of a game at step t

$$v_t = v_0 + \sum_{j=1}^{t} \max_i |s^i_j|,$$

where v_0 is a nonnegative constant. Evidently, $v_{t-1} \le v_t$ for all t.

A probabilistic learning algorithm is called *asymptotically consistent* in the mean (in the modified sense) in a game with N experts if

$$\limsup_{T\to\infty} \frac{1}{v_T} E(s_{1:T} - \min_{i=1,\dots N} s^i_{1:T}) \le 0.$$

A probabilistic algorithm is called Hannan consistent (in the modified sense) if

$$\limsup_{T\to\infty} \frac{1}{v_T} \left(s_{1:T} - \min_{i=1,\dots N} s^i_{1:T} \right) \le 0 \tag{4}$$

almost surely.

Notice that the notions of asymptotic consistency in the mean and Hannan consistency may be non-equivalent for unbounded one-step losses.

A game is called *non-degenerate* if $v_t \to \infty$ as $t \to \infty$.

Denote $\Delta v_t = v_t - v_{t-1}$. The number

$$\mathrm{fluc}(t) = \frac{\Delta v_t}{v_t} = \frac{\max_i |s_t^i|}{v_t},$$

is called *scaled fluctuation* of the game at the step t.

By definition $0 \le \mathrm{fluc}(t) \le 1$ for all t (put $0/0 = 0$).

The following simple proposition shows that each probabilistic learning algorithm is not asymptotically optimal in some game such that $\mathrm{fluc}(t) \not\to 0$ as $t \to \infty$. For simplicity, we consider the case of two experts and nonnegative losses.

Proposition 1 *For any probabilistic algorithm of choosing an expert and for any* ε *such that* $0 < \varepsilon < 1$ *two experts exist such that* $v_t \to \infty$ *as* $t \to \infty$ *and*

$$\mathrm{fluc}(t) \ge 1 - \varepsilon,$$
$$\frac{1}{v_t} E(s_{1:t} - \min_{i=1,2} s_{1:t}^i) \ge \frac{1}{2}(1-\varepsilon)$$

for all t.

Proof. Given a probabilistic algorithm of choosing an expert and ε such that $0 < \varepsilon < 1$, define recursively one-step losses s_t^1 and s_t^2 of expert 1 and expert 2 at any step $t = 1, 2, \ldots$ as follows. By $s_{1:t}^1$ and $s_{1:t}^2$ denote the cumulative losses of these experts incurred at steps $\le t$, let v_t be the corresponding volume, where $t = 1, 2, \ldots$.

Define $v_0 = 1$ and $M_t = 4v_{t-1}/\varepsilon$ for all $t \ge 1$. For $t \ge 1$, define $s_t^1 = M_t$ and $s_t^2 = 0$ if $P\{I_t = 1\} \ge \frac{1}{2}$, and define $s_t^1 = 0$ and $s_t^2 = M_t$ otherwise.

Let s_t be one-step loss of the master algorithm and $s_{1:t}$ be its cumulative loss at step $t \ge 1$. We have

$$E(s_{1:t}) \ge E(s_t) = s_t^1 P\{I_t = 1\} + s_t^2 P\{I_t = 2\} \ge \frac{1}{2} M_t$$

for all $t \ge 1$. Also, since $v_t = v_{t-1} + M_t = (1 + 4/\varepsilon) v_{t-1}$ and $\min_i s_{1:t}^i \le v_{t-1}$, the normalized expected regret of the master algorithm is bounded from below

$$\frac{1}{v_t} E(s_{1:t} - \min_i s_{1:t}^i) \ge \frac{2/\varepsilon - 1}{1 + 4/\varepsilon} \ge \frac{1}{2}(1-\varepsilon).$$

for all t. By definition

$$\mathrm{fluc}(t) = \frac{M_t}{v_{t-1} + M_t} = \frac{1}{1 + \varepsilon/4} \ge 1 - \varepsilon$$

for all t. $\triangle$

Proposition 1 shows that we should impose some restrictions of asymptotic behavior of $\mathrm{fluc}(t)$ to prove the asymptotic consistency of a probabilistic algorithm.

3. Follow the Perturbed Leader Algorithm with Adaptive Weights

In this section we construct the FPL algorithm with adaptive weights protected from unbounded one-step losses.

Let $\gamma(t)$ be a computable non-increasing real function such that $0 < \gamma(t) < 1$ for all t. In that follows we usually assume $\gamma(t) \to 0$ as $t \to \infty$; for example, $\gamma(t) = 1/(t+c)^{\delta}$, where $c > 0$ and $\delta > 0$. Let also a be a positive real number. Define

$$\alpha_t = \frac{1}{2}\left(1 - \frac{\ln \frac{a(1+\ln N)}{2(e^{4/a}-1)}}{\ln \gamma(t)}\right) \text{ and} \tag{5}$$

$$\mu_t = a(\gamma(t))^{\alpha_t} = \sqrt{\frac{2a(e^{4/a}-1)}{(1+\ln N)}}(\gamma(t))^{1/2} \tag{6}$$

for all t, where $e = 2.72\ldots$ is the base of the natural logarithm.[2]

Without loss of generality we suppose that $\gamma(t) < \min\{A, A^{-1}\}$ for all t, where

$$A = \frac{2(e^{4/a}-1)}{a(1+\ln N)}.$$

We can obtain this choosing an appropriate value of the initial constant v_0. Then $0 < \alpha_t < 1$ for all t.

We consider an FPL algorithm with a variable learning rate

$$\varepsilon_t = \frac{1}{\mu_t v_{t-1}}, \tag{7}$$

where μ_t is defined by (6) and the volume v_{t-1} depends on experts actions on steps $< t$. By definition $v_t \geq v_{t-1}$ and $\mu_t \leq \mu_{t-1}$ for $t = 1, 2, \ldots$. Also, by definition $\mu_t \to 0$ as $t \to \infty$ if $\gamma(t) \to 0$.

Let $\xi_t^1, \ldots \xi_t^N, t = 1, 2, \ldots$, be a sequence of i.i.d random variables distributed exponentially with the density $p(x) = \exp\{-x\}$. In what follows we omit the lower index t.

We suppose without loss of generality that $s_0^i = v_0 = 0$ for all i and $\varepsilon_0 = \infty$.

The FPL algorithm PROT is defined on Figure 2.

Let $s_{1:T} = \sum_{t=1}^{T} s_t^{I_t}$ be the cumulative loss of the FPL algorithm on steps $\leq T$.

The following theorem shows that if the game is non-degenerate and $\Delta v_t = o(v_t)$ as $t \to \infty$ with a computable bound then the FPL-algorithm with variable learning rate (7) is asymptotically consistent.

We suppose that the experts are oblivious, that is, they do not use in their work random actions of the learning algorithm. The inequality (9) of Theorem 2 below is reformulated and proved for non-oblivious experts at the end this section.

Theorem 2 *Assume that a computable non-increasing real function* $\gamma(t)$ *exists such that* $0 < \gamma(t) < 1$ *and*

$$\mathrm{fluc}(t) \leq \gamma(t) \tag{8}$$

2. The choice of the optimal value of α_t will be explained later. It will be obtained by minimization of the corresponding member of the sum (36).

FPL algorithm PROT.
FOR $t = 1, \dots T$
Choose an expert with the minimal perturbed cumulated loss on steps $< t$

$$I_t = \mathrm{argmin}_{i=1,2,\dots N} \left\{ s^i_{1:t-1} - \frac{1}{\varepsilon_t} \xi^i \right\}.$$

Receive one-step losses s^i_t for experts $i = 1, \dots, N$, define ε_{t+1} by (7), and

$$v_t = v_{t-1} + \max_i s^i_t.$$

Receive one-step loss $s_t = s^{I_t}_t$ of the master algorithm.
ENDFOR

Figure 2: FPL algorithm PROT

for all t. Then for any $\varepsilon > 0$ the expected cumulated loss of the FPL *algorithm* PROT *with variable learning rate (7), where a parameter $a > 0$ depends on ε, is bounded:*

$$E(s_{1:T}) \le \min_i s^i_{1:T} + 2\sqrt{(8+\varepsilon)(1+\ln N)} \sum_{t=1}^{T} (\gamma(t))^{1/2} \Delta v_t \tag{9}$$

for all t.[3]

In case of nonnegative unbounded losses $s^i_t \in [0, +\infty)$ we have a bound

$$E(s_{1:T}) \le \min_i s^i_{1:T} + 2\sqrt{(2+\varepsilon)(1+\ln N)} \sum_{t=1}^{T} (\gamma(t))^{1/2} \Delta v_t. \tag{10}$$

Let also, the game be non-degenerate and $\gamma(t) \to 0$ as $t \to \infty$. Then the algorithm PROT *is asymptotically consistent in the mean*

$$\limsup_{T \to \infty} \frac{1}{v_T} E(s_{1:T} - \min_{i=1,\dots N} s^i_{1:T}) \le 0. \tag{11}$$

Proof. The proof of this theorem follows the proof-scheme of Kalai and Vempala (2003) and Hutter and Poland (2004).

Let α_t be a sequence of real numbers defined by (5); recall that $0 < \alpha_t < 1$ for all t.

The analysis of optimality of the FPL algorithm is based on an intermediate predictor IFPL (Infeasible FPL) (see Figure 3) with the learning rate ε'_t defined by (12).

The IFPL algorithm predicts under the knowledge of $s^i_{1:t}, i = 1, \dots N$, which may not be available at beginning of step t. Using unknown value of ε'_t is the main distinctive feature of our version of IFPL.

The expected one-step and cumulated losses of the FPL and IFPL algorithms at steps t and T are denoted

$$l_t = E(s^{I_t}_t) \text{ and } r_t = E(s^{J_t}_t),$$

3. In case of bounded losses when $\Delta v_t = 1$ we have $v_t = t$ and $\gamma(t) = 1/t$. In this case the regret in the bound (9) has a standard form $O(\sqrt{T \ln N})$.

IFPL algorithm.
FOR $t = 1, \dots T$
Define the learning rate

$$\varepsilon'_t = \frac{1}{\mu_t v_t}, \text{ where } \mu_t = a(\gamma(t))^{\alpha_t}, \tag{12}$$

where v_t is the volume of the game at step t and α_t is defined by (5).
Choose an expert with the minimal perturbed cumulated loss on steps $\leq t$

$$J_t = \text{argmin}_{i=1,2,\dots N}\{s^i_{1:t} - \frac{1}{\varepsilon'_t}\xi^i\}.$$

Receive the one step loss $s^{J_t}_t$ of the IFPL algorithm.
ENDFOR

Figure 3: IFPL algorithm

$$l_{1:T} = \sum_{t=1}^{T} l_t \text{ and } r_{1:T} = \sum_{t=1}^{T} r_t,$$

respectively, where $s^{I_t}_t$ is the one-step loss of the FPL algorithm at step t and $s^{J_t}_t$ is the one-step loss of the IFPL algorithm, and E denotes the mathematical expectation. Recall that $I_t = \text{argmin}_i\{s^i_{1:t-1} - \frac{1}{\varepsilon_t}\xi^i\}$ and $J_t = \text{argmin}_i\{s^i_{1:t} - \frac{1}{\varepsilon'_t}\xi^i\}$.

Lemma 3 *The cumulated expected losses of the* FPL *and* IFPL *algorithms with learning rates defined by (7) and (12) satisfy the inequality*

$$l_{1:T} \leq r_{1:T} + 2(e^{4/a} - 1)\sum_{t=1}^{T}(\gamma(t))^{1-\alpha_t}\Delta v_t \tag{13}$$

for all T, where α_t is defined by (5).

Proof. Let $c_1, \dots c_N$ be arbitrary nonnegative real numbers. For any $1 \leq j \leq N$ define

$$m_j = \min_{i \neq j}\{s^i_{1:t-1} - \frac{1}{\varepsilon_t}c_i\},$$
$$m'_j = \min_{i \neq j}\{s^i_{1:t} - \frac{1}{\varepsilon'_t}c_i\}.$$

Assume that these minima are achieved at $i = j_1$ and $i = j_2$ correspondingly:

$$m_j = s^{j_1}_{1:t-1} - \frac{1}{\varepsilon_t}c_{j_1},$$
$$m'_j = s^{j_2}_{1:t} - \frac{1}{\varepsilon'_t}c_{j_2} = s^{j_2}_{1:t-1} + s^{j_2}_t - \frac{1}{\varepsilon'_t}c_{j_2}$$

for some j_1 and j_2. By definition $j_1 \neq j$ and $j_2 \neq j$. Then we have

$$m_j = s^{j_1}_{1:t-1} - \frac{1}{\varepsilon_t} c_{j_1} \leq s^{j_2}_{1:t-1} - \frac{1}{\varepsilon_t} c_{j_2} \leq$$

$$\leq s^{j_2}_{1:t-1} + s^{j_2}_t + \Delta v_t - \frac{1}{\varepsilon_t} c_{j_2} = \tag{14}$$

$$= s^{j_2}_{1:t} + \Delta v_t - \frac{1}{\varepsilon'_t} c_{j_2} + \left(\frac{1}{\varepsilon'_t} - \frac{1}{\varepsilon_t} \right) c_{j_2} =$$

$$= m'_j + \Delta v_t + \left(\frac{1}{\varepsilon'_t} - \frac{1}{\varepsilon_t} \right) c_{j_2}. \tag{15}$$

We add Δv_t to the right-hand side of the inequality (14) since the term $s^{j_2}_t$ may be negative in case of signed losses. In case of nonnegative losses we need not to do this.

Comparing conditional probabilities

$$P\{I_t = j | \xi^i = c_i, i \neq j\} \text{ and } P\{J_t = j | \xi^i = c_i i \neq j\}$$

is the core of the proof of the lemma.

The following chain of equalities and inequalities is valid:

$$P\{I_t = j | \xi^i = c_i, i \neq j\} =$$

$$= P\{s^j_{1:t-1} - \frac{1}{\varepsilon_t} \xi^j \leq m_j | \xi^i = c_i, i \neq j\} =$$

$$= P\{\xi^j \geq \varepsilon_t (s^j_{1:t-1} - m_j) | \xi^i = c_i, i \neq j\} =$$

$$= P\{\xi^j \geq \varepsilon'_t (s^j_{1:t-1} - m_j) + (\varepsilon_t - \varepsilon'_t)(s^j_{1:t-1} - m_j) | \xi^i = c_i, i \neq j\} \leq \tag{16}$$

$$\leq P\{\xi^j \geq \varepsilon'_t (s^j_{1:t-1} - m_j) +$$

$$+ (\varepsilon_t - \varepsilon'_t)(s^j_{1:t-1} - s^{j_2}_{1:t-1} + \frac{1}{\varepsilon_t} c_{j_2}) | \xi^i = c_i, i \neq j\} = \tag{17}$$

$$= P\{\xi^j \geq \varepsilon'_t (s^j_{1:t-1} - m_j) +$$

$$+ (\varepsilon_t - \varepsilon'_t)(s^j_{1:t-1} - s^{j_2}_{1:t-1}) + (\varepsilon_t - \varepsilon'_t) \frac{1}{\varepsilon_t} c_{j_2} | \xi^i = c_i, i \neq j\} \leq \tag{18}$$

$$\leq P\{\xi^j \geq \varepsilon'_t (s^j_{1:t} - s^j_t - m'_j - \Delta v_t - \left(\frac{1}{\varepsilon'_t} - \frac{1}{\varepsilon_t} \right) c_{j_2}) +$$

$$+ (\varepsilon_t - \varepsilon'_t)(s^j_{1:t-1} - s^{j_2}_{1:t-1}) + (\varepsilon_t - \varepsilon'_t) \frac{1}{\varepsilon_t} c_{j_2} | \xi^i = c_i, i \neq j\} = \tag{19}$$

$$= P\{\xi^j \geq \varepsilon'_t (s^j_{1:t} - m'_j) +$$

$$+ (\varepsilon_t - \varepsilon'_t)(s^j_{1:t-1} - s^{j_2}_{1:t-1}) - \varepsilon'_t (s^j_t + \Delta v_t) | \xi^i = c_i, i \neq j\} =$$

$$= P\{\xi^j > \frac{1}{\mu_t v_t} (s^j_{1:t} - m'_j) +$$

$$+ \left(\frac{1}{\mu_t v_{t-1}} - \frac{1}{\mu_t v_t} \right) (s^j_{1:t-1} - s^{j_2}_{1:t-1}) - \frac{s^j_t + \Delta v_t}{\mu_t v_t} | \xi^i = c_i, i \neq j\} \leq \tag{20}$$

$$\leq P\{\xi^j > \frac{1}{\mu_t v_t} (s^j_{1:t} - m'_j) +$$

$$+\frac{\Delta v_t}{\mu_t v_t}\frac{(s^j_{1:t-1}-s^{j_2}_{1:t-1})}{v_{t-1}}-\frac{2\Delta v_t}{\mu_t v_t}|\xi^i=c_i,i\neq j\}\leq \tag{21}$$

$$\leq \exp\left\{\frac{\Delta v_t}{\mu_t v_t}\left|\frac{s^j_{1:t-1}-s^{j_2}_{1:t-1}}{v_{t-1}}-2\right|\right\}\times \tag{22}$$

$$\times P\{J_t=j|\xi^i=c_i,i\neq j\}. \tag{23}$$

Here the inequality (16)–(17) follows from (14) and $\varepsilon_t\geq\varepsilon'_t$. The inequality (18–(19) follows from (15). We have used in transition from (20) to (21) the equality $v_t-v_{t-1}=\Delta v_t$ and the inequality $|s^j_t|\leq\Delta v_t$ for all j and t. We have used in transition from (21) to (22)–(23) the inequality $P\{\xi\geq a+b\}\leq e^{|b|}P\{\xi\geq a\}$ for any random variable ξ distributed according to the exponential law, where a and b are arbitrary real numbers.[4]

We have in (22)

$$\left|\frac{s^j_{1:t-1}-s^{j_2}_{1:t-1}}{v_{t-1}}\right|\leq 2, \tag{24}$$

since $\left|\frac{s^i_{1:t-1}}{v_{t-1}}\right|\leq 1$ for all t and i. Also, $\Delta v_t/v_t\leq\gamma(t)$ and $\mu_t=a(\gamma(t))^{\alpha_t}$. Therefore, we obtain

$$P\{I_t=j|\xi^i=c_i,i\neq j\}\leq$$
$$\leq\exp\left\{\frac{4}{\mu_t}\frac{\Delta v_t}{v_t}\right\}P\{J_t=j|\xi^i=c_i,i\neq j\}\leq$$
$$\leq\exp\{(4/a)(\gamma(t))^{1-\alpha_t}\}P\{J_t=j|\xi^i=c_i,i\neq j\}. \tag{25}$$

Since, the inequality (25) holds for all c_i, it also holds unconditionally :

$$P\{I_t=j\}\leq\exp\{(4/a)(\gamma(t))^{1-\alpha_t}\}P\{J_t=j\}. \tag{26}$$

for all $t=1,2,\ldots$ and $j=1,\ldots N$.

Since $s^j_t+\Delta v_t\geq 0$ for all j and t, we obtain from (26)

$$l_t+\Delta v_t=E(s^{I_t}_t+\Delta v_t)=\sum_{j=1}^N(s^j_t+\Delta v_t)P(I_t=j)\leq$$
$$\leq\exp\{(4/a)(\gamma(t))^{1-\alpha_t}\}\sum_{j=1}^N(s^j_t+\Delta v_t)P(J_t=j)=$$
$$=\exp\{(4/a)(\gamma(t))^{1-\alpha_t}\}(E(s^{J_t}_t)+\Delta v_t)=$$
$$=\exp\{(4/a)(\gamma(t))^{1-\alpha_t}\}(r_t+\Delta v_t)\leq$$
$$\leq(1+(e^{4/a}-1))(\gamma(t))^{1-\alpha_t})(r_t+\Delta v_t)=$$
$$=r_t+\Delta v_t+(e^{4/a}-1)(\gamma(t))^{1-\alpha_t}(r_t+\Delta v_t)\leq$$
$$\leq r_t+\Delta v_t+2(e^{4/a}-1)(\gamma(t))^{1-\alpha_t}\Delta v_t. \tag{27}$$

4. For $a\leq 0$, we have $P\{\xi\geq a+b\}\leq e^{|b|}P\{\xi\geq a\}$ for all b, since $P\{\xi\geq a\}=1$ and $P\{\xi\geq a+b\}\leq 1$; for $a>0$, $P\{\xi\geq a+b\}\leq e^{-b}P\{\xi\geq a\}$ for all b.

In the last line of (27) we have used the inequality $|r_t| \le \Delta v_t$ for all t and the inequality $e^{sr} \le 1 + (e^s - 1)r$ for all $0 \le r \le 1$ and $s > 0$.

Subtracting Δv_t from both sides of the inequality (27) and summing it by $t = 1, \dots T$, we obtain

$$l_{1:T} \le r_{1:T} + 2(e^{4/a} - 1) \sum_{t=1}^{T} (\gamma(t))^{1-\alpha_t} \Delta v_t$$

for all T. Lemma 3 is proved. $\triangle$

The following lemma, which is an analogue of the result of Kalai and Vempala (2003), gives a bound for the IFPL algorithm.

Lemma 4 *The expected cumulative loss of the* IFPL *algorithm with the learning rate (12) is bounded :*

$$r_{1:T} \le \min_i s^i_{1:T} + a(1 + \ln N) \sum_{t=1}^{T} (\gamma(t))^{\alpha_t} \Delta v_t \tag{28}$$

for all T, where α_t is defined by (5).

Proof. The proof is along the line of the proof of Hutter and Poland (2004) with an exception that now the sequence ε'_t is not monotonic.

Let in this proof, $\mathbf{s_t} = (s^1_t, \dots s^N_t)$ be a vector of one-step losses and $\mathbf{s_{1:t}} = (s^1_{1:t}, \dots s^N_{1:t})$ be a vector of cumulative losses of the experts algorithms. Also, let $\xi = (\xi^1, \dots \xi^N)$ be a vector whose coordinates are random variables.

Recall that $\varepsilon'_t = 1/(\mu_t v_t)$, $\mu_t \le \mu_{t-1}$ for all t, and $v_0 = 0$, $\varepsilon'_0 = \infty$.

Define $\tilde{\mathbf{s}}_{\mathbf{1:t}} = \mathbf{s_{1:t}} - \frac{1}{\varepsilon'_t}\xi$ for $t = 1, 2, \dots$. Consider the vector of one-step losses $\tilde{\mathbf{s}}_{\mathbf{t}} = \mathbf{s_t} - \xi\left(\frac{1}{\varepsilon'_t} - \frac{1}{\varepsilon'_{t-1}}\right)$ for the moment.

For any vector $\mathbf{s}$ and a unit vector $\mathbf{d}$ denote

$$M(\mathbf{s}) = \mathrm{argmin}_{\mathbf{d} \in D}\{\mathbf{d} \cdot \mathbf{s}\},$$

where $D = \{(0, \dots 1), \dots, (1, \dots 0)\}$ is the set of N unit vectors of dimension N and "$\cdot$" is the inner product of two vectors.

We first show that

$$\sum_{t=1}^{T} M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \tilde{\mathbf{s}}_{\mathbf{t}} \le M(\tilde{\mathbf{s}}_{\mathbf{1:T}}) \cdot \tilde{\mathbf{s}_{\mathbf{1:T}}}. \tag{29}$$

For $T = 1$ this is obvious. For the induction step from $T - 1$ to T we need to show that

$$M(\tilde{\mathbf{s}}_{\mathbf{1:T}}) \cdot \tilde{\mathbf{s}}_{\mathbf{T}} \le M(\tilde{\mathbf{s}}_{\mathbf{1:T}}) \cdot \tilde{\mathbf{s}}_{\mathbf{1:T-1}} - M(\tilde{\mathbf{s}}_{\mathbf{1:T-1}}) \cdot \tilde{\mathbf{s}}_{\mathbf{1:T-1}}.$$

This follows from $\tilde{\mathbf{s}}_{\mathbf{1:T}} = \tilde{\mathbf{s}}_{\mathbf{1:T-1}} + \tilde{\mathbf{s}}_{\mathbf{T}}$ and

$$M(\tilde{\mathbf{s}}_{\mathbf{1:T}}) \cdot \tilde{\mathbf{s}}_{\mathbf{1:T-1}} \ge M(\tilde{\mathbf{s}}_{\mathbf{1:T-1}}) \cdot \tilde{\mathbf{s}}_{\mathbf{1:T-1}}.$$

We rewrite (29) as follows

$$\sum_{t=1}^{T} M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \mathbf{s_t} \le M(\tilde{\mathbf{s}}_{\mathbf{1:T}}) \cdot \tilde{\mathbf{s}}_{\mathbf{1:T}} + \sum_{t=1}^{T} M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \xi\left(\frac{1}{\varepsilon'_t} - \frac{1}{\varepsilon'_{t-1}}\right). \tag{30}$$

By definition of M we have

$$M(\tilde{\mathbf{s}}_{\mathbf{1:T}}) \cdot \tilde{\mathbf{s}}_{\mathbf{1:T}} \leq M(\mathbf{s}_{\mathbf{1:T}}) \cdot \left(\mathbf{s}_{\mathbf{1:T}} - \frac{\xi}{\varepsilon'_T} \right) =$$

$$= \min_{\mathbf{d} \in D} \{ \mathbf{d} \cdot \mathbf{s}_{\mathbf{1:T}} \} - M(\mathbf{s}_{\mathbf{1:T}}) \cdot \frac{\xi}{\varepsilon'_T} \ . \tag{31}$$

The expectation of the last term in (31) is equal to $\frac{1}{\varepsilon'_T} = \mu_T v_T$.

The second term of (30) can be rewritten

$$\sum_{t=1}^{T} M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \xi \left(\frac{1}{\varepsilon'_t} - \frac{1}{\varepsilon'_{t-1}} \right) =$$

$$= \sum_{t=1}^{T} (\mu_t v_t - \mu_{t-1} v_{t-1}) M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \xi \ . \tag{32}$$

We will use the standard inequality for the mathematical expectation E

$$0 \leq E(M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \xi) \leq E(\max_i \xi^i) \leq 1 + \ln N. \tag{33}$$

The proof of this inequality uses ideas from Kalai and Vempala (2003) and Hutter and Poland (2004) (Lemma 1).

We have for the exponentially distributed random variables ξ^i, $i = 1, \ldots N$,

$$P\{\max_i \xi^i \geq a\} = P\{\exists i (\xi^i \geq a)\} \leq \sum_{i=1}^{N} P\{\xi^i \geq a\} = N \exp\{-a\}. \tag{34}$$

Since for any non-negative random variable η, $E(\eta) = \int_0^\infty P\{\eta \geq y\} dy$, by (34) we have

$$E(\max_i \xi^i - \ln N) =$$

$$= \int_0^\infty P\{\max_i \xi^i - \ln N \geq y\} dy \leq$$

$$\leq \int_0^\infty N \exp\{-y - \ln N\} dy = 1.$$

Therefore, $E(\max_i \xi^i) \leq 1 + \ln N$.

By (33) the expectation of (32) has the upper bound

$$\sum_{t=1}^{T} E(M(\tilde{\mathbf{s}}_{\mathbf{1:t}}) \cdot \xi)(\mu_t v_t - \mu_{t-1} v_{t-1}) \leq (1 + \ln N) \sum_{t=1}^{T} \mu_t \Delta v_t.$$

Here we have used the inequality $\mu_t \leq \mu_{t-1}$ for all t,

Since $E(\xi^i) = 1$ for all i, the expectation of the last term in (31) is equal to

$$E\left(M(\mathbf{s}_{\mathbf{1:T}}) \cdot \frac{\xi}{\varepsilon'_T} \right) = \frac{1}{\varepsilon'_T} = \mu_T v_T. \tag{35}$$

Combining the bounds (30)-(32) and (35), we obtain

$$r_{1:T} = E\left(\sum_{t=1}^{T} M(\tilde{\mathbf{s}}_{\mathbf{1:t}})\cdot \mathbf{s_t}\right) \leq$$

$$\leq \min_i s^i_{1:T} - \mu_T v_T + (1+\ln N)\sum_{t=1}^{T}\mu_t \Delta v_t \leq$$

$$\leq \min_i s^i_{1:T} + (1+\ln N)\sum_{t=1}^{T}\mu_t \Delta v_t.$$

Lemma is proved. $\triangle$.

We finish now the proof of the theorem.

The inequality (13) of Lemma 3 and the inequality (28) of Lemma 4 imply the inequality

$$E(s_{1:T}) \leq \min_i s^i_{1:T} +$$

$$+\sum_{t=1}^{T}(2(e^{4/a}-1)(\gamma(t))^{1-\alpha_t} + a(1+\ln N)(\gamma(t))^{\alpha_t})\Delta v_t. \tag{36}$$

for all T.

The optimal value (5) of α_t can be easily obtained by minimization of each member of the sum (36) by α_t. In this case μ_t is equal to (6) and (36) is equivalent to

$$E(s_{1:T}) \leq \min_i s^i_{1:T} + 2\sqrt{2a(e^{4/a}-1)(1+\ln N)}\sum_{t=1}^{T}(\gamma(t))^{1/2}\Delta v_t, \tag{37}$$

where a is a parameter of the algorithm PROT.

Also, for each $\varepsilon > 0$ an a exists such that $2a(e^{4/a}-1) < 8+\varepsilon$. Therefore, we obtain (9).

We have $\sum_{t=1}^{T}\Delta v_t = v_T$ for all T, $v_t \to \infty$ and $\gamma(t) \to 0$ as $t \to \infty$. Then by Toeplitz lemma (see Lemma 9 of Section A)

$$\frac{1}{v_T}\left(2\sqrt{(8+\varepsilon)(1+\ln N)}\sum_{t=1}^{T}(\gamma(t))^{1/2}\Delta v_t\right) \to 0$$

as $T \to \infty$. Therefore, the FPL algorithm PROT is asymptotically consistent in the mean, that is, the relation (11) of Theorem 2 is proved.

In case where all losses are nonnegative: $s^i_t \in [0,+\infty)$, the inequality (24) can be replaced on

$$\left|\frac{s^j_{1:t-1} - s^{j_2}_{1:t-1}}{v_{t-1}}\right| \leq 1$$

for all t and i. We need not to add the term Δv_t to the right-hand side of the inequality (14). Also, we need not to add Δv_t to both parts of inequality (27).

In this case an analysis of the proof of Lemma 3 shows that the bound (37) can be replaced on

$$E(s_{1:T}) \leq \min_i s^i_{1:T} + 2\sqrt{a(e^{2/a}-1)(1+\ln N)}\sum_{t=1}^{T}(\gamma(t))^{1/2}\Delta v_t,$$

where a is a parameter of the algorithm PROT.

For each $\varepsilon > 0$ an a exists such that $a(e^{2/a} - 1) < 2 + \varepsilon$. Using this parameter a, we obtain a version of (9) for nonnegative losses—the inequality (10). $\triangle$

We study now the Hannan consistency of our algorithm.

Theorem 5 *Assume that all conditions of Theorem 2 hold and*

$$\sum_{t=1}^{\infty} (\gamma(t))^2 < \infty. \tag{38}$$

Then the algorithm PROT *is Hannan consistent:*

$$\limsup_{T\to\infty} \frac{1}{v_T} \left(s_{1:T} - \min_{i=1,\dots N} s^i_{1:T} \right) \le 0 \tag{39}$$

almost surely.

Proof. So far we assumed that perturbations $\xi^1, \dots, \xi^N$ are sampled only once at time $t = 0$. This choice was favorable for the analysis. As it easily seen, under expectation this is equivalent to generating new perturbations $\xi^1_t, \dots, \xi^N_t$ at each time step t; also, we assume that all these perturbations are i.i.d for $i = 1, \dots, N$ and $t = 1, 2, \dots$. Lemmas 3, 4 and Theorem 2 remain valid for this case. This method of perturbation is needed to prove the Hannan consistency of the algorithm PROT.

We use some version of the strong law of large numbers to prove the Hannan consistency of the algorithm PROT.

Proposition 6 *Let $g(x)$ be a positive nondecreasing real function such that $x/g(x)$, $g(x)/x^2$ are non-increasing for $x > 0$ and $g(x) = g(-x)$ for all x.*

Let the assumptions of Theorem 2 hold and

$$\sum_{t=1}^{\infty} \frac{g(\Delta v_t)}{g(v_t)} < \infty. \tag{40}$$

Then the FPL *algorithm* PROT *is Hannan consistent, that is, (4) holds as $T \to \infty$ almost surely.*

Proof. The proof is based on the following lemma.

Lemma 7 *Let a_t be a nondecreasing sequence of real numbers such that $a_t \to \infty$ as $t \to \infty$ and X_t be a sequence of independent random variables such that $E(X_t) = 0$, for $t = 1, 2, \dots$. Let also, $g(x)$ satisfies assumptions of Proposition 6. Then the inequality*

$$\sum_{t=1}^{\infty} \frac{E(g(X_t))}{g(a_t)} < \infty \tag{41}$$

implies

$$\frac{1}{a_T} \sum_{t=1}^{T} X_t \to 0 \tag{42}$$

as $T \to \infty$ almost surely.

The proof of this lemma is given in Section A.

Put $X_t = (s_t - E(s_t))/2$, where s_t is the loss of the FPL algorithm PROT at step t, and $a_t = v_t$ for all t. By definition $|X_t| \le \Delta v_t$ for all t. Then (41) is valid, and by (42)

$$\frac{1}{v_T}(s_{1:T} - E(s_{1:T})) = \frac{1}{v_T}\sum_{t=1}^{T}(s_t - E(s_t)) \to 0$$

as $T \to \infty$ almost surely. This limit and the limit (11) imply (39). $\triangle$

By Lemma 6 the algorithm PROT is Hannan consistent, since (38) implies (40) for $g(x) = x^2$. Theorem 5 is proved. $\triangle$

Non-asymptotic version of Theorem 5 can be obtained but this requires more heavy technics from probability theory (see Petrov 1975).

4. Specializations of Theorems 2 and 5

In this section we discuss some special cases of Theorems 2 and 5.

In case of bounded experts losses $s_t^i \in [0,1]$, assume that an auxiliary "bad" expert i_0 exists for which $s_t^{i_0} = 1$ for all t. Then $\Delta v_t = 1$ and the volume becomes time: $v_t = t$ for all t (we put $v_0 = 0$). So, we can take $\gamma(t) = t^{-1}$ for all t. In this case the regret (10) of Theorem 2 is equal to $4\sqrt{(2+\varepsilon)(1+\ln N)T}$ that is very close to classical bounds from Hutter and Poland (2004), Kalai and Vempala (2003) and Lugosi and Cesa-Bianchi (2006).

Allenberg et al. (2006) and Poland and Hutter (2005) considered polynomially bounded one-step losses. We consider a specific example of the bound (9) for polynomial case.

Corollary 8 *Assume that $|s_t^i| \le t^\alpha$ for all t and $i = 1,\dots N$, and $v_t \ge t^{\alpha+\delta}$ for all t, where α and δ are positive real numbers. Let also, in the algorithm* PROT, *$\gamma(t) = t^{-\delta}$ and μ_t is defined by (6). Then*

- *(i) the algorithm* PROT *is asymptotically consistent in the mean for any $\alpha > 0$ and $\delta > 0$;*
- *(ii) this algorithm is Hannan consistent for any $\alpha > 0$ and $\delta > \frac{1}{2}$;*
- *(iii) the expected loss of this algorithm is bounded :*

$$E(s_{1:T}) \le \min_i s_{1:T}^i + 2\sqrt{(8+\varepsilon)(1+\ln N)}T^{1-\frac{1}{2}\delta+\alpha} \tag{43}$$

as $T \to \infty$, where $\varepsilon > 0$ is a parameter of the algorithm.[5]

This corollary follows directly from Theorem 2, where condition (38) of Theorem 2 holds for $\delta > \frac{1}{2}$.

If $\delta = 1$ the regret from (43) is asymptotically equivalent to the regret from Allenberg et al. (2006) (see Section 1).

For $\alpha = 0$ we have the case of bounded loss function ($|s_t^i| \le 1$ for all i and t). The FPL algorithm PROT is asymptotically consistent in the mean if $v_t \ge \beta(t)$ for all t, where $\beta(t)$ is an arbitrary positive unbounded non-decreasing computable function (we can get $\gamma(t) = 1/\beta(t)$ in this case). This algorithm is Hannan consistent if (38) holds, that is,

$$\sum_{t=1}^{\infty}(\beta(t))^{-2} < \infty.$$

5. Recall that given ε we tune the parameter a of the algorithm PROT.

For example, this condition be satisfied for $\beta(t) = t^{1/2}\ln t$.

Let us show that the bound (9) of Theorem 2 that holds against oblivious experts also holds against non-oblivious (adaptive) ones.

In non-oblivious case, it is natural to generate at each time step t of the algorithm PROT a new vector of perturbations $\xi_{\mathbf{t}} = (\xi_t^1, \ldots, \xi_t^N)$, ξ_0 is empty set. Also, it is assumed that all these perturbations are i.i.d according to the exponential distribution P, where $i = 1, \ldots, N$ and $t = 1, 2, \ldots$. Denote $\xi_{\mathbf{1:t}} = (\xi_1, \ldots, \xi_t)$.

Non-oblivious experts can react at each time step t on past decisions $s_1, s_2, \ldots s_{t-1}$ of the FPL algorithm and on values of $\xi_1, \ldots, \xi_{t-1}$.

Therefore, losses of experts and regret depend now from random perturbations:

$$s_t^i = s_t^i(\xi_{\mathbf{1:t-1}}),\ i = 1, \ldots, N,$$
$$\Delta v_t = \Delta v_t(\xi_{\mathbf{1:t-1}}),$$

where $t = 1, 2, \ldots$.

In non-oblivious case, condition (8) is a random event. We assume in Theorem 2 that in the game of prediction with expert advice regulated by the FPL-protocol the event

$$\mathrm{fluc}(t) \le \gamma(t) \text{ for all } t$$

holds almost surely.

An analysis of the proof of Theorem 2 shows that in non-oblivious case, the bound (9) is an inequality for the random variable

$$\sum_{t=1}^{T} E(s_t) - \min_i s_{1:T}^i -$$
$$-2\sqrt{(8+\varepsilon)(1+\ln N)} \sum_{t=1}^{T} (\gamma(t))^{1/2} \Delta v_t \le 0,$$

which holds almost surely with respect to the product distribution P^{t-1}, where the loss of the FPL algorithm s_t depend on a random perturbation ξ_t at step t and on losses of all experts on steps $< t$. Also, E is the expectation with respect to P.

Taking expectation $E_{1:T-1}$ with respect to the product distribution P^{t-1} we obtain a version of (9) for non-oblivious case

$$E_{1:T}\left(s_{1:T} - \min_i s_{1:T}^i - 2\sqrt{(8+\varepsilon)(1+\ln N)} \sum_{t=1}^{T} (\gamma(t))^{1/2} \Delta v_t \right) \le 0$$

for all T.

5. An Example: Zero-sum Experts

In this section we present an example of a game, where losses of experts cannot be bounded in advance.[6]

6. This example is a modified version of an example from V'yugin (2009a).

Let $S = S(t)$ be a function representing evolution of a stock price. Two experts will represent two concurrent methods of buying and selling shares of this stock.

Let M and T be positive integer numbers and let the time interval $[0,T]$ be divided on a large number M of subintervals. Define a discrete time series of stock prices

$$S_0 = S(0), S_1 = S(T/(M)), S_2 = S(2T/(M)) \ldots, S_M = S(T). \tag{44}$$

In this paper, volatility is an informal notion. We say that the difference $(S_T - S_0)^2$ represents the macro volatility and the sum $\sum_{i=0}^{T-1} (\Delta S_i)^2$, where $\Delta S_i = S_{i+1} - S_i$, $i = 1, \ldots T-1$, represents the micro volatility of the time series (44).

The game between an investor and the market looks as follows: the investor can use the long and short selling. At beginning of time step t Investor purchases the number C_t of shares of the stock by S_{t-1} each. At the end of trading period the market discloses the price S_{t+1} of the stock, and the investor incur his current income or loss $s_t = C_t \Delta S_t$ at the period t. We have the following equality

$$(S_T - S_0)^2 = (\sum_{t=0}^{T-1} \Delta S_t)^2 =$$
$$= \sum_{t=0}^{T-1} 2(S_t - S_0)\Delta S_t + \sum_{t=0}^{T-1} (\Delta S_t)^2. \tag{45}$$

The equality (45) leads to the two strategies for investor which are represented by two experts. At the beginning of step t Experts 1 and 2 hold the number of shares

$$C_t^1 = 2C(S_t - S_0), \tag{46}$$
$$C_t^2 = -C_t^1, \tag{47}$$

where C is an arbitrary positive constant.

These strategies at step t earn the incomes $s_t^1 = 2C(S_t - S_0)\Delta S_t$ and $s_t^2 = -s_t^1$. The strategy (46) earns in first T steps of the game the income

$$s_{1:T}^1 = \sum_{t=1}^{T} s_t^1 = 2C((S_T - S_0)^2 - \sum_{t=1}^{T-1} (\Delta S_t)^2).$$

The strategy (47) earns in first T steps the income $s_{1:T}^2 = -s_{1:T}^1$.

The number of shares C_t^1 in the strategy (46) or number of shares $C_t^2 = -C_t^1$ in the strategy (47) can be positive or negative. The one-step gains s_t^1 and $s_t^2 = -s_t^1$ are unbounded and can be positive or negative: $s_t^i \in (-\infty, +\infty)$.

Informally speaking, the first strategy will show a large return if

$$(S_T - S_0)^2 \gg \sum_{i=0}^{T-1} (\Delta S_i)^2;$$

the second one will show a large return when

$$(S_T - S_0)^2 \ll \sum_{i=0}^{T-1} (\Delta S_i)^2.$$

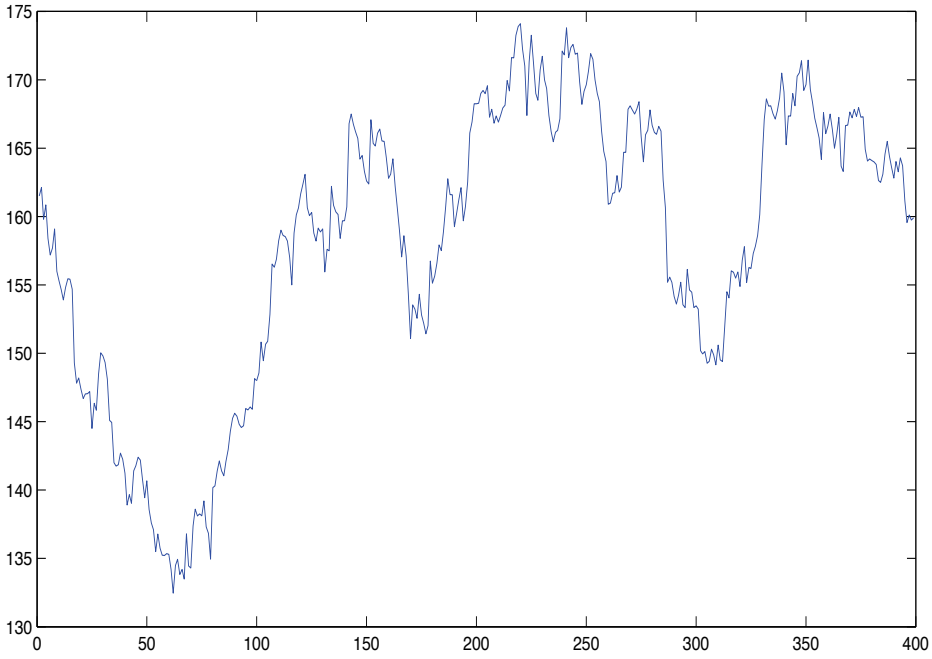


Figure 4: Evolution of Gazprom stock price

There is an uncertainty domain for these strategies, that is, a case when both $\gg$ and $\ll$ do not hold. The idea of these strategies is based on the paper of Cheredito (2003) (see also Rogers 1997 and Delbaen and Schachermayer 1994) who have constructed arbitrage strategies for a financial market that consists of money market account and a stock whose price follows a fractional Brownian motion with drift or an exponential fractional Brownian motion with drift. Vovk (2003) has reformulated these strategies for discrete time. We use these strategies to define a mixed strategy which incur gain when macro and micro volatilities of time series differ. There is no uncertainty domain for continuous time.

We analyze this game in the decision theoretic online learning (DTOL) framework (Freund and Schapire, 1997). We introduce *Learner* that can choose between two strategies (46) and (47). To change from follow the leader framework to DTOL we derandomize the FPL algorithm PROT.[7] We interpret the expected one-step gain $E(s_t)$ gain as the weighted average of one-step gains of experts strategies. In more detail, at each step t, *Learner* divide his investment in proportion to the probabilities of expert strategies (46) and (47) computed by the FPL algorithm and suffers the gain

$$G_t = 2C(S_t - S_0)(P\{I_t = 1\} - P\{I_t = 2\})\Delta S_t$$

at any step t, where C is an arbitrary positive constant; $G_{1:T} = \sum_{t=1}^{T} G_t = E(s_{1:T})$ is the *Learner's* cumulative gain.

7. To apply Theorem 2 we interpreted gain as a negative loss.

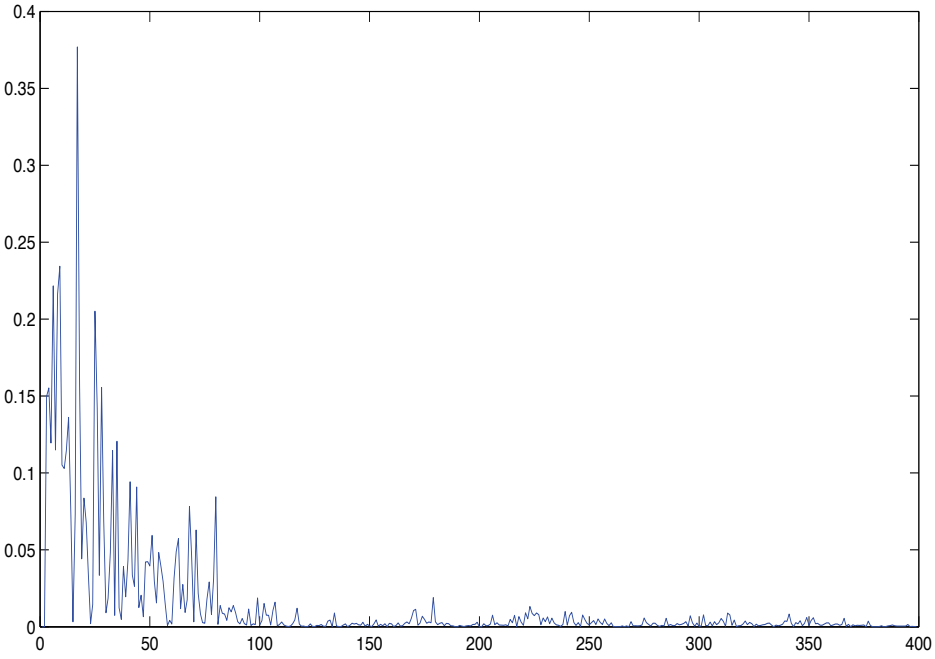


Figure 5: Fluctuation of the game

Assume that $|s_t^1| = o(\sum_{i=1}^t |s_i^1|)$ as $t \to \infty$. Let $\gamma(t) = \mu$ for all t, where μ is arbitrary small positive number. Then for any $\varepsilon > 0$

$$G_{1:T} \geq \left| \sum_{t=1}^{T} s_t^1 \right| - 2\mu^{1/2} \sqrt{(8+\varepsilon)(1+\ln N)} \left(\sum_{t=1}^{T} |s_t^1| + v_0 \right)$$

for all sufficiently large T, and for some $v_0 \geq 0$.

Under condition of Theorem 2 we show that strategy of algorithm PROT is "defensive" in some weak sense :

$$G_{1:T} - \left| \sum_{t=1}^{T} s_t^1 \right| \geq -o\left(\sum_{t=1}^{T} |s_t^1| + v_0 \right) \tag{48}$$

as $T \to \infty$.

Some experimental results are shown on Figures 4–6. The strategies (46) and (47) were applied to the Russian Gazprom stock (ticker symbol—GAZP) downloaded from FINAM site.[8] We get $C = 600$. We have used the stock closing price time series on period from 02 July to 02 September 2009 with periodicity 60 minutes between two neighboring time-points; the size of time series is 400 points. The stock price was volatile during the playing period, its value changed slightly during this period from 163.45 Rub to 159.90 Rub (see Figure 4).

8. FINAM is at `http://www.finam.ru/analysis/export/default.asp`.

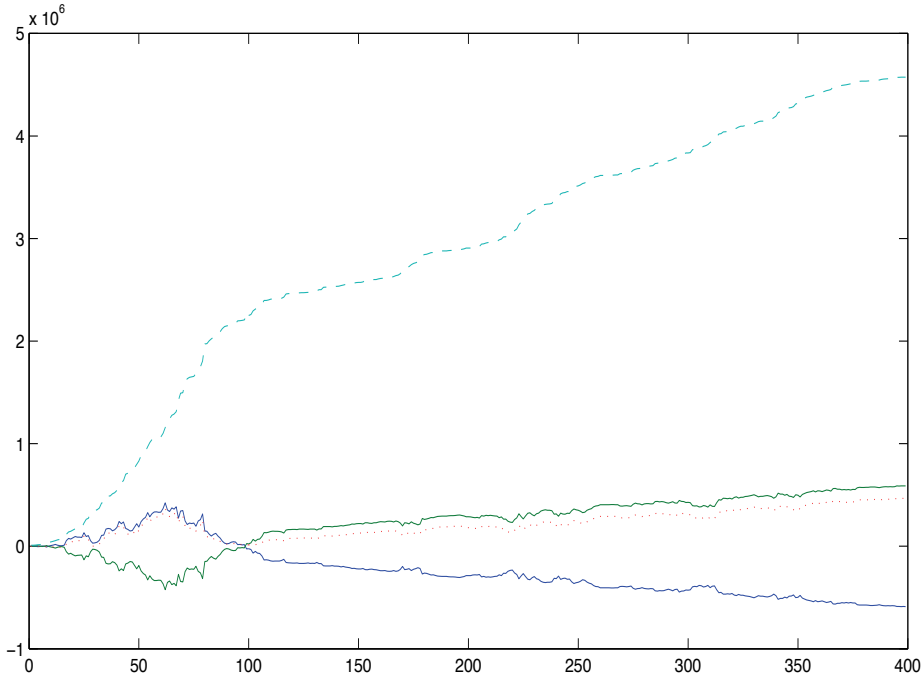


Figure 6: Two symmetric solid lines—gains of two zero sums strategies, dotted line—expected gain of the algorithm PROT (without transaction costs), dashed line—volume of the game

Two symmetric solid lines on Figure 6 are gains of two zero sums strategies (46) and (47), dotted line—expected gain of the algorithm PROT, dashed line—volume of the game. The scaled fluctuation of the game is presented on Figure 5. We get $\gamma(t) = t^{-1/2}$. The first strategy (46) was favorite at about 100 first steps, the second strategy (47) was favorite at the rest of the playing period. The algorithm PROT suffered sufficiently large income—456970 Rub (without transaction costs) (see Figure 6) and 230099 Rub when transaction costs were subtracted.

6. Conclusion

In this paper we try to extend methods of the theory of prediction with expert advice for the case when experts one-step gains cannot be bounded in advance. The traditional measures of performance are invalid for general unbounded case.

To measure the asymptotic performance of our algorithm, we replace the traditional time-scale on a volume-scale. New notion of volume of a game and scaled fluctuation of a game are introduced in this paper. In case of two zero-sum experts, the volume equals to the sum of all transactions between experts.

Using the notion of the scaled fluctuation of a game, we can define very broad classes of games (experts) for which our algorithm PROT is asymptotically consistent in the modified sense. The

restrictions imposed on such games are formulated in a relative form: "the logarithmic derivative" of the volume of the game must be $o(t)$ as $t \to \infty$.

Our work supplements results of Cesa-Bianchi et al. (2007), where the bounds for a regret were obtained under the very general assumptions. Authors of this paper do not study asymptotic consistency of their algorithm. Our bounds for the regret are defined in terms of a volume of the game and our learning algorithm is asymptotically consistent in the mean and almost surely.

Algorithms for unbounded losses have appeared in the literature, but none of the papers deal with FPL and "fast-growing" losses. Looking closely at the requirements of this paper, the quantity $\mathrm{fluc}(t)$ has to decrease to 0, which to imply that the rate of growth of the losses has to be slower than exponential. Given the results of Allenberg et al. (2006), who can deal with polynomial growth of loss, this paper is more general in the regime "faster than polynomial, but slower than exponential".

A motivating example of a game with two zero-sum experts from Section 5 shows some practical significance of these problems. The FPL algorithm with variable learning rates is simple to implement and it is bringing satisfactory experimental results when prices follow fractional Brownian motion. The drawback of this playing strategy is that the defense condition (48) is too weak—it has only an asymptotic form. In cases, where regimes of high and low volatilities quickly changing the algorithm PROT may suffer a large loss. This is an open problems for further research: how to construct a defensive strategy for *Learner* in sense of Shafer and Vovk (2001)? This means that *Learner* starting with some initial capital never goes to debt and suffer a gain when macro and micro volatilities differ.

There are other open problems for further research. Can we incorporate our results obtained in fluctuation-volume setting into the framework presented in Cesa-Bianchi et al. (2007), where a powerful technics for the Weighed Majority algorithm based on second order quantity—variance of losses, was developed?

We have used the FPL algorithm, since its analysis remains easy for an adaptive learning rate, in contrast to WM-derivatives. It would be useful to analyze the performance of the well known algorithms from DTOL framework (like "Hedge" of Freund and Schapire 1997 or "Normal Hedge" of Chaudhuri et al. 2009) for the case of unbounded losses in terms of the volume and scaled fluctuation of a game.

Some improvement of the regret (9) can be achieved using in (27) a more tight bound of the exponent $e^r \le 1 + r + (e-2)r^2$ (for $|r| \le 1$) in place of the linear bound used in the proof of Lemma 3.

There is a gap between Proposition 1 and Theorem 2, since we assume in this theorem that the game satisfies $\mathrm{fluc}(t) \le \gamma(t) \to 0$, where $\gamma(t)$ is computable. Also, the function $\gamma(t)$ is a parameter of our algorithm PROT. Does there exists an asymptotically consistent learning algorithm in case where $\limsup_{t\to\infty} \mathrm{fluc}(t) = 0$ and where the function $\gamma(t)$ is not a parameter of this algorithm?

Can we apply "double trick" method for the sequence $\mathrm{fluc}(t)$, $t = 1,2,\ldots$, to avoid parameter $\gamma(t)$ from the learning algorithm is an open question. A problem is that $\mathrm{fluc}(t)$ is not monotone though $\limsup_{t\to\infty} \mathrm{fluc}(t) = 0$.

Let $\gamma_i(t)$ be any computable (by i and t) sequence of non-increasing (by t) functions such that for any i, $0 < \gamma_i(t) \le 1$ for all t and $\gamma_i(t) \to 0$ as $t \to \infty$.[9] We can construct a version of the algorithm

9. A case $\sup_i \gamma_i(t) = 1$ for all t is possible for these functions.

PROT which is asymptotically consistent in the mean for any game satisfying

$$\limsup_{t\to\infty} \frac{\mathrm{fluc}(t)}{\gamma_i(t)} < \infty \tag{49}$$

for some i. To solve this problem define a computable non-increasing function $\gamma(t)$ such that

- $0 < \gamma(t) \le 1$,
- $\gamma(t) \to 0$ as $t \to \infty$,
- for any i there exists an t_i such that $\gamma(t) \ge \gamma_i(t)$ for all $t \ge t_i$.

Evidently, the algorithm PROT with the parameter $\gamma(t)$ is asymptotically consistent in the mean for any game such that (49) holds for some i.

We consider in this paper only the full information case. An analysis of these problems under partial monitoring is a subject for a further research.

Acknowledgments

This research was partially supported by Russian foundation for fundamental research: 09-07-00180-a and 09-01-00709a.

Appendix A. Proof of Lemma 7

The proof of Lemma 7 is based on Kolmogorov's theorem on three series and its corollaries. For completeness of presentation we reconstruct the proof from Petrov (1975) (Chapter IX, Section 2).

For any random variable X and a positive number c denote

$$X^c = \begin{cases} X \text{ if } |X| \le c \\ 0 \text{ otherwise.} \end{cases}$$

The Kolmogorov's theorem on three series says:

For any sequence of independent random variables X_t, $t = 1,2,\ldots$, the following implications hold

- If the series $\sum_{t=1}^{\infty} X_t$ is convergent almost surely then the series $\sum_{t=1}^{\infty} EX_t^c$, $\sum_{t=1}^{\infty} DX_t^c$ and $\sum_{t=1}^{\infty} P\{|X_t| \ge c\}$ are convergent for each $c > 0$, where E is the mathematical expectation and D is the variation.
- The series $\sum_{t=1}^{\infty} X_t$ is convergent almost surely if all these series are convergent for some $c > 0$.

See Shiryaev (1980) for the proof.

Assume conditions of Lemma 7 hold. We will prove that

$$\sum_{t=1}^{\infty} \frac{Eg(X_t)}{g(a_t)} < \infty \tag{50}$$

implies

$$\sum_{t=1}^{\infty} \frac{X_t}{a_t} < \infty$$

almost surely. From this, by Kroneker's lemma 10 (see below), the series

$$\frac{1}{a_t} \sum_{t=1}^{\infty} X_t \tag{51}$$

is convergent almost surely.

Let V_t be a distribution function of the random variable X_t. Since g non-increases,

$$P\{|X_t| > a_t\} \le \int_{|x| \ge a_t} \frac{g(x)}{g(a_t)} dV_t(x) \le \frac{Eg(X_t)}{g(a_t)}.$$

Then by (50)

$$\sum_{t=1}^{\infty} P\left\{\left|\frac{X_t}{a_t}\right| \ge 1\right\} < \infty \tag{52}$$

almost surely. Denote

$$Z_t = \begin{cases} X_t \text{ if } |X_t| \le a_t \\ 0 \text{ otherwise.} \end{cases}$$

By definition $x^2/g(x)) \le a_t/g(a_t)$ for $|x| < a_t$. Rearranging, we obtain $x^2/a_t \le g(x)/g(a_t)$ for these x. Therefore,

$$EZ_t^2 = \int\limits_{|x|<a_t} x^2 dV_t(x) \le \frac{a_t^2}{g(a_t)} \int\limits_{|x|<a_t} g(x) dV_t(x) \le \frac{a_t^2}{g(a_t)} Eg(X_t).$$

By (50) we obtain

$$\sum_{t=1}^{\infty} E\left(\frac{Z_t}{a_t}\right)^2 < \infty. \tag{53}$$

Since $EX_t = \int\limits_{-\infty}^{\infty} x dV_t(x) = 0$,

$$|EZ_t| = \left| \int\limits_{|x|>a_t} x dV_t(x) \right| \le \frac{a_t}{g(a_t)} \int\limits_{|x|>a_t} g(x) dV_t(x) \le \frac{a_t}{g(a_t)} Eg(X_t). \tag{54}$$

By (50)

$$\sum_{t=1}^{\infty} E\left(\frac{X_t}{a_t}\right)^1 \le \sum_{t=1}^{\infty} \left| E\left(\frac{Z_t}{a_t}\right) \right| < \infty.$$

From (52)–(54) and the theorem on three series we obtain (51).

We have used Toeplitz and Kroneker's lemmas.

Lemma 9 *(Toeplitz) Let x_t be a sequence of real numbers and b_t be a sequence of nonnegative real numbers such that $a_t = \sum_{i=1}^{t} b_i \to \infty$, $x_t \to x$ and $|x| < \infty$. Then*

$$\frac{1}{a_t} \sum_{i=1}^{t} b_i x_i \to x. \tag{55}$$

Proof. For any $\varepsilon > 0$ an t_ε exists such that $|x_t - x| < \varepsilon$ for all $t \geq t_\varepsilon$. Then

$$\left| \frac{1}{a_t} \sum_{i=1}^{t} b_i (x_i - x) \right| \leq \frac{1}{a_t} \sum_{i<t_\varepsilon} |b_i (x_i - x)| + \varepsilon$$

for all $t \geq t_\varepsilon$. Since $a_t \to \infty$, we obtain (55).

Lemma 10 *(Kroneker) Assume $\sum_{t=1}^{\infty} x_t < \infty$ and $a_t \to \infty$ Then*

$$\frac{1}{a_t} \sum_{i=1}^{t} a_i x_i \to 0.$$

The proof is the straightforward corollary of Toeplitz lemma.

References

C. Allenberg, P. Auer, L. Gyorfi and Gy. Ottucsak. Hannan consistency in on-Line learning in case of unbounded losses under partial monitoring. In Jose L. Balcazar, Philip M. Long, Frank Stephan, editors, *Proceedings of the 17th International Conference ALT 2006, Lecture Notes in Computer Science 4264*, pages 229–243, Springer-Verlag, Berlin, Heidelberg, 2006.

K. Chaudhuri, Y. Freund and D. Hsu. A parameter-free hedging algorithm. *Technical Report arXiv:0903.2851v1 [cs.LG]*, March 2009.

P. Cheredito. Arbitrage in fractional Brownian motion. *Finance and Statistics*, 7:533–553, 2003.

N. Cesa-Bianchi, Y. Mansour and G. Stoltz. Improved second-order bounds for prediction with expert advice. *Machine Learning*, 66(2–3):321–352, 2007.

F. Delbaen and W. Schachermayer. A general version of the fundamental theorem of asset pricing. *Mathematische Annalen*, 300:463–520, 1994.

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*, 55:119–139, 1997.

J. Hannan. Approximation to Bayes risk in repeated plays. In M. Dresher, A.W. Tucker, and P. Wolfe, editors, *Contributions to the Theory of Games 3*, pages 97-139, Princeton University Press, 1957.

M. Hutter and J. Poland. Prediction with expert advice by following the perturbed leader for general weights. In S.Ben-Dawid, J.Case, A.Maruoka, editors *Proceedings of the 15th International Conference ALT 2004, Lecture Notes in Artificial Intelligence 3244*, pages 279–293, Springer-Verlag, Berlin, Heidelberg, 2004.

M. Hutter and J. Poland. Adaptive online prediction by following the perturbed leader. *Journal of Machine Learning Research*, 6:639–660, 2005.

A. Kalai and S. Vempala. Efficient algorithms for online decisions. In Bernhard Scholkopf, Manfred K. Warmuth, editors, *Proceedings of the 16th Annual Conference on Learning Theory COLT 2003, Lecture Notes in Computer Science 2777*, pages 506–521, Springer-Verlag, Berlin, 2003. Extended version in *Journal of Computer and System Sciences*, 71:291–307, 2005.

N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

G. Lugosi and N. Cesa-Bianchi. *Prediction, Learning and Games*. Cambridge University Press, New York, 2006.

V.V. Petrov. Sums of independent random variables. *Ergebnisse der Mathematik und Ihrer Grenzgebiete*, Band 82, Springer-Verlag, New York, Heidelberg, Berlin, 1975.

J. Poland and M. Hutter. Defensive universal learning with experts for general weights. In S.Jain, H.U.Simon and E.Tomita, editors, *Proceedings of the 16th International Conference ALT 2005, Lecture Notes in Artificial Intelligence 3734*, pages 356–370, Springer-Verlag, Berlin, Heidelberg, 2005.

C. Rogers. Arbitrage with fractional Brownian motion. *Mathematical Finance*, 7:95–105, 1997.

G. Shafer and V. Vovk. *Probability and Finance. It's Only a Game!*, Wiley, New York, 2001.

A.N. Shiryaev. *Probability*. Springer-Verlag, Berlin, 1980.

V. Vovk. Aggregating strategies. In M. Fulk and J. Case, editors, *Proceedings of the 3rd Annual Workshop on Computational Learning Theory*, pages 371–383, Morgan Kaufmann, San Mateo, CA, 1993.

V. Vovk. A game-theoretic explanation of the $\sqrt{dt}$ effect. Working paper #5, Available online at `http://www.probabilityandfinance.com`, 2003.

V.V. V'yugin. The follow perturbed leader algorithm protected from unbounded one-step losses. In Ricard Gavalda, Gabor Lugosi, Thomas Zeugmann, and Sandra Zilles, editors, *Proceedings of the 20th International Conference ALT 2009, Lecture Notes in Artificial Intelligence 5809*, pages. 38-52, Springer-Verlag, Berlin Heidelberg, 2009.

V.V. V'yugin. Learning volatility of discrete time series using prediction with expert advice. In O. Watanabe and T. Zeugmann, editors, *SAGA 2009, Lecture Notes in Computer Science 5792*, pages 16-30, Springer-Verlag, Berlin, Heidelberg, 2009.

Journal of Machine Learning Research 12 (2011) 267-300 Submitted 9/09; Revised 7/10; Published 1/11

A Bayesian Approximation Method for Online Ranking

Ruby C. Weng CHWENG@NCCU.EDU.TW
Department of Statistics
National Chengchi University
Taipei 116, Taiwan

Chih-Jen Lin CJLIN@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Editor: Thore Graepel

Abstract

This paper describes a Bayesian approximation method to obtain online ranking algorithms for games with multiple teams and multiple players. Recently for Internet games large online ranking systems are much needed. We consider game models in which a k-team game is treated as several two-team games. By approximating the expectation of teams' (or players') performances, we derive simple analytic update rules. These update rules, without numerical integrations, are very easy to interpret and implement. Experiments on game data show that the accuracy of our approach is competitive with state of the art systems such as TrueSkill, but the running time as well as the code is much shorter.

Keywords: Bayesian inference, rating system, Bradley-Terry model, Thurstone-Mosteller model, Plackett-Luce model

1. Introduction

Many have proposed online updating algorithms for paired comparison experiments. These online algorithms are especially useful when the number of teams to be ranked and the number of games are very large. For the ranking of many sports, possibly the most prominent ranking system in use today is Elo (1986). The Elo ranking system has been used successfully by leagues organized around two-player games, such as world football league, the US Chess Federation (USCF) or the World Chess Federation (FIDE), and a variety of others. Glickman (1999) proposed the Glicko updating system, which improves over Elo by incorporating the variability in parameter estimates. To the best of our knowledge, Glicko is the first Bayesian ranking system. To begin, prior to a rating period, a player's skill (θ) is assumed to follow a Gaussian distribution which can be characterized by two numbers: the average skill of the player (μ) and the degree of uncertainty in the player's skill (σ). Then, Glicko models the game outcomes by the Bradley-Terry model (Bradley and Terry, 1952) and updates players' skills after a rating period. Glickman (1999) also reported that the Glicko system performs best when the number of games per player is around 5-10 in a rating period. Though the Elo and Glicko ranking systems have been successful, they are designed for two-player games. In video games a game often involves more than two players or teams. To address this problem, recently Microsoft Research developed TrueSkill (Herbrich et al., 2007), a ranking system for Xbox

©2011 Ruby C. Weng and Chih-Jen Lin.

Live. TrueSkill is also a Bayesian ranking system using a Gaussian belief over a player's skill, but it differs from Glicko in several ways. First, it is designed for multi-team/multi-player games, and it updates skills after each game rather than a rating period. Secondly, Glicko assumes that the performance difference follows the logistic distribution (the model is termed the Bradley-Terry model), while TrueSkill uses the Gaussian distribution (termed the Thurstone-Mosteller model). Moreover, TrueSkill models the draws and offers a way to measure the quality of a game between any set of teams. The way TrueSkill estimates skills is by constructing a graphical model and using approximate message passing. In the easiest case, a two-team game, the TrueSkill update rules are fairly simple. However, for games with multiple teams and multiple players, the update rules are not possible to write down as they require an iterative procedure.

The present paper concerns the ranking of players from outcomes of multiple players or games. We consider a k-team game as a single match and discuss the possibility of obtaining efficient update algorithms. We introduce a Bayesian approximation method to derive simple analytic rules for updating team strength in multi-team games. These update rules avoid a numerical integration and are easy to interpret and implement. Strength of players in a team are then updated by assuming that a team's skill is the sum of skills of ts members. Our framework can be applied by considering various ranking models. In this paper, we demonstrate the use of the Bradley-Terry model, the Thurstone-Mosteller model, and the Plackett-Luce model. Experiments on game data show that the accuracy of our approach is competitive with the TrueSkill ranking system, but the running time as well as the code are shorter. Our method is faster because we employ analytic update rules rather than iterative procedures in TrueSkill.

The organization of this paper is as follows. In Section 2, we briefly review the modeling of ranked data. Section 3 presents our approximation method and gives update equations of using the Bradley-Terry model. Update rules of using other ranking models are given in Section 4. As Glicko is also based on the Bradley-Terry model, for a comparison purpose we describe its approximation procedures in Section 5. Experimental studies are provided in Section 6. Section 7 concludes the paper. Some notation is given in Table 1.

2. Review of Models and Techniques

This section reviews existing methods for modeling ranked data and discusses approximation techniques for Bayesian inference.

2.1 Modeling Ranked Data

Given the game outcome of k teams, we define $r(i)$ as the rank of team i. If teams $i_1,\ldots,i_d$ are tied together, we have

$$r(i_1) = \cdots = r(i_d),$$

and let the team q ranked next have

$$r(q) = r(i_1) + d.$$

For example, if four teams participate in a game, their ranks may be

$$r(1) = 2, r(2) = 2, r(3) = 4, r(4) = 1, \tag{1}$$

where teams 1 and 2 are both ranked the second. Then team 3, which ranked the next, has $r(3) = 4$. We also need the "inverse" of r, so that $r(i)$ indicates the index of the ith ranked team. However, the

Notation	Explanation
k	number of teams participating in a game
n_i	number of players in team i
θ_{ij}	strength of the jth player in team i
$N(\mu_{ij},\sigma_{ij}^2)$	prior distribution of θ_{ij}
Z_{ij}	standardized quantity of θ_{ij}; see (45)
θ_i	strength of team i; $\theta_i=\sum_{j=1}^{n_i}\theta_{ij}$
β_i^2	uncertainty about the performance of team i
X_i	performance of team i ($X_i\sim N(\theta_i,\beta_i^2)$ for Thurstone-Mosteller model)
$N(\mu_i,\sigma_i^2)$	prior distribution of θ_i
μ_i	$\sum_{j=1}^{n_i}\mu_{ij}$
σ_i^2	$\sum_{j=1}^{n_i}\sigma_{ij}^2$
Z_i	standardized quantity of θ_i; see (24)
$r(i)$	rank of team i in a game; smaller is better; see Section 2.1
$r(i)$:	index of the ith ranked team; "inverse" of r; see Section 2.1
ε	draw margin (Thurstone-Mosteller model)
ϕ	probability density function of a standard normal distribution; see (66)
Φ	cumulative distribution function of a standard normal distribution
ϕ_k	probability density function of a k-variate standard normal distribution
Φ_k	cumulative distribution function of a k-variate standard normal distribution
κ	a small positive value to avoid σ_i^2 becoming negative; see (28) and (44)
D	the game outcome
$E(\cdot)$	expectation with respect to a random variable

Table 1: Notation

function r is not one-to-one if ties occur, so the inverse is not directly available. We choose r to be any one-to-one mapping from $\{1,\ldots,k\}$ to $\{1,\ldots,k\}$ satisfying

$$r(r(i))\le r(r(i+1)),\forall i. \tag{2}$$

For example, if r is as in Equation (1), then r could be

$$r(1)=4,r(2)=1,r(3)=2,r(4)=3.$$

We may have $r(2)=2$ and $r(3)=1$ instead, though in this paper choosing any r satisfying (2) is enough.

A detailed account of modeling ranked data is by Marden (1995). For simplicity, in this section we assume that ties do not occur though ties are handled in later sections. Two most commonly used models for ranked data are the Thurstone-Mosteller model (Thurstone, 1927) and the Bradley-Terry model. Suppose that each team is associated with a continuous but unobserved random variable X_i, representing the actual performance. The observed ordering that team $r(1)$ comes in first, team $r(2)$ comes in second and so on is then determined by the X_i's:

$$X_{r(1)}>X_{r(2)}>\cdots>X_{r(k)}. \tag{3}$$

Thurstone (1927) invented (3) and proposed using the normal distribution. The resulting likelihood associated with (3) is

$$P(X_{r(1)} - X_{r(2)} > 0, \ldots, X_{r(k-1)} - X_{r(k)} > 0), \tag{4}$$

where $X_{r(i)} - X_{r(i+1)}$ follows a normal distribution. In particular, if $k = 2$ and X_i follows $N(\theta_i, \beta_i^2)$, where θ_i is the strength of team i and β_i^2 is the uncertainty of the actual performance X_i, then

$$P(X_i > X_q) = \Phi\left(\frac{\theta_i - \theta_q}{\sqrt{\beta_i^2 + \beta_q^2}}\right), \tag{5}$$

where Φ denotes the cumulative distribution function of a standard normal density.

Numerous papers have addressed the ranking problem using models like (5). However, most of them consider an off-line setting. That is, they obtain the likelihood using all available data and maximize the likelihood. Such an approach is suitable if data are not large. Recent attempts to extend this off-line approach to multiple players and multiple teams include Huang et al. (2006). However, for large systems which constantly have results being added/dropped, an online approach is more appropriate.

The Elo system is an online rating scheme which models the probability of game output as (5) with $\beta_i = \beta_q$ and, after each game, updates the strength θ_i by

$$\theta_i \leftarrow \theta_i + K(s - P(i \text{ wins})), \tag{6}$$

where K is some constant, and $s = 1$ if i wins and 0 otherwise. This formula is a very intuitive way to update strength after a game. More discussions of (6) can be seen in, for example, Glickman (1999). The Elo system with the logistic variant corresponds to the Bradley-Terry model (Bradley and Terry, 1952). The Bradley-Terry model for paired comparisons has the form

$$P(X_i > X_q) = \frac{v_i}{v_i + v_q}, \tag{7}$$

where $v_i > 0$ is the strength of team i. The model (7) dates back to Zermelo (1929) and can be derived in several ways. For instance, it can be obtained from (3) by letting X_i follow a Gumbel distribution with the cumulative distribution function

$$P(X_i \le x) = \exp(-\exp(-(x - \theta_i))), \text{ where } \theta_i = \log v_i.$$

Then $X_i - X_q$ follows a logistic distribution with the cumulative distribution function

$$P(X_i - X_q \le x) = \frac{e^{\theta_q}}{e^{\theta_i - x} + e^{\theta_q}}. \tag{8}$$

Using $x = 0$ and $P(X_i > X_q) = 1 - P(X_i \le X_q)$, we obtain (7). In fact, most currently used Elo variants for chess data use a logistic distribution rather than Gaussian because it is argued that weaker players have significantly greater winning chances than the Gaussian model predicts.[1] Figure 1 shows i's winning probability $P(X_i > X_q)$ against the skill difference $\theta_i - \theta_q$ for the two models (5) and (8). The $(\beta_i^2 + \beta_q^2)^{1/2}$ in (5) are set as $4/\sqrt{2\pi} \approx 1.6$ so that the two winning probability curves have the same slope at $\theta_i = \theta_q$. Clearly, given that the two models closely match when two teams have

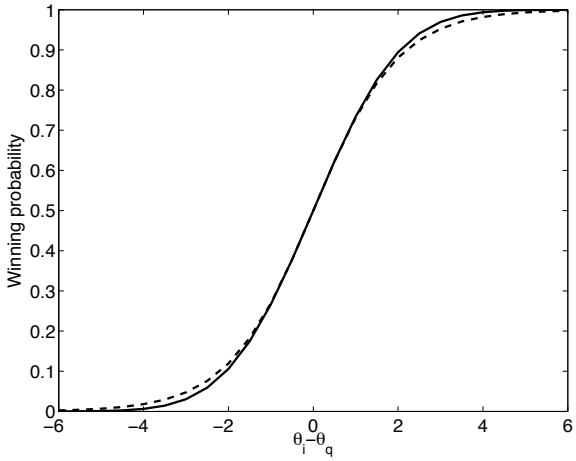


Figure 1: Winning probability $P(X_i > X_q)$. Solid (blue): Gaussian distribution (5), Dashed (black): logistic distribution (8).

about the same skill levels, the logistic model gives a weak team i a higher winning chance than the Gaussian model does.

In addition to Elo and Glicko, other online systems have been proposed. For example, Menke and Martinez (2008) propose using Artificial Neural Networks. Though this approach can handle multiple players per team, it aims to handle only two teams per game.

For comparisons involving $k \geq 3$ teams per game, the Bradley-Terry model has been generalized in various ways. The Plackett-Luce model (Marden, 1995) is one of such models. This model, motivated by a k-horse race, has the form

$$P(r(1),\ldots,r(k)) = \frac{e^{\theta_{r_1}}}{e^{\theta_{r_1}}+\cdots+e^{\theta_{r_k}}} \times \frac{e^{\theta_{r_2}}}{e^{\theta_{r_2}}+\cdots+e^{\theta_{r_k}}} \times \cdots \times \frac{e^{\theta_{r_k}}}{e^{\theta_{r_k}}}. \tag{9}$$

An intuitive explanation of this model is a multistage ranking in which one first chooses the most favorite, then chooses the second favorite out of the remaining, etc.

When $k \geq 3$, as the $X_{r(i)} - X_{r(i+1)}$'s in (4) are dependent, the calculation of the joint probability (4) involves a $(k-1)$-dimensional integration, which may be difficult to calculate. Therefore, TrueSkill uses a factor graph and the approximate message passing (Kschischang et al., 2001) to infer the marginal belief distribution over the skill of each team. In fact, some messages in the factor graph are non Gaussian and these messages are approximated via moment matching, using the Expectation Propagation algorithm (Minka, 2001).

2.2 Approximation Techniques for Bayesian Inference

From a Bayesian perspective, both the observed data and the model parameters are considered random quantities. Let D denote the observed data, and θ the unknown quantities of interest. The joint distribution of D and θ is determined by the prior distribution $P(\theta)$ and the likelihood $P(D|\theta)$:

$$P(D,\theta) = P(D|\theta)P(\theta).$$

1. According to `http://en.wikipedia.org/wiki/Elo_rating_system`, USCF and FIDE use formulas based on the logistic distribution.

After observing D, Bayes theorem gives the distribution of θ conditional on D:

$$P(\theta|D) = \frac{P(\theta,D)}{P(D)} = \frac{P(\theta,D)}{\int P(\theta,D)d\theta}.$$

This is the *posterior distribution* of θ, which is useful for estimation. Quantities about the posterior distribution such as moments, untiles, etc can be expressed in terms of posterior expectations of some functions $g(\theta)$; that is,

$$E[g(\theta)|D] = \frac{\int g(\theta)P(\theta,D)d\theta}{\int P(\theta,D)d\theta}. \tag{10}$$

The probability $P(D)$, called *evidence* or *marginal likelihood* of the data, is useful for model selection. Both $P(\theta|D)$ and $P(D)$ are major objects of Bayesian inference.

The integrations involved in Bayesian inference are usually intractable. The approximation techniques can be divided into deterministic and nondeterministic methods. The nondeterministic method refers to the Monte Carlo integration such as Markov Chain Monte Carlo (MCMC) methods, which draw samples approximately from the desired distribution and forms sample averages to estimate the expectation. However, when it comes to sequential updating with new data, the MCMC methods may not be computationally feasible, the reason being that it does not make use of the analysis from the previous data; see, for example, Section 2.8 in Glickman (1993).

The popular deterministic approaches include Laplace method, variational Bayes, expectation propagation, among others. The Laplace method is a technique for approximating integrals:

$$\int e^{nf(\mathbf{x})}d\mathbf{x} \approx \left(\frac{2\pi}{n}\right)^{\frac{k}{2}} |-\nabla^2 f(\mathbf{x}_0)|^{-\frac{1}{2}} e^{nf(\mathbf{x}_0)},$$

where $\mathbf{x}$ is k-dimensional, n is a large number, $f : R^k \to R$ is twice differentiable with a unique global maximum at $\mathbf{x}_0$, and $|\cdot|$ is the determinant of a matrix. By writing $P(\theta,D) = \exp(\log P(\theta,D))$, one can approximate the integral $\int P(\theta,D)d\theta$. This method has been applied in Bayesian statistics; for example, see Tierney and Kadane (1986) and Kass and Raftery (1995).

The variational Bayes methods are a family of techniques for approximating these intractable integrals. They construct a lower bound on the marginal likelihood and then try to optimize this bound. They also provide an approximation to the posterior distribution which is useful for estimation.

The Expectation Propagation algorithm (Minka, 2001) is an iterative approach to approximate posterior distributions. It tries to minimize Kullback-Leibler divergence between the true posterior and the approximated distribution. It can be viewed as an extension of assumed-density filtering to batch situation. The TrueSkill system (Herbrich et al., 2007) is based on this algorithm.

Now we review an identity for integrals in Lemma 1 below, which forms the basis of our approximation method. Some definitions are needed. A function $f : R^k \to R$ is called almost differentiable if there exists a function $\nabla f : R^k \to R^k$ such that

$$f(\mathbf{z}+\mathbf{y}) - f(\mathbf{z}) = \int_0^1 \mathbf{y}^T \nabla f(\mathbf{z}+t\mathbf{y})dt \tag{11}$$

for $\mathbf{z},\mathbf{y} \in R^k$. Of course, a continuously differentiable function f is almost differentiable with ∇f equal to the gradient, and (11) is the indefinite integral in multi-dimensional case.

Given $h : R^k \to R$, let $h_0 = \int h(\mathbf{z}) d\Phi_k(\mathbf{z})$ be a constant, $h_k(\mathbf{z}) = h(\mathbf{z})$,

$$h_j(z_1,\ldots,z_j) = \int_{R^{k-j}} h(z_1,\ldots,z_j,\mathbf{w}) d\Phi_{k-j}(\mathbf{w}), \text{ and} \tag{12}$$

$$g_j(z_1,\ldots,z_k) = e^{z_j^2/2} \int_{z_j}^{\infty} [h_j(z_1,\ldots,z_{j-1},w) - h_{j-1}(z_1,\ldots,z_{j-1})] e^{-w^2/2} dw, \tag{13}$$

for $-\infty < z_1,\ldots,z_k < \infty$ and $j = 1,\ldots,k$. Then let

$$Uh = [g_1,\ldots,g_k]^T \qquad \text{and} \qquad Vh = \frac{U^2 h + (U^2 h)^T}{2}, \tag{14}$$

where $U^2 h$ is the $k \times k$ matrix whose jth column is $U g_j$ and g_j is as in (13).

Let Γ be a measure of the form:

$$d\Gamma(\mathbf{z}) = f(\mathbf{z}) \phi_k(\mathbf{z}) d\mathbf{z}, \tag{15}$$

where f is a real-valued function (not necessarily non-negative) defined on R^k.

Lemma 1 *(W-Stein's Identity) Suppose that $d\Gamma$ is defined as in (15), where f is almost differentiable. Let h be a real-valued function defined on R^k. Then,*

$$\int h(\mathbf{z}) d\Gamma(\mathbf{z}) = \int f(\mathbf{z}) d\Phi_k(\mathbf{z}) \cdot \int h(\mathbf{z}) d\Phi_k(\mathbf{z}) + \int (Uh(\mathbf{z}))^T \nabla f(\mathbf{z}) d\Phi_k(\mathbf{z}), \tag{16}$$

provided all the integrals are finite.

Lemma 1 was given by Woodroofe (1989). The idea of this identity originated from Stein's lemma (Stein, 1981), but the latter considers the expectation with respect to a normal distribution (i.e., the integral $\int h(\mathbf{z}) d\Phi_k(\mathbf{z})$), while the former studies the integration with respect to a "nearly normal distribution" Γ in the sense of (15). Stein's lemma is famous and of interest because of its applications to James-Stein estimator (James and Stein, 1961) and empirical Bayes methods.

The proof of this lemma is in Proposition 1 of Woodroofe (1989). For self-completeness, we sketch it for the 1-dimensional case in Appendix A. Essentially the proof is based on exchanging the order of integration (Fibini theorem), and it is the very idea for proving Stein's lemma. Due to this reason, Woodroofe termed (16) a version of Stein's identity. However, to distinguish it from Stein's lemma, here we refer to it as W-Stein's identity.

Now we assume that $\partial f / \partial z_j,\ j = 1,\ldots,k$ are almost differentiable. Then, by writing

$$(Uh(\mathbf{z}))^T \nabla f(\mathbf{z}) = \sum_{i=1}^{k} g_i(\mathbf{z}) \frac{\partial f(\mathbf{z})}{\partial z_i}$$

and applying (16) with h and f replacing by g_i and $\partial f / \partial z_i$, we obtain

$$\int g_i \frac{\partial f}{\partial z_i} d\Phi_k(\mathbf{z}) = \Phi_k(g_i) \int \frac{\partial f}{\partial z_i} d\Phi_k(\mathbf{z}) + \int (U(g_i))^T \nabla \left(\frac{\partial f}{\partial z_i} \right) d\Phi_k(\mathbf{z}), \tag{17}$$

provided all the integrals are finite. Note that $\Phi_k(g_i)$ in the above equation is a constant defined as

$$\Phi_k(g_i) = \int g_i(\mathbf{z}) \phi_k(\mathbf{z}) d\mathbf{z}.$$

By summing up both sides of (17) over $i = 1, \ldots, k$, we can rewrite (16) as

$$\begin{aligned}\int h(\mathbf{z}) f(\mathbf{z}) d\Phi_k(\mathbf{z}) &= \int f(\mathbf{z}) d\Phi_k(\mathbf{z}) \cdot \int h(\mathbf{z}) d\Phi_k(\mathbf{z}) + (\Phi_k U h)^T \int \nabla f(\mathbf{z}) d\Phi_k(\mathbf{z}) \\ &\quad + \int \mathrm{tr}\left[(Vh(\mathbf{z}))\nabla^2 f(\mathbf{z})\right] d\Phi_k(\mathbf{z}); \end{aligned} \tag{18}$$

see Proposition 2 of Woodroofe and Coad (1997) and Lemma 1 of Weng and Woodroofe (2000). Here $\Phi_k U h = (\Phi_k(g_1), ..., \Phi_k(g_k))^T$, "tr" denotes the trace of a matrix, and $\nabla^2 f$ the Hessian matrix of f. An extension of this lemma is in Weng (2010).

Let $\mathbf{Z} = [Z_1, \ldots, Z_k]^T$ be a k-dimensional random vector with the probability density

$$C\phi_k(\mathbf{z}) f(\mathbf{z}), \tag{19}$$

where

$$C = \left(\int \phi_k(\mathbf{z}) f(\mathbf{z}) d\mathbf{z} \right)^{-1}$$

is the normalizing constant. Lemma 1 can be applied to obtain expectations of functions of $\mathbf{Z}$ in the following corollary.

Corollary 2 *Suppose that* $\mathbf{Z}$ *has probability density (19). Then,*

$$\int f d\Phi_k = C^{-1} \text{ and } Eh(\mathbf{Z}) = \int h(\mathbf{z}) d\Phi_k(\mathbf{z}) + E\left[(Uh(\mathbf{Z}))^T \frac{\nabla f(\mathbf{Z})}{f(\mathbf{Z})} \right]. \tag{20}$$

Further, (18) *and* (20) *imply*

$$Eh(\mathbf{Z}) = \int h(\mathbf{z}) d\Phi_k(\mathbf{z}) + (\Phi_k U h)^T E\left[\frac{\nabla f(\mathbf{Z})}{f(\mathbf{Z})} \right] + E\left[\mathrm{tr}\left(Vh(\mathbf{Z}) \frac{\nabla^2 f(\mathbf{Z})}{f(\mathbf{Z})} \right) \right]. \tag{21}$$

In particular, if $h(\mathbf{z}) = z_i$, then by (14) it follows $Uh(\mathbf{z}) = \mathbf{e}_i$ (a function from R^k to R^k); and if $h(\mathbf{z}) = z_i z_j$ and $i < j$, then $Uh(\mathbf{z}) = z_i \mathbf{e}_j$, where $\{\mathbf{e}_1, \cdots, \mathbf{e}_k\}$ denote the standard basis for R^k. For example, if $k = 3$ and $h(\mathbf{z}) = z_1 z_2$, then $Uh(\mathbf{z}) = [0, z_1, 0]^T$ and $U^2 h(\mathbf{z})$ is the matrix whose $(1,2)$ entry is 1 and the rest entries are zeros; see Appendix B for details. With these special h functions, (20) and (21) become

$$E[\mathbf{Z}] = E\left[\frac{\nabla f(\mathbf{Z})}{f(\mathbf{Z})} \right], \tag{22}$$

$$E[Z_i Z_q] = \delta_{iq} + E\left[\frac{\nabla^2 f(\mathbf{Z})}{f(\mathbf{Z})} \right]_{iq}, \quad i, q = 1, \ldots, k, \tag{23}$$

where $\delta_{iq} = 1$ if $i = q$ and 0 otherwise, and $[\cdot]_{iq}$ indicates the (i,q) component of a matrix.

In the current context of online ranking, since the skill θ is assumed to follow a Gaussian distribution, the update procedure is mainly for the mean and the variance. Therefore, (22) and (23) will be useful. The detailed approximation procedure is in the next section.

3. Method

In this section, we first present our proposed method for updating team and individual skills. Then, we give the detailed derivation for the Bradley-Terry model.

3.1 Approximating the Expectations

Let θ_i be the strength parameter of team i whose ability is to be estimated. Bayesian online rating systems such as Glicko and TrueSkill start by assuming that θ_i has a prior distribution $N(\mu_i, \sigma_i^2)$ with μ_i and σ_i^2 known, next model the game outcome by some probability models, and then update the skill (by either analytic or numerical approximations of the posterior mean and variance of θ_i) at the end of the game. These revised mean and variance are considered as prior information for the next game, and the updating procedure is repeated.

Equations (22) and (23) can be applied to online skill updates. To start, suppose that team i has a strength parameter θ_i and assume that the prior distribution of θ_i is $N(\mu_i, \sigma_i^2)$. Upon the completion of a game, their skills are characterized by the posterior mean and variance of $\theta = [\theta_1, \ldots, \theta_k]^T$. Let D denote the result of a game and $\mathbf{Z} = [Z_1, \ldots, Z_k]^T$ with

$$Z_i = \frac{\theta_i - \mu_i}{\sigma_i}, i = 1, \ldots, k, \tag{24}$$

where k is the number of teams. The posterior density of $\mathbf{Z}$ given the game outcome D is

$$P(\mathbf{z}|D) = C\phi_k(\mathbf{z}) f(\mathbf{z}),$$

where $f(\mathbf{z})$ is the probability of game outcome $P(D|\mathbf{z})$. Thus, $P(\mathbf{z}|D)$ is of the form (19). Subsequently we omit D in all derivations.

Next, we shall update the skill as the posterior mean and variance of θ. Equations (22), (23) and the relation between Z_i and θ_i in (24) give that

$$\begin{aligned} \mu_i^{\text{new}} =& E[\theta_i] = \mu_i + \sigma_i E[Z_i] \\ =& \mu_i + \sigma_i E\left[\frac{\partial f(\mathbf{Z})/\partial Z_i}{f(\mathbf{Z})}\right] \end{aligned} \tag{25}$$

and

$$\begin{aligned} (\sigma_i^{\text{new}})^2 =& \text{Var}[\theta_i] = \sigma_i^2 \text{Var}[Z_i] \\ =& \sigma_i^2 \left(E[Z_i^2] - E[Z_i]^2\right) \\ =& \sigma_i^2 \left(1 + E\left[\frac{\nabla^2 f(\mathbf{Z})}{f(\mathbf{Z})}\right]_{ii} - E\left[\frac{\partial f(\mathbf{Z})/\partial Z_i}{f(\mathbf{Z})}\right]^2\right). \end{aligned} \tag{26}$$

The relation between the current and the new skills are explained below. By chain rule and the definition of Z_i in (24), the second term on the right side of (25) can be written as

$$\sigma_i E\left[\frac{\partial f(\mathbf{Z})/\partial Z_i}{f(\mathbf{Z})}\right] = E\left[\frac{\partial f(\mathbf{Z})/\partial \theta_i}{f(\mathbf{Z})}\right] = E\left[\frac{\partial \log f(\mathbf{Z})}{\partial \theta_i}\right],$$

which is the average of the relative rate of change of f (the probability of game outcome) with respect to strength θ_i. For instance, suppose that team 1 beats team 2. Then, the larger θ_1 is, the more likely we have such an outcome. Hence, f is increasing in θ_1, and the adjustment to team 1's skill is the average of the relative rate of change of team 1's winning probability with respect to its strength θ_1. On the other hand, a larger θ_2 is less likely to result in this outcome; hence, f is

decreasing in θ_2 and the adjustment to team 2's skill will be negative. Similarly, we can write the last two terms on the right side of (26) as

$$\sigma_i^2 \left(E\left[\frac{\nabla^2 f(\mathbf{Z})}{f(\mathbf{Z})}\right]_{ii} - E\left[\frac{\partial f(\mathbf{Z})/\partial Z_i}{f(\mathbf{Z})}\right]^2 \right) = E\left[\frac{\partial^2 \log f(\mathbf{Z})}{\partial \theta_i^2}\right],$$

which is the average of the rate of change of $\partial(\log f)/\partial\theta_i$ with respect to θ_i.

We propose approximating expectations in (25) and (26) to obtain the update rules:

$$\mu_i \leftarrow \mu_i + \Omega_i, \tag{27}$$
$$\sigma_i^2 \leftarrow \sigma_i^2 \max(1-\Delta_i, \kappa), \tag{28}$$

where

$$\Omega_i = \sigma_i \left.\frac{\partial f(\mathbf{z})/\partial z_i}{f(\mathbf{z})}\right|_{\mathbf{z}=\mathbf{0}} \tag{29}$$

and

$$\begin{aligned}\Delta_i &= -\left.\frac{\partial^2 f(\mathbf{z})/\partial^2 z_i}{f(\mathbf{z})}\right|_{\mathbf{z}=\mathbf{0}} + \left(\left.\frac{\partial f(\mathbf{z})/\partial z_i}{f(\mathbf{z})}\right|_{\mathbf{z}=\mathbf{0}}\right)^2 \\ &= -\left.\frac{\partial}{\partial z_i}\left(\frac{\partial f(\mathbf{z})/\partial z_i}{f(\mathbf{z})}\right)\right|_{\mathbf{z}=\mathbf{0}}.\end{aligned} \tag{30}$$

We set $\mathbf{z} = \mathbf{0}$ so that θ is replaced by μ. Such a substitution is reasonable as we expect that the posterior density of θ to be concentrated on μ. Then the right-hand sides of (27)-(28) are functions of μ and σ, so we can use the current values to obtain new estimates. Due to the approximation (30), $1-\Delta_i$ may be negative. Hence in (28) we set a small positive lower bound κ to avoid a negative σ_i^2. Further, we find that the prediction results may be affected by how fast the variance σ_i^2 is reduced in (28). More discussion on this issue is in Section 3.5.

3.2 Error Analysis of the Approximation

This section discusses the error induced by evaluating the expectations in (25) and (26) at a single $\mathbf{z} = 0$, and then suggests a correction by including the prior uncertainty of skill in the variance of the actual performance. For simplicity, below we only consider a two-team game using the Thurstone-Mosteller model. Another reason of using the Thurstone-Mosteller model is that we can exactly calculate the posterior probability. To begin, suppose that the variance of *i*th team's actual performance is β_i^2. Then, for the Thurstone-Mosteller model, the joint posterior density of (θ_1, θ_2) is proportional to

$$\phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\phi\left(\frac{\theta_2-\mu_2}{\sigma_2}\right)\Phi\left(\frac{\theta_1-\theta_2}{\sqrt{\beta_1^2+\beta_2^2}}\right),$$

and the marginal posterior density of θ_1 is proportional to

$$\begin{aligned}
&\int_{-\infty}^{\infty} \phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\phi\left(\frac{\theta_2-\mu_2}{\sigma_2}\right)\Phi\left(\frac{\theta_1-\theta_2}{\sqrt{\beta_1^2+\beta_2^2}}\right)d\theta_2 \\
= \;& \phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\int_{-\infty}^{\infty}\phi\left(\frac{\theta_2-\mu_2}{\sigma_2}\right)\int_{-\infty}^{\theta_1}\frac{1}{\sqrt{2\pi}(\sqrt{\beta_1^2+\beta_2^2})}e^{-\frac{(y-\theta_2)^2}{2(\beta_1^2+\beta_2^2)}}dyd\theta_2 \\
= \;& \sigma_2\phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{\theta_1-\mu_2}{\sqrt{\beta_1^2+\beta_2^2+\sigma_2^2}}\right), \qquad (31)
\end{aligned}$$

where the last two equalities are obtained by writing the function $\Phi(\cdot)$ as an integral of ϕ (see (66)) and then interchanging the orders of the double integral. From (31), the posterior mean of θ_1 given D is

$$E(\theta_1) = \frac{\int_{-\infty}^{\infty}\theta_1\phi(\frac{\theta_1-\mu_1}{\sigma_1})\Phi(\frac{\theta_1-\mu_2}{\sqrt{\beta_1^2+\beta_2^2+\sigma_2^2}})d\theta_1}{\int_{-\infty}^{\infty}\phi(\frac{\theta_1-\mu_1}{\sigma_1})\Phi(\frac{\theta_1-\mu_2}{\sqrt{\beta_1^2+\beta_2^2+\sigma_2^2}})d\theta_1}. \qquad (32)$$

Again, by writing the function $\Phi(\cdot)$ as an integral and interchanging the orders of the integrals, we obtain that the numerator and the denominator of the right side of (32) are respectively

$$\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}}\right)\left(\mu_1+\frac{\sigma_1^2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}}\frac{\phi(\frac{\mu_1-\mu_2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}})}{\Phi(\frac{\mu_1-\mu_2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}})}\right)$$

and

$$\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}}\right).$$

Therefore, the exact posterior mean of θ_1 is

$$E(\theta_1)=\mu_1+\frac{\sigma_1^2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}}\frac{\phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}}\right)}{\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\sum_{i=1}^2(\beta_i^2+\sigma_i^2)}}\right)}. \qquad (33)$$

Now we check our estimation. According to (25), (27), and (29),

$$\begin{aligned}
E(\theta) =& \mu_1+\sigma_1 E\left[\frac{\partial f(\mathbf{Z})/\partial Z_1}{f(\mathbf{Z})}\right] \qquad (34)\\
\approx& \mu_1+\sigma_1\left.\frac{\partial f(\mathbf{z})/\partial z_1}{f(\mathbf{z})}\right|_{\mathbf{z}=\mathbf{0}}, \qquad (35)
\end{aligned}$$

where

$$f(\mathbf{z})=\Phi\left(\frac{\theta_1-\theta_2}{\sqrt{\beta_1^2+\beta_2^2}}\right) \text{ and } z_i=\frac{\theta_i-\mu_i}{\sigma_i}, i=1,2.$$

The derivation later in (93) shows that (35) leads to the following estimation for $E(\theta_1)$:

$$\mu_1 + \frac{\sigma_1^2}{\sqrt{\beta_1^2+\beta_2^2}} \frac{\phi\left(\frac{\mu_1-\mu_2}{\sqrt{\beta_1^2+\beta_2^2}}\right)}{\Phi\left(\frac{\mu_1-\mu_2}{\sqrt{\beta_1^2+\beta_2^2}}\right)}. \tag{36}$$

The only difference between (33) and (36) is that the former uses $\beta_1^2+\beta_2^2+\sigma_1^2+\sigma_2^2$, while the latter has $\beta_1^2+\beta_2^2$. Therefore, the approximation from (34) to (35) causes certain bias. We can correct the error by substituting β_i^2 with $\beta_i^2+\sigma_i^2$ when using our approximation method. In practice, we use $\beta_i^2 = \beta^2+\sigma_i^2$, where β^2 is a constant.

The above arguments also apply to the Bradley-Terry model. We leave the details in Appendix C.

3.3 Modeling Game Outcomes by Factorization

To derive update rules using (27)-(30), we must define $f(\mathbf{z})$ and then calculate Ω_i, Δ_i. Suppose that there are k teams in a game. We shall consider models for which the $f(\mathbf{z})$ in (19) can be factorized as

$$f(\mathbf{z}) = \prod_{q=1}^{m} f_q(\mathbf{z}) \tag{37}$$

for some $m > 0$. If $f_q(\mathbf{z})$ involves only several elements of $\mathbf{z}$, the above factorization may lead to an easier gradient and Hessian calculation in (22) and (23). The expectation on the right side of (22) involves the following calculation:

$$\begin{aligned} \frac{\partial f/\partial z_i}{f} &= \frac{\partial \log \prod_{q=1}^{m} f_q(\mathbf{z})}{\partial z_i} = \sum_{q=1}^{m} \frac{\partial \log f_q(\mathbf{z})}{\partial z_i} \\ &= \sum_{q=1}^{m} \frac{\partial f_q/\partial z_i}{f_q}. \end{aligned} \tag{38}$$

Then all we need is to ensure that calculating $\frac{\partial f_q/\partial z_i}{f_q}$ is feasible.

Clearly the Plackett-Luce model (9) has the form of (37). However, the Thurstone's model (3) with the Gaussian distribution can hardly be factorized into the form (37). The main reason is that the probability (4) of a game outcome involves a $(k-1)$-dimensional integration, which is intractable. One may address this problem by modeling a k-team game outcome as $(k-1)$ two-team games (between all teams on neighboring ranks); that is,

$$f(\mathbf{z}) = \prod_{i=1}^{k-1} P(\text{outcome between teams ranked } i\text{th and } (i+1)\text{st}). \tag{39}$$

Alternatively, we may consider the game result of k teams as $k(k-1)/2$ two-team games. Then

$$f(\mathbf{z}) = \prod_{i=1}^{k} \prod_{q=i+1}^{k} P(\text{outcome between team } i \text{ and team } q). \tag{40}$$

Both (39) and (40) are of the form (37). In Section 3.5, we shall demonstrate the calculation to obtain update rules. Subsequently we refer to (39) as the *partial-pair* approach, while (40) as the *full-pair* approach.

3.4 Individual Skill Update

Now, we consider the case where there are multiple players in each team. Suppose that the ith team has n_i players, the jth player in the ith team has strength θ_{ij}, and the prior distribution of θ_{ij} is $N(\mu_{ij}, \sigma_{ij}^2)$. Let θ_i denote the strength of the ith team. As in Huang et al. (2006) and Herbrich et al. (2007), we assume that a team's skill is the sum of its members' skills. Thus,

$$\theta_i = \sum_{j=1}^{n_i} \theta_{ij} \text{ for } i = 1, \ldots, k, \tag{41}$$

and the prior distribution of θ_i is

$$\theta_i \sim N(\mu_i, \sigma_i^2), \text{ where } \mu_i = \sum_{j=1}^{n_i} \mu_{ij} \text{ and } \sigma_i^2 = \sum_{j=1}^{n_i} \sigma_{ij}^2. \tag{42}$$

Similar to (27)-(28), we propose updating the skill of the jth player in team i by

$$\mu_{ij} \leftarrow \mu_{ij} + \frac{\sigma_{ij}^2}{\sigma_i^2}\Omega_i, \tag{43}$$

$$\sigma_{ij}^2 \leftarrow \sigma_{ij}^2 \max\left(1 - \frac{\sigma_{ij}^2}{\sigma_i^2}\Delta_i, \kappa\right), \tag{44}$$

where Ω_i and Δ_i are defined in (29) and (30), respectively and κ is a small positive value to ensure a positive σ_{ij}^2. Equations (43) and (44) say that Ω_i, the mean skill change of team i, is partitioned to n_i parts with the magnitude proportional to σ_{ij}^2. These rules can be obtained from the following derivation. Let Z_{ij} be the normalized quantity of the random variable θ_{ij}; that is,

$$Z_{ij} = (\theta_{ij} - \mu_{ij})/\sigma_{ij}. \tag{45}$$

As in (27), we could update μ_{ij} by

$$\mu_{ij} \leftarrow \mu_{ij} + \sigma_{ij} \left.\frac{\partial f(\mathbf{z})/\partial z_{ij}}{f}\right|_{\mathbf{z}=\mathbf{0}}, \tag{46}$$

where $f(\mathbf{z})$ is the probability of game outcomes and

$$\mathbf{z} = [z_{11}, \ldots, z_{1n_1}, \ldots, z_{k1}, \ldots, z_{kn_k}]^T.$$

Since we assume a team's strength is the sum of its members', from (24), (41), (42), and (45) we have

$$Z_i = \frac{\theta_i - \mu_i}{\sigma_i} = \frac{\sum_j \sigma_{ij} Z_{ij}}{\sigma_i}; \tag{47}$$

hence, it is easily seen that $f(\mathbf{z})$ is simply a reparametrization of $f(\mathbf{z})$ (defined in Section 3.1):

$$f(\mathbf{z}) = f\left(\sum_{j=1}^{n_1} \frac{\sigma_{1j} z_{1j}}{\sigma_1}, \ldots, \sum_{j=1}^{n_k} \frac{\sigma_{kj} z_{kj}}{\sigma_k}\right) = f(\mathbf{z})$$

With (47),

$$\frac{\partial f(\mathbf{z})}{\partial z_{ij}} = \frac{\partial f(\mathbf{z})}{\partial z_i} \cdot \frac{\partial z_i}{\partial z_{ij}} = \frac{\sigma_{ij}}{\sigma_i} \frac{\partial f(\mathbf{z})}{\partial z_i}$$

and (46) becomes

$$\mu_{ij} \leftarrow \mu_{ij} + \frac{\sigma_{ij}^2}{\sigma_i^2} \cdot \sigma_i \left. \frac{\partial f(\mathbf{z})/\partial z_i}{f} \right|_{\mathbf{z}=\mathbf{0}}.$$

Following the definition of Ω_i in (29) we obtain the update rule (43), which says that within team i the adjustment to μ_{ij} is proportional to σ_{ij}^2. The update rule (44) for the individual variance can be derived similarly.

3.5 Example: Bradley-Terry Model (Full-pair)

In this section, we consider the Bradley-Terry model and derive the update rules using the full-pair setting in (40). Following the discussion in Equations. (7)-(8), the difference $X_i - X_q$ between two teams follows a logistic distribution. However, by comparing the Thurstone-Mosteller model (5) and the Bradley-Terry model (7), clearly the Bradley-Terry model lacks variance parameters β_i^2 and β_q^2, which account for the performance uncertainty. We thus extend the Bradley-Terry model to include variance parameters; see Appendix C. The resulting model is

$$P(\text{team } i \text{ beats } q) \equiv f_{iq}(\mathbf{z}) = \frac{e^{\theta_i/c_{iq}}}{e^{\theta_i/c_{iq}} + e^{\theta_q/c_{iq}}}, \tag{48}$$

where

$$c_{iq}^2 = \beta_i^2 + \beta_q^2 \text{ and } \theta_i = \mu_i + \sigma_i z_i.$$

The parameter β_i is the uncertainty about the actual performance X_i. However, in the model specification, the uncertainty of X_i is not related to σ_i. Following the error analysis of the approximation in Section 3.2 for the Thurstone-Mosteller model, we show in Appendix C that σ_i^2 can be incorporated to

$$\beta_i^2 = \sigma_i^2 + \beta^2,$$

where β^2 is some positive constant.

There are several extensions to the Bradley-Terry model incorporating ties. In Glicko (Glickman, 1999), a tie is treated as a half way between a win and a loss when constructing the likelihood function. That is,

$$\begin{aligned} P(i \text{ draws with } q) &= (P(i \text{ beats } q) P(q \text{ beats } i))^{1/2} \\ &= \sqrt{f_{iq}(\mathbf{z}) f_{qi}(\mathbf{z})}. \end{aligned} \tag{51}$$

By considering all pairs, the resulting $f(\mathbf{z})$ is (40). To obtain update rules (27)-(28), we need to calculate $\partial f/\partial z_i$. We see that terms related to z_i in the product form of (40) are

$$P(\text{outcome of } i \text{ and } q), \forall q = 1, \ldots, k, q \neq i. \tag{52}$$

With (38) and (51),

$$\begin{aligned} &\frac{\partial f/\partial z_i}{f} \\ &= \sum_{q:r(q)<r(i)} \frac{\partial f_{qi}/\partial z_i}{f_{qi}} + \sum_{q:r(q)>r(i)} \frac{\partial f_{iq}/\partial z_i}{f_{iq}} + \frac{1}{2} \sum_{q:r(q)=r(i), q\neq i} \left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}} + \frac{\partial f_{iq}/\partial z_i}{f_{iq}} \right). \end{aligned} \tag{53}$$

Algorithm 1 Update rules using the Bradley-Terry model with full-pair

1. Given a game result and the current $\mu_{ij}, \sigma_{ij}^2, \forall i, \forall j$. Given β^2 and $\kappa > 0$. Decide a way to set γ_q in (50)

2. For $i = 1, \ldots, k$, set
$$\mu_i = \sum_{j=1}^{n_i} \mu_{ij}, \quad \sigma_i^2 = \sum_{j=1}^{n_i} \sigma_{ij}^2.$$

3. For $i = 1, \ldots, k$,

   3.1. Team skill update: obtain Ω_i and Δ_i in (27) and (28) by the following steps.

      3.1.1. For $q = 1, \ldots, k, q \neq i$,
$$c_{iq} = (\sigma_i^2 + \sigma_q^2 + 2\beta^2)^{1/2}, \qquad \hat{p}_{iq} = \frac{e^{\mu_i/c_{iq}}}{e^{\mu_i/c_{iq}} + e^{\mu_q/c_{iq}}}, \tag{49}$$
$$\delta_q = \frac{\sigma_i^2}{c_{iq}}(s - \hat{p}_{iq}), \quad \eta_q = \gamma_q \big(\frac{\sigma_i}{c_{iq}}\big)^2 \hat{p}_{iq}\hat{p}_{qi}, \text{ where } s = \begin{cases} 1 & \text{if } r(q) > r(i), \\ 1/2 & \text{if } r(q) = r(i), \\ 0 & \text{if } r(q) < r(i). \end{cases} \tag{50}$$

      3.1.2. Calculate
$$\Omega_i = \sum_{q:q\neq i} \delta_q, \qquad \Delta_i = \sum_{q:q\neq i} \eta_q.$$

   3.2. Individual skill update

   For $j = 1, \ldots, n_i$,
$$\mu_{ij} \leftarrow \mu_{ij} + \frac{\sigma_{ij}^2}{\sigma_i^2}\Omega_i, \qquad \sigma_{ij}^2 \leftarrow \sigma_{ij}^2 \max\left(1 - \frac{\sigma_{ij}^2}{\sigma_i^2}\Delta_i, \kappa\right).$$

Using (24) and (48), it is easy to calculate that
$$\frac{\partial f_{qi}}{\partial z_i} = \frac{-e^{\theta_i/c_{iq}} e^{\theta_q/c_{iq}}}{c_{iq}(e^{\theta_i/c_{iq}} + e^{\theta_q/c_{iq}})^2} \cdot \frac{\partial \theta_i}{\partial z_i} = \frac{-\sigma_i}{c_{iq}} f_{iq} f_{qi} \tag{54}$$
and
$$\frac{\partial f_{iq}}{\partial z_i} = \frac{(e^{\theta_i/c_{iq}} + e^{\theta_q/c_{iq}})e^{\theta_i/c_{iq}} - e^{\theta_i/c_{iq}} e^{\theta_i/c_{iq}}}{c_{iq}(e^{\theta_i/c_{iq}} + e^{\theta_q/c_{iq}})^2} \cdot \sigma_i = \frac{\sigma_i}{c_{iq}} f_{iq} f_{qi}.$$
Therefore, an update rule following (27) and (29) is
$$\mu_i \leftarrow \mu_i + \Omega_i, \tag{55}$$
where
$$\Omega_i = \sigma_i^2 \left(\sum_{q:r(q)<r(i)} \frac{-\hat{p}_{iq}}{c_{iq}} + \sum_{q:r(q)>r(i)} \frac{\hat{p}_{qi}}{c_{iq}} + \frac{1}{2} \sum_{q:r(q)=r(i), q\neq i} \left(\frac{-\hat{p}_{iq}}{c_{iq}} + \frac{\hat{p}_{qi}}{c_{iq}} \right) \right) \tag{56}$$

and

$$\hat{p}_{iq} \equiv \frac{e^{\mu_i/c_{iq}}}{e^{\mu_i/c_{iq}} + e^{\mu_q/c_{iq}}} \tag{57}$$

is an estimate of $P(\text{team } i \text{ beats team } q)$. Since $\hat{p}_{iq} + \hat{p}_{qi} = 1$, (56) can be rewritten as

$$\Omega_i = \sum_{q:q\neq i} \frac{\sigma_i^2}{c_{iq}}(s - \hat{p}_{iq}), \qquad \text{where } s = \begin{cases} 1 & \text{if } r(q) > r(i), \\ \frac{1}{2} & \text{if } r(q) = r(i), \\ 0 & \text{if } r(q) < r(i). \end{cases} \tag{58}$$

To apply (26) and (30) for updating σ_i, we use (53) to obtain

$$\begin{aligned} \frac{\partial}{\partial z_i}\left(\frac{\partial f/\partial z_i}{f}\right) = &\sum_{q:r(q)<r(i)} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}}\right) + \sum_{q:r(q)>r(i)} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right) \\ &+ \frac{1}{2}\sum_{q:r(q)=r(i),q\neq i}\left(\frac{\partial}{\partial z_i}\left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}}\right) + \frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right)\right). \end{aligned} \tag{59}$$

From (54),

$$\frac{\partial}{\partial z_i}\left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}}\right) = \frac{\partial(-f_{iq}/c_{iq})}{\partial z_i} = -\frac{\sigma_i^2}{c_{iq}^2} f_{iq} f_{qi}$$

and similarly

$$\frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right) = -\frac{\sigma_i^2}{c_{iq}^2} f_{iq} f_{qi}. \tag{60}$$

From (30), by setting $\mathbf{z} = \mathbf{0}$, Δ_i should be the sum of (60) over all $q \neq i$. However, we mentioned in the end of Section 3.1 that controlling the reduction of σ_i^2 is sometimes important. In particular, σ_i^2 should not be reduced too fast. Hence we introduce an additional parameter γ_q so that the update rule is

$$\sigma_i^2 \leftarrow \sigma_i^2 \max\left(1 - \sum_{q:q\neq i} \gamma_q \xi_q, \kappa\right),$$

where

$$\xi_q = \frac{\sigma_i^2}{c_{iq}^2} \hat{p}_{iq} \hat{p}_{qi}$$

is from (60) and $\gamma_q \leq 1$ is decided by users; further discussions on the choice of γ_q are in Section 6. Algorithm 1 summarizes the procedure.

The formulas (55) and (58) resemble the Elo system. The Elo treats θ_i as nonrandom and its update rule is in (6):

$$\theta_i \leftarrow \theta_i + K(s - p_{iq}^*),$$

where K is a constant (e.g., $K = 32$ in the USCF system for amateur players) and

$$p_{iq}^* = \frac{10^{\theta_i/400}}{10^{\theta_i/400} + 10^{\theta_q/400}}$$

is the approximate probability that i beats q; see Equations. (11) and (12) in Glickman (1999). Observe that p_{iq}^* is simply a variance free and reparameterized version of $\hat{p}_{iq}$ in (57). As for Glicko, it is a Bayesian system but designed for paired comparisons over a rating period. Detailed comparisons with Glicko are in Section 5.

Algorithm 2 Update rules using the Bradley-Terry model with partial-pair

The procedure is the same as Algorithm 1 except Step 3:

3. Let $r(a), a = 1,\ldots,k$ be indices of teams ranked from the first to the last

   For $a = 1,\ldots,k$,

   3.1. Team skill update: let $i \equiv r(a)$ and obtain Ω_i and Δ_i in (27) and (28) by the following steps.

   3.1.1. Define a set Q as

$$Q \equiv \begin{cases} \{r(a+1)\} & \text{if } a = 1, \\ \{r(a-1)\} & \text{if } a = k, \\ \{r(a-1), r(a+1)\} & \text{otherwise.} \end{cases} \tag{61}$$

   For $q \in Q$

   Calculate δ_q, η_q by the same way as (49)-(50) of Algorithm 1.

   3.1.2. Calculate

$$\Omega_i = \sum_{q \in Q} \delta_q \quad \text{and} \quad \Delta_i = \sum_{q \in Q} \eta_q. \tag{62}$$

   3.2 Individual skill update: same as Algorithm 1.

4. Update Rules Using Other Ranking Models

If we assume different distributions of the team performance X_i or model the game results by other ways than the Bradley-Terry model, the same framework in Sections 3.1-3.3 can still be applied. In this section, we present several variants of our proposed method.

4.1 Bradley-Terry Model (Partial-pair)

We now consider the partial-pair approach in (39). With the definition of r in (2), the function $f(\mathbf{z})$ can be written as

$$f(\mathbf{z}) = \prod_{a=1}^{k-1} f_{r(a)r(a+1)}(\mathbf{z}), \tag{63}$$

where we define $f_{r(a)r(a+1)}(\mathbf{z})$ as follows:

$$i \equiv r(a), \quad q \equiv r(a+1),$$
$$f_{iq} = \begin{cases} f_{iq} & \text{if } r(i) < r(q), \\ \sqrt{f_{iq} f_{qi}} & \text{if } r(i) = r(q). \end{cases} \tag{64}$$

Note that f_{iq} and f_{qi} are defined in (48) of Section 3.5. Since the definition of r in (2) ensures $r(i) \le r(q)$, in (64) we do not need to handle the case of $r(i) > r(q)$. By a derivation similar to that in Section 3.5, we obtain update rules in Algorithm 2. Clearly, Algorithm 2 differs from Algorithm 1 in only Step 3. The reason is that $\partial f(\mathbf{z})/\partial z_i$ is only related to game outcomes between $r(a)$ and teams of adjacent ranks, $r(a-1)$ and $r(a+1)$. In (61), we let Q be the set of these teams. Thus,

Q contains at most two elements, and Ω_i and Δ_i in (62) are calculated using δ_q and η_q with $q \in Q$. Details of the derivation are in Appendix D.

4.2 Thurstone-Mosteller Model (Full-pair and Partial-pair)

In this section, we consider the Thurstone-Mosteller model by assuming that the actual performance of team i is

$$X_i \sim N(\theta_i, \beta_i^2),$$

where $\beta_i^2 = \sigma_i^2 + \beta^2$ as in Section 3.5. The performance difference $X_i - X_q$ follows a normal distribution $N(\theta_i - \theta_q, c_{iq}^2)$ with $c_{iq}^2 = \sigma_i^2 + \sigma_q^2 + 2\beta^2$. If one considers partial pairs

$$P(\text{team } i \text{ beats team } q) = P(X_i > X_q) = \Phi\left(\frac{\theta_i - \theta_q}{c_{iq}}\right)$$

and uses (51) to obtain $P(i \text{ draws with } q)$, then a derivation similar to that for the Bradley-Terry model leads to certain update rules. Instead, here we follow Herbrich et al. (2007) to let ε be the draw margin that depends on the game mode and assume that the probabilities that i beats q and a draw occurs are respectively

$$P(\text{team } i \text{ beats team } q) = P(X_i > X_q + \varepsilon) = \Phi\left(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}}\right)$$

and

$$\begin{aligned} &P(\text{team } i \text{ draws with } q) = P(|X_i - X_q| < \varepsilon) \\ =&\Phi\left(\frac{\varepsilon - (\theta_i - \theta_q)}{c_{iq}}\right) - \Phi\left(\frac{-\varepsilon - (\theta_i - \theta_q)}{c_{iq}}\right). \end{aligned} \tag{65}$$

We can then obtain $f(\mathbf{z})$ using the full-pair setting (40). The way to derive update rules is similar to that for the Bradley-Terry model though some details are different. We summarize the procedure in Algorithm 3. Detailed derivations are in Appendix E.

Interestingly, if $k = 2$ (i.e., two teams), then the update rules (if i beats q) in Algorithm 3 are reduced to

$$\begin{aligned} \mu_i \leftarrow& \mu_i + \frac{\sigma_i^2}{c_{iq}} V\left(\frac{\mu_i - \mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right), \\ \mu_q \leftarrow& \mu_q - \frac{\sigma_q^2}{c_{iq}} V\left(\frac{\mu_i - \mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right), \end{aligned}$$

where the function V is defined in (67). These update rules are the same as the case of $k = 2$ in the TrueSkill system (see `http://research.microsoft.com/en-us/projects/trueskill/details.aspx`).

As a comparison, we note that TrueSkill considers partial-pair and obtains players' skills by a factor graph and the approximate message passing. In fact, some messages in the factor graph are non Gaussian and these messages are approximated via moment matching, using the *Expectation Propagation* algorithm (Minka, 2001). Their algorithm is effective, but simple update rules are not available for the cases of multiple teams/players.

Algorithm 3 Update rules using Thurstone-Mosteller model with full-pair

The procedure is the same as Algorithm 1 except Step 3.1.1:

3.1.1 For $q = 1,\ldots,k; q \neq i$,

$$\delta_q = \frac{\sigma_i^2}{c_{iq}} \times \begin{cases} V(\frac{\mu_i-\mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) & \text{if } r(q) > r(i), \\ \tilde{V}(\frac{\mu_i-\mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) & \text{if } r(q) = r(i), \\ -V(\frac{\mu_q-\mu_i}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) & \text{if } r(q) < r(i), \end{cases}$$

$$\eta_q = (\frac{\sigma_i}{c_{iq}})^2 \times \begin{cases} W(\frac{\mu_i-\mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) & \text{if } r(q) > r(i), \\ \tilde{W}(\frac{\mu_i-\mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) & \text{if } r(q) = r(i), \\ W(\frac{\mu_q-\mu_i}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) & \text{if } r(q) < r(i), \end{cases}$$

where

$$c_{iq} = (\sigma_i^2 + \sigma_q^2 + 2\beta^2)^{1/2},$$

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \quad \Phi(x) = \int_{-\infty}^{x} \phi(u)du, \tag{66}$$

$$V(x,t) = \phi(x-t)/\Phi(x-t), \quad W(x,t) = V(x,t)(V(x,t) + (x-t)), \tag{67}$$

$$\tilde{V}(x,t) = -\frac{\phi(t-x) - \phi(-t-x)}{\Phi(t-x) - \Phi(-t-x)}, \tag{68}$$

$$\tilde{W}(x,t) = \frac{(t-x)\phi(t-x) - (-(t+x))\phi(-(t+x))}{\Phi(t-x) - \Phi(-t-x)} + \tilde{V}(x,t)^2. \tag{69}$$

4.3 Plackett-Luce Model

We now discuss the situation of using the Plackett-Luce model. If ties are not allowed, an extension of the Plackett-Luce model (9) incorporating variance parameters is

$$f(\mathbf{z}) = \prod_{q=1}^{k} f_q(\mathbf{z}) = \prod_{q=1}^{k} \left(\frac{e^{\theta_q/c}}{\sum_{s \in C_q} e^{\theta_s/c}} \right), \tag{70}$$

where

$$z_i = \frac{\theta_i - \mu_i}{\sigma_i}, c = \left(\sum_{i=1}^{k} (\sigma_i^2 + \beta^2) \right)^{1/2} \text{ and } C_q = \{i : r(i) \geq r(q)\}.$$

Instead of the same c in $e^{\theta_q/c}$, similar to the Bradley-Terry model, we can define c_q to sum up $\sigma_i^2, i \in C_q$. However, here we take the simpler setting of using the same c. Note that $f_q(\mathbf{z})$ corresponds to the probability that team q is the winner among teams in C_q. In (9), $f(\mathbf{z})$ is represented using $r(1),\ldots,r(k)$, but (70) is a reformulation using $r(1),\ldots,r(k)$.

We extend this model to allow ties. If teams $i_1,\ldots,i_d$ are tied together, then $r(i_1)=\cdots=r(i_d)$. A generalization of the tie probability (51) gives the likelihood based on these d stages as:

$$\left(\frac{e^{\theta_{i_1}/c}}{\sum_{s:r(s)\geq r(i_1)}e^{\theta_s/c}}\times\cdots\times\frac{e^{\theta_{i_d}/c}}{\sum_{s:r(s)\geq r(i_d)}e^{\theta_s/c}}\right)^{1/d}. \tag{71}$$

We can explain (71) as follows. Now d factors in (71) all correspond to the likelihood of the same rank, so we multiply them and take the dth root. The new $f(\mathbf{z})$ becomes

$$f(\mathbf{z})=\prod_{q=1}^{k}f_q(\mathbf{z})=\prod_{q=1}^{k}\left(\frac{e^{\theta_q/c}}{\sum_{s\in C_q}e^{\theta_s/c}}\right)^{1/A_q}, \tag{72}$$

where

$$A_q=|\{s:r(s)=r(q)\}| \text{ and } f_q(\mathbf{z})=\left(\frac{e^{\theta_q/c}}{\sum_{s\in C_q}e^{\theta_s/c}}\right)^{1/A_q},\quad q=1,\ldots,k.$$

If ties do not occur, $A_q=1$, so (72) goes back to (70). By calculations shown in Appendix F, the update rules are in Algorithm 4.

5. Description of Glicko

Since our Algorithm 1 and the Glicko system are both based on the Bradley-Terry model, it is of interest to compare these two algorithms. We describe the derivation of Glicko in this section. Note that notation in this section may be slightly different from other sections of this paper.

Consider a rating period of paired comparisons. Assume that prior to a rating period the distribution of a player's strength θ is $N(\mu,\sigma^2)$, with μ and σ^2 known. Assume that, during the rating period, the player plays n_j games against opponent j, where $j=1,\ldots,m$, and that the jth opponent's strength θ_j follows $N(\mu_j,\sigma_j^2)$, with μ_j and σ_j^2 known. Let s_{jk} be the outcome of the kth game against opponent j, with $s_{jk}=1$ if the player wins, $s_{jk}=0.5$ if the game results in a tie, and $s_{jk}=0$ if the player loses. Let D be the collection of game results during this period. The interest lies in the marginal posterior distribution of θ given D:

$$P(\theta|D) \;=\; \int\cdots\int P(\theta_1,\ldots,\theta_m|D)P(\theta|\theta_1,\ldots,\theta_m,D)d\theta_1\cdots d\theta_m, \tag{75}$$

where $P(\theta|\theta_1,\ldots,\theta_m,D)$ is the posterior distribution of θ conditional on opponents' strengths,

$$P(\theta|\theta_1,\ldots,\theta_m,D) \;\propto\; \phi(\theta|\mu,\sigma^2)P(D|\theta,\theta_1,\ldots,\theta_m). \tag{76}$$

Here $P(D|\theta,\theta_1,\ldots,\theta_m)$ is the likelihood for all parameters. The approximation procedure is described in steps (I)-(V) below, where step (I) is from Section 3.3 of Glickman (1999) and steps (II)-(IV) are summarized from his Appendix A.

(I) Glickman (1999) stated that "The key idea is that the marginal posterior distribution of a player's strength is determined by integrating out the opponents' strengths over their *prior* distribution rather than over their posterior distribution." That is, the posterior distribution of opponents' strengths $P(\theta_1,\ldots,\theta_m|D)$ is approximated by the prior distribution

$$\phi(\theta_1|\mu_1,\sigma_1^2)\cdots\phi(\theta_m|\mu_m,\sigma_m^2).$$

Algorithm 4 Update rules using the Plackett-Luce model

The procedure is the same as Algorithm 1 except Step 3:

3. Find and store

$$c = \left(\sum_{i=1}^{k} (\sigma_i^2 + \beta^2) \right)^{1/2},$$
$$A_q = |\{s : r(s) = r(q)\}|, \quad q = 1, \ldots, k$$
$$\sum_{s \in C_q} e^{\theta_s / c}, q = 1, \ldots, k, \quad \text{where } C_q = \{i : r(i) \geq r(q)\}.$$

For $i = 1, \ldots, k$,

3.1. Team skill update: obtain Ω_i and Δ_i in (27) and (28) by the following steps.

3.1.1. For $q = 1, \ldots, k$,

$$\delta_q = \frac{\sigma_i^2}{c A_q} \times \begin{cases} 1 - \hat{p}_{i,C_q} & \text{if } q = i, \\ -\hat{p}_{i,C_q} & \text{if } r(q) \leq r(i), q \neq i, \\ 0 & \text{if } r(q) > r(i), \end{cases}$$

$$\eta_q = \frac{\gamma_q \sigma_i^2}{c^2 A_q} \times \begin{cases} \hat{p}_{i,C_q}(1 - \hat{p}_{i,C_q}) & \text{if } r(q) \leq r(i), \\ 0 & \text{if } r(q) > r(i), \end{cases}$$

where

$$\hat{p}_{i,C_q} = \frac{e^{\theta_i / c}}{\sum_{s \in C_q} e^{\theta_s / c}}.$$

3.1.2 Same as Algorithm 1.

3.2 Same as Algorithm 1.

Then, together with (75) and (76) it follows that, approximately

$$\begin{aligned} P(\theta|D) &\propto \phi(\theta|\mu, \sigma^2) \int \cdots \int \phi(\theta_1|\mu_1, \sigma_1^2) \cdots \phi(\theta_m|\mu_m, \sigma_m^2) P(D|\theta, \theta_1, \ldots, \theta_m) d\theta_1 \cdots d\theta_m \\ &\propto \phi(\theta|\mu, \sigma^2) \underbrace{\prod_{j=1}^{m} \left\{ \int \left[\prod_{k=1}^{n_j} \left(\frac{(10^{(\theta - \theta_j)/400})^{s_{jk}}}{1 + 10^{(\theta - \theta_j)/400}} \right) \phi(\theta_j|\mu_j, \sigma_j^2) \right] d\theta_j \right\}}_{P(D|\theta)}, \end{aligned} \tag{77}$$

where the last line follows by treating terms in the likelihood that do not depend on θ (which correspond to games played between other players) as constant. We denote a term in (77) as $P(D|\theta)$ for subsequent analysis.

(II) $P(D|\theta)$ in (77) is the likelihood integrated over the opponents' prior strength distribution. Then, (77) becomes

$$P(\theta|D) \propto \phi(\theta|\mu, \sigma^2) P(D|\theta). \tag{78}$$

Algorithm 5 Update rules of Glicko with a single game

1. Given a game result and the current $\mu_1, \mu_2, \sigma_1^2, \sigma_2^2$. Set

$$q = \frac{\log 10}{400}. \tag{73}$$

2. For $i = 1, 2$

$$g(\sigma_i^2) = \frac{1}{\sqrt{1 + \frac{3q^2\sigma_i^2}{\pi^2}}}. \tag{74}$$

3. For $i = 1, 2$, set $j \neq i$ and

$$p_j^* = \frac{1}{1 + 10^{-g(\sigma_j^2)(\mu_i - \mu_j)/400}}, \qquad (\delta_i^2)^* = \left[q^2 (g(\sigma_j^2))^2 p_j^* (1 - p_j^*)\right]^{-1}.$$

4. Update rule: For $i = 1, 2$, set $j \neq i$

$$\begin{aligned} \mu_i &\leftarrow \mu_i + \frac{q}{\frac{1}{\sigma_i^2} + \frac{1}{(\delta_i^2)^*}} g(\sigma_j^2)(s_{ij} - p_j^*), \text{ where } s_{ij} = \begin{cases} 1 & \text{if } i \text{ wins,} \\ 1/2 & \text{if draw,} \\ 0 & \text{if } i \text{ loses,} \end{cases} \\ \sigma_i^2 &\leftarrow \left(\frac{1}{\sigma_i^2} + \frac{1}{(\delta_i^2)^*} \right)^{-1}. \end{aligned}$$

In this step, $P(D|\theta)$ is approximated by a product of logistic cumulative distribution functions:

$$P(D|\theta) \approx \prod_{j=1}^{m} \prod_{k=1}^{n_j} \int \frac{(10^{(\theta - \theta_j)/400})^{s_{jk}}}{1 + 10^{(\theta - \theta_j)/400}} \phi(\theta_j | \mu_j, \sigma_j^2) d\theta_j. \tag{79}$$

(III) In this step, $P(D|\theta)$ is further approximated by a normal distribution. First, one approximates each logistic cdf in the integrand of (79) by a normal cdf with the same mean and variance so that the integral can be evaluated in a closed form to a normal cdf. This yields the approximation

$$\int \frac{(10^{(\theta - \theta_j)/400})^{s_{jk}}}{1 + 10^{(\theta - \theta_j)/400}} \phi(\theta_j | \mu_j, \sigma_j^2) d\theta_j \approx \frac{\left(10^{g(\sigma_j^2)(\theta - \mu_j)/400}\right)^{s_{jk}}}{1 + 10^{g(\sigma_j^2)(\theta - \mu_j)/400}},$$

where $g(\sigma_j^2)$ is defined in (74). Therefore, the (approximate) marginal likelihood in (79) is

$$P(D|\theta) \approx \prod_{j=1}^{m} \prod_{k=1}^{n_j} \frac{\left(10^{g(\sigma_j^2)(\theta - \mu_j)/400}\right)^{s_{jk}}}{1 + 10^{g(\sigma_j^2)(\theta - \mu_j)/400}}. \tag{80}$$

Second, by central limit theorem we approximate this marginal likelihood (80) by a normal density $\phi(\theta|\hat{\theta}, \delta^2)$, where $\hat{\theta}$ is the mode of this marginal likelihood and δ^2 is minus of inverse of Hessian

of the log marginal likelihood evaluated at $\hat{\theta}$. Then, together with (78) we obtain an approximation:

$$\begin{aligned} P(\theta|D) &\propto \phi(\theta|\mu,\sigma^2)\phi(\theta|\hat{\theta},\delta^2) \\ &\propto \phi\left(\theta \mid \frac{\frac{\mu}{\sigma^2}+\frac{\hat{\theta}}{\delta^2}}{\frac{1}{\sigma^2}+\frac{1}{\delta^2}}, \left(\frac{1}{\sigma^2}+\frac{1}{\delta^2}\right)^{-1}\right). \end{aligned}$$

Therefore, the update of μ and σ^2 (i.e., posterior mean and variance) is:

$$\sigma^2 \leftarrow \left(\frac{1}{\sigma^2}+\frac{1}{\delta^2}\right)^{-1} \quad \text{and} \quad \mu \leftarrow \frac{\frac{\mu}{\sigma^2}+\frac{\hat{\theta}}{\delta^2}}{\frac{1}{\sigma^2}+\frac{1}{\delta^2}} = \mu + \frac{\frac{1}{\delta^2}}{\frac{1}{\sigma^2}+\frac{1}{\delta^2}}(\hat{\theta}-\mu). \tag{81}$$

Note that we obtain $\hat{\theta}$ by equating the derivative of $\log P(D|\theta)$ to zero, and approximating δ^2 by substituting μ for $\hat{\theta}$. The expression of approximation for δ^2 is

$$\delta^2 \approx \left(q^2 \sum_{j=1}^{m} n_j (g(\sigma_j^2))^2 p_j(\mu)(1-p_j(\mu))\right)^{-1}, \tag{82}$$

where q is defined in (73), $g(\sigma_j^2)$ is defined in (74) and

$$p_j(\mu) = \frac{1}{1+10^{-g(\sigma_j^2)(\mu-\mu_j)/400}}, \tag{83}$$

which is an approximate probability that the player beats opponent j.

(IV) Finally, $\hat{\theta}-\mu$ in (81) is approximated as follows. From (80) it follows that

$$\frac{d}{d\theta}\log P(D|\theta) \approx \sum_{j=1}^{m}\sum_{k=1}^{n_j} \frac{\log 10}{400}\left\{g(\sigma_j^2)\left(s_{jk} - \frac{1}{1+10^{-g(\sigma_j^2)(\theta-\mu_j)/400}}\right)\right\}. \tag{84}$$

If we define

$$h(\theta) = \sum_{j=1}^{m}\sum_{k=1}^{n_j} \frac{g(\sigma_j^2)}{1+10^{-g(\sigma_j^2)(\theta-\mu_j)/400}}, \tag{85}$$

then setting the right-hand side of (84) to zero gives

$$h(\hat{\theta}) = \sum_{j=1}^{m}\sum_{k=1}^{n_j} g(\sigma_j^2) s_{jk}. \tag{86}$$

Then, a Taylor series expansion of $h(\theta)$ around μ gives

$$h(\hat{\theta}) \approx h(\mu) + (\hat{\theta}-\mu)h'(\mu), \tag{87}$$

where

$$h'(\mu) = q\sum_{j=1}^{m}\sum_{k=1}^{n_j}(g(\sigma_j^2))^2 p_j(\mu))(1-p_j(\mu)) = q\sum_{j=1}^{m} n_j (g(\sigma_j^2))^2 p_j(\mu))(1-p_j(\mu)) \tag{88}$$

Game type	# games	# players	BT-full	BT-partial	PL	TM-full	TrueSkill
Free for All	5,943	60,022	**30.59%**	32.40%	31.74%	44.65%	30.82%
Small Teams	27,539	4,992	**33.97%**	**33.97%**	**33.97%**	36.46%	35.23%
Head to Head	6,227	1,672	32.53%	32.53%	32.53%	**32.41%**	32.44%
Large Teams	1,199	2,576	**37.30%**	**37.30%**	**37.30%**	39.37%	38.15%

Table 2: Data description and prediction errors by various methods. The method with the smallest error is bold-faced. The column "TrueSkill" is copied from a table in Herbrich et al. (2007). Note that we use the same way as TrueSkill to calculate prediction errors.

Game type	BT-full	PL	TM-full
Free for All	31.24%	31.73%	33.13%
Small Teams	33.84%	33.84%	36.50%
Head to Head	32.55%	32.55%	32.74%
Large Teams	37.30%	37.30%	39.13%

Table 3: Prediction errors using $\gamma_q = 1/k$ in (50), where k is the number of teams in a game.

with $p_j(\mu)$ defined in (83). Using (86), $h(\mu)$ by (85), and (88), we can apply (87) to obtain an estimate of $\hat{\theta} - \mu$. Then with (82), (81) becomes

$$\mu \leftarrow \mu + \frac{q}{\frac{1}{\sigma^2} + \frac{1}{\delta^2}} \sum_{j=1}^{m} \sum_{k=1}^{n_j} g(\sigma_j^2)(s_{jk} - p_j(\mu)).$$

However, when there is only one game, $P(D|\theta)$ in (80) would have just one term (because $m = 1$ and $n_1 = 1$), and it is a monotone function. Therefore, the mode $\hat{\theta}$ of $P(D|\theta)$ would be either ∞ or $-\infty$ and the central limit theorem can not be applied. Although this problem seems to disappear when the approximation in step (IV) is employed, the justification of the whole procedure may be weak. In fact, the Glicko system treats a collection of games within a "rating period" to have simultaneous occurrences, and it works best when the number of games in a rating period is moderate, say an average of 5-10 games per player in a rating period.[2] The Glicko algorithm for a single game is in Algorithm 5.

6. Experiments

We conduct experiments to assess the performance of our algorithms and TrueSkill on the game data set used by Herbrich et al. (2007). The data are generated by Bungie Studios during the beta testing of the Xbox title Halo 2.[3] The set contains data from four different game types:

- Free for All: up to 8 players in a game. Each team has a single player.
- Small Teams: up to 12 players in 2 teams.[4]
- Head to Head: 2 players in a game. Each player is considered as a team.
- Large Teams: up to 16 players in 2 teams.

2. According to `http://math.bu.edu/people/mg/glicko/glicko.doc/glicko.html`.
3. Credits for the use of the Halo 2 Beta Data set are given to Microsoft Research Ltd. and Bungie.
4. Herbrich et al. (2007) indicate that for "Small Teams," each team has no more than 4 players, and for "Large Teams," each has no more than 8. However, we find a few exceptions.

The numbers of games and players are given in Table 2. In the following, let BT, TM, and PL denote Bradley-Terry, Thurstone-Mosteller, and Plackett-Luce models, respectively; BT-full and BT-partial denote BT with full-pair and partial-pair, and similarly for TM-full and TM-partial. The TrueSkill code is obtained at `http://blogs.technet.com/apg/archive/2008/06/16/trueskill-in-f.aspx`.

6.1 Implementation and Evaluation

Below we discuss initial values and parameters. Generally we follow the setting in Herbrich et al. (2007).

- Initial $\mu_i = 25$ and $\sigma_i^2 = (25/3)^2, \forall i$.
- The additional variance of performance $\beta^2 = (25/6)^2$.
- $\varepsilon = 0.1$ is the draw margin in (65) for the Thurstone-Mosteller model.
- $\kappa = 0.0001$ is the positive lower bound in (28) to avoid negative σ_i^2. The result is insensitive to this parameter as in general $1 - \Delta_i$ is larger than κ.
- γ_q in (50) is set as σ_i / c_{iq} for BT-full. The same γ_q is applied to BT-partial and TM-full. For PL, we use $\gamma_q = \sigma_i / c$. The use of γ_q is further discussed later in this section.

The update rules for the Thurstone-Mosteller model need to calculate the cumulative distribution function $\Phi(x)$, which is not available in most programming languages. We adopt the same way as in TrueSkill to implement the function $\Phi(x)$. Moreover, if the Thurstone-Mosteller model is used, some numerical difficulties may occur. When $x - t$ in (67) is small,

$$\phi(x-t) \approx 0 \text{ and } \Phi(x-t) \approx 0, \tag{89}$$

so the calculation of $V(x,t)$ via $\phi(x-t)/\Phi(x-t)$ is inaccurate. We employ the same safeguard as in TrueSkill:

$$\text{If } \Phi(x-t) \leq 2.222758749 \times 10^{-162}, \text{ then } V(x,t) \text{ is assigned as } -x+t.$$

Note that $-x+t$ is the limit of $V(x,t)$ when $x-t \to -\infty$. We also need some safeguards in calculating $\tilde{V}$ and $\tilde{W}$.

We implement our methods in both C and F#. The F# code is used for the running time comparison with TrueSkill, which is also written in F#. On the same computer, TrueSkill takes 13 seconds to run the "Free for All" data, but BT-full needs only 1.2 seconds. Our method is more efficient because it uses analytic update rules. In contrast, TrueSkill requires an iterative procedure. Moreover, it is simpler to implement our update rules. Using F#, our code takes less than 100 lines, but TrueSkill needs more than 500 lines. Sources used for experiments in this paper are available at

`http://www.csie.ntu.edu.tw/~cjlin/papers/online_ranking`

For the evaluation of prediction results, following Herbrich et al. (2007), we consider the error of using the current μ to predict the outcome of the next game. We check only team pairs whose ranks are different. For example, if there are three teams A, B, and C and the rank of one game is $(1,1,2)$, then only the two pairs (A,C) and (B,C) count. Further, if before the game we have $\mu_A = \mu_C$ and the game output shows rank(A) $<$ rank(C), it is considered a wrong prediction. This situation seldom happens as μ is a real-valued vector, but it does occur in early games because all players' μ were set equally in the beginning. We have confirmed with TrueSkill authors that these detailed settings are the same as what they used in Herbrich et al. (2007). The prediction error rate is the fraction of total team pairs (from the second to the last game) that are wrongly predicted.

Game type	BT-full	BT-partial	PL	TM-full	TrueSkill
Free for All	**35.44**%	36.70%	36.31%	46.11%	35.58%

Table 4: Prediction errors (difficult cases). Team pairs with rank differences no more than two are considered. We consider only "Free for All" because the TrueSkill code provided by authors does not handle multi-player teams and we have not conducted suitable modifications. Moreover, under our selection rule, all games in "Head to Head" will be selected and results are the same as Table 2. Hence this set is not included either.

Avg. Occurances	Num. Pairs	BT-full	TrueSkill
$\leq$5	23,567	38.74%	39.15%
$\leq$10	69,145	36.22%	36.41%
$\leq$20	148,654	34.54%	34.52%
$\leq$40	276,203	32.64%	32.64%
No restriction	595,500	30.59%	30.74%

(a) Free for All

Num. Pairs	BT-full	TrueSkill
2,367	38.70%	38.36%
3,748	35.17%	34.61%
4,852	33.29%	33.02%
5,501	32.61%	32.61%
5,715	32.53%	32.49%

(b) Head to Head

Table 5: Prediction errors for competitions where players have only played few games. Games with the average number of players' past appearances no more than the value in the first column are considered. The last row includes all games. The second column indicates the number of total team pairs used for the evaluation. The 30.74% and 32.49% rates by TrueSkill are slightly different from 30.82% and 32.44% in Table 2, respectively, because the former is from running the F# code provided by TrueSkill authors, but the latter is copied from Herbrich et al. (2007).

6.2 Comparison on Prediction Errors

We report the prediction error in Table 2 and make the following observations. First, BT-full, BT-partial, and PL have the same error rate except "Free for All." This result is reasonable as when every game involves only two teams, using full pairs, partial pairs or the Plackett-Luce model does not make any difference. Second, when the number of teams is more than two (i.e., Free for All), BT-full is better than BT-partial. The same observation holds when comparing TM-full and TM-partial (numbers not shown). A possible explanation is that the full-pair approach uses more information. Third, using the Bradley-Terry model yields superior results to the Thurstone-Mosteller model. The error of using TM-full on "Free for All" is very high. Besides, numerical problems discussed in (89) do not occur for the Bradley-Terry model. Fourth, TM-full, which uses the same likelihood model as TrueSkill, is consistently worse than TrueSkill, indicating that the much faster, single-pass approximation may come at the expense of less accurate prediction. Finally, our proposed method for BT-full and PL is competitive with TrueSkill.

The reason why TM-full performs poorly for "Free for All" in Table 2 might be that σ_i quickly goes to zero and μ_i becomes a huge positive/negative value. The parameter γ_q in (50) can help to control how fast the variance σ_i^2 is reduced. In Table 2, γ_q is set as σ_i/c_{iq}. Table 3 gives results of using $\gamma_q = 1/k$, where k is the number of teams in a game. For "Free for All," k is around 8, so γ_q is quite small. Clearly, a slower reduction of σ_i^2 significantly improves the performance of TM-full, while the results of BT-full and PL do not change much.

We conduct a further comparison using only team pairs which are more difficult for prediction. For "Free for All," the team pairs whose ranks in a game are closer can be viewed as difficult cases

for prediction. We take all pairs with rank differences no more than two and compare the prediction errors by our methods and TrueSkill. The results, shown in Table 4, are consistent with those in Table 2.

After a team (or player) has played many games, the obtained ability becomes more accurate. To check the performance when teams have only played few games, we select games where the average number of players' past appearances is small. We present results in Table 5. Clearly if players in a game have only played few games, the prediction is more difficult.

We also implement the single game version of Glicko (Algorithm 5) for "Head to Head" and find the prediction error to be 33.88%, a bit worse than those in Table 2. Such a result is expected as Glicko is not designed to update skills after each single game.

Finally, we discuss how to apply our proposed technique in practice. Following the experimental results and the numerical concerns, TM is not recommended. Further as BT-full is slightly better than BT-partial, it seems that to factorize a multi-team game to several two-team games, we should use as much information as possible. Therefore, in applying our approximation, BT-full and PL may be the first choice. As TM-full uses the same likelihood as TrueSkill and performs worse, our approximation, while very simple, may be more sensitive to the likelihood used.

7. Discussion and Conclusions

Huang and Frey (2008) propose a graphical model, cumulative distribution network (CDF), which can be used for online ranking. They experiment with the same data used by Herbrich et al. (2007) and report superior results. However, they use a full covariance matrix over all skills of all players. This setting provides more information for accurate predictions, but may not be practical for large-scale systems.

Guiver and Snelson (2009) apply Power EP (expectation propagation) to perform Bayesian inference for parameters of the Plackett-Luce model. They conduct experiments in an offline setting on NASCAR 2002 car racing results and the MovieLens data set. It is worth studying the performance in online setting. We leave it for future work.

In summary, this paper approximates the expectation of teams' performances to derive simple update rules for online ranking. The proposed method is efficient and can be easily applied to large-scale systems with multiple teams and multiple players. While the approximation of the expectation is only a kind of heuristics, experiments show that its application to BT-full and PL models is competitive with state of the art approaches such as TrueSkill. Further, the implementation is simpler and the running time is shorter.

Acknowledgments

The authors thank Ralf Herbrich and Thore Graepel of Microsoft Research Cambridge for providing their F# codes (Free for All and Head to Head) and for answering many questions.

Appendix A. A Sketch of the Proof for Lemma 1

We borrow a few lines from Woodroofe (1989) to sketch the proof for (16) in the 1-dimensional case. Let $'$ denote the differentiation and Φh denote $\int_R h(z)d\Phi(z)$. By assumptions in Lemma 1, we

have $f(z) = \int_{-\infty}^{z} f'(y)dy$ and

$$\begin{aligned} & \int_R h(z)d\Gamma(z) - \int_R f(z)d\Phi(z) \cdot \int_R h(z)d\Phi(z) \\ = & \int_R f(z)\phi(z)[h(z) - \Phi h]dz = \int_R \Big\{ \int_{-\infty}^{z} f'(y)dy \Big\} \phi(z)[h(z) - \Phi h]dz \\ = & \int_R \Big\{ \int_{y}^{\infty} \phi(z)[h(z) - \Phi h]dz \Big\} f'(y)dy = \int_R Uh(y)f'(y)\phi(y)dy, \end{aligned}$$

where the interchange of orders of integration is justified by assumed integrability conditions.

Appendix B. An Example on Calculating Uh and Vh in (14)

We take $k = 3$ and $h(\mathbf{z}) = z_1 z_2$ to illustrate the calculation of Uh and Vh. First by (12) we obtain

$$\begin{aligned} & h_0 = \int z_1 z_2 d\Phi_3(\mathbf{z}) = 0, \\ & h_1(z_1) = \int h(z_1, w_1, w_2) d\Phi_2(w_1, w_2) = \int z_1 w_1 d\Phi_2(w_1, w_2) = 0, \\ & h_2(z_1, z_2) = \int h(z_1, z_2, w) d\Phi_1(w) = \int z_1 z_2 d\Phi(w) = z_1 z_2, \\ & h_3(z_1, z_2, z_3) = h(z_1, z_2, z_3) = z_1 z_2. \end{aligned}$$

Next from (13) it follows that

$$\begin{aligned} g_1(\mathbf{z}) &= e^{z_1^2/2} \int_{z_1}^{\infty} [h_1(w) - h_0] e^{-w^2/2} dw = 0, \\ g_2(\mathbf{z}) &= e^{z_2^2/2} \int_{z_2}^{\infty} [h_2(z_1, w) - h_1(z_1)] e^{-w^2/2} dw = e^{z_2^2/2} \int_{z_2}^{\infty} z_1 w e^{-w^2/2} dw = z_1, \\ g_3(\mathbf{z}) &= e^{z_3^2/2} \int_{z_3}^{\infty} [h_3(z_1, z_2, w) - h_2(z_1, z_2)] e^{-w^2/2} dw = 0; \end{aligned}$$

hence, by (14) we have $Uh(\mathbf{z}) = (g_1, g_2, g_3)^T = (0, z_1, 0)^T$. Applying the same steps to g_i gives $Ug_1 = Ug_3 = [0,0,0]^T$ and $Ug_2 = [1,0,0]^T$. Therefore, by (14) we obtain

$$Vh = \frac{1}{2}\left(U^2 h + (U^2 h)^T\right) = \frac{1}{2}\left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right) = \begin{bmatrix} 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Appendix C. A Bradley-Terry Model with Variance Parameters

Our approach is motivated by the relation between the normal model (5) and the Bradley-Terry model (7). To begin, we reparametrize v_i in (7) as $e^{\theta_i/c}$ and similarly for v_q so that (7) can be written as

$$P(X_i > X_q) = \frac{e^{(\theta_i - \theta_q)/c}}{1 + e^{(\theta_i - \theta_q)/c}}. \tag{90}$$

Next, observe that the cumulative distribution function of a logistic distribution with mean 0 and variance $(c\pi/\sqrt{3})^2$ is

$$F(x) = \frac{e^{x/c}}{(1 + e^{x/c})},$$

which can be approximated by the cumulative distribution function of a normal distribution with the same mean and variance. Therefore,

$$\begin{aligned}\frac{e^{(\theta_i-\theta_q)/c}}{1+e^{(\theta_i-\theta_q)/c}} &\approx \int_{-\infty}^{\theta_i-\theta_q} \frac{1}{\sqrt{2\pi}(c\pi)/\sqrt{3}} e^{-u^2/(2(c\pi/\sqrt{3})^2)} du \\ &= \Phi\left(\frac{\theta_i-\theta_q}{c\pi/\sqrt{3}}\right). \end{aligned} \tag{91}$$

The idea of approximating the logistic distribution in an integral by a Gaussian one has appeared in Aitchison and Begg (1976), Glickman (1993), and references therein. By comparing (91) with (5), it suggests to take $c^2 \propto (\beta_i^2+\beta_q^2)$ and then replace v_i and v_q in (7) with $e^{\theta_i/c}$ and $e^{\theta_q/c}$. In summary, we have shown that (90) can be obtained by assuming that each team has a performance uncertainty parameter β_i^2, and that when teams i and q compete, their actual performance follow Gumbel distributions with cumulative distribution function

$$P(X_i \le x) = \exp(-\exp(-(x-\frac{\theta_i}{c}))),$$

where $c^2 = \beta_i^2+\beta_q^2$. Note that this model presumes that a team's actual performance depends on the team it competes with.

Regarding the error induced by evaluating the expectations in (25) and (26), we can apply the same analysis in Section 3.2 to the Bradley-Terry model. Here we give details. By (48), the joint posterior density of (θ_1,θ_2) is proportional to

$$\phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\phi\left(\frac{\theta_2-\mu_2}{\sigma_2}\right)\frac{e^{\theta_1/c_{12}}}{e^{\theta_1/c_{12}}+e^{\theta_2/c_{12}}}.$$

Next, by an approximation like (91), the marginal posterior density of θ_1 is approximately proportional to

$$\begin{aligned} & \phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\int\phi\left(\frac{\theta_2-\mu_2}{\sigma_1}\right)\int_{-\infty}^{\theta_1}\frac{1}{\sqrt{2\pi}(\alpha c_{12})}e^{-\frac{y-\theta_2}{2(\alpha c_{12})^2}}dyd\theta_2 \\ \approx\ & \phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\Phi\left(\frac{\theta_1-\mu_2}{\sqrt{(\alpha c_{12})^2+\sigma_2^2}}\right) \\ \approx\ & \phi\left(\frac{\theta_1-\mu_1}{\sigma_1}\right)\frac{e^{\theta_1/c'_{12}}}{e^{\theta_1/c'_{12}}+e^{\theta_2/c'_{12}}}, \end{aligned}$$

where $\alpha=\pi/\sqrt{3}$ as in (91) and $(c'_{12})^2=\alpha^2c_{12}^2+\sigma_2^2$. As in the previous paragraph, we can calculate the posterior mean of θ_1, and again the result suggests that the bias induced by our approximation method can be reduced by substituting β_i^2 with $\beta_i^2+\sigma_i^2$.

Appendix D. Derivations of Update Rules for the Bradley-Terry Model (Partial-pair)

To calculate $\partial f/\partial z_i$, if $i=r(a)$, then in (63) there are only two terms related to i:

$$f_{r(a-1)r(a)}(\mathbf{z}) \text{ and } f_{r(a)r(a+1)}(\mathbf{z}).$$

Define Q as in (61). Then,

$$\frac{\partial f/\partial z_i}{f} = \sum_{q:q\in Q, r(q)<r(i)} \frac{\partial f_{qi}/\partial z_i}{f_{qi}} + \sum_{q:q\in Q, r(q)>r(i)} \frac{\partial f_{iq}/\partial z_i}{f_{iq}} + \frac{1}{2} \sum_{q:q\in Q, r(q)=r(i), q\neq i} \left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}} + \frac{\partial f_{iq}/\partial z_i}{f_{iq}} \right).$$

Next,

$$\frac{\partial}{\partial z_i}\left(\frac{\partial f/\partial z_i}{f}\right) = \sum_{q:q\in Q, r(q)<r(i)} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}}\right) + \sum_{q:q\in Q, r(q)>r(i)} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right) + \frac{1}{2} \sum_{q:q\in Q, r(q)=r(i), q\neq i} \left(\frac{\partial}{\partial z_i}\left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}}\right) + \frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right)\right).$$

These two results are almost the same as (53) and (59) used for the full-pair case. Hence δ_q and η_q are calculated by the same way as in Algorithm 1, but for Ω_i and Δ_i, instead of taking the sum over all $q = 1,\ldots,k; q \neq i$, in (62) we sum up only elements in the set Q.

Appendix E. Derivations of Update Rules for the Thurstone-Mosteller Model

Define

$$f_{iq}(\mathbf{z}) \equiv P(\text{team } i \text{ beats team } q) = \Phi\left(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}}\right)$$

and

$$f_{iq}(\mathbf{z}) \equiv P(\text{team } i \text{ draws with team } q) = \Phi\left(\frac{\varepsilon - (\theta_i - \theta_q)}{c_{iq}}\right) - \Phi\left(\frac{-\varepsilon - (\theta_i - \theta_q)}{c_{iq}}\right),$$

where $\theta_i = \sigma_i z_i + \mu_i$. Then

$$P(\text{outcome of team } i \text{ and } q) = \begin{cases} f_{iq}(\mathbf{z}) & \text{if } r(i) > r(q), \\ f_{qi}(\mathbf{z}) & \text{if } r(i) < r(q), \\ f_{iq}(\mathbf{z}) & \text{if } r(i) = r(q). \end{cases}$$

Similar to the derivation for the Bradley-Terry model in (52) and (53),

$$\frac{\partial f/\partial z_i}{f} = \sum_{q:r(q)<r(i)} \frac{\partial f_{qi}/\partial z_i}{f_{qi}} + \sum_{q:r(q)>r(i)} \frac{\partial f_{iq}/\partial z_i}{f_{iq}} + \sum_{q:r(q)=r(i)} \frac{\partial f_{iq}/\partial z_i}{f_{iq}}.$$

Using the relation between ϕ and Φ in (66),

$$\frac{\partial}{\partial \theta_i}\Phi\left(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}}\right) = \phi\left(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}}\right)\frac{1}{c_{iq}}. \tag{92}$$

Therefore,

$$\frac{\partial f_{iq}/\partial z_i}{f_{iq}} = \frac{1}{c_{iq}} \frac{\phi(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}})}{\Phi(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}})} \cdot \frac{\partial \theta_i}{\partial z_i} = \frac{\sigma_i}{c_{iq}} V\left(\frac{\theta_i - \theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right), \tag{93}$$

where the function V is defined in (67). Similarly,

$$\frac{\partial f_{qi}}{\partial z_i} = \frac{-\sigma_i}{c_{iq}} V\left(\frac{\theta_q - \theta_i}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right).$$

For $f_{iq}(\theta)$,

$$\frac{\partial f_{iq}}{\partial z_i} = \frac{-\sigma_i}{c_{iq}} \left(\phi(\frac{\varepsilon - (\theta_i - \theta_q)}{c_{iq}}) - \phi(\frac{-\varepsilon - (\theta_i - \theta_q)}{c_{iq}})\right),$$

so

$$\frac{\partial f_{iq}/\partial z_i}{f_{iq}} = \frac{-\sigma_i}{c_{iq}} \frac{\phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})}{\Phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \Phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})} = \frac{\sigma_i}{c_{iq}} \tilde{V}\left(\frac{\theta_i - \theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right), \tag{94}$$

where the function $\tilde{V}$ is defined in (68). Then the update rule is

$$\begin{aligned}
\mu_i \leftarrow & \mu_i + \sigma_i \left.\frac{\partial f(\mathbf{z})/\partial z_i}{f}\right|_{\mathbf{z}=\mathbf{0}} \\
\leftarrow & \mu_i + \sigma_i^2 \left(\sum_{q:r(q)<r(i)} \frac{-1}{c_{iq}} V\left(\frac{\mu_q - \mu_i}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right) + \sum_{q:r(q)>r(i)} \frac{1}{c_{iq}} V\left(\frac{\mu_i - \mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right) \right. \\
& \left. + \sum_{q:r(q)=r(i), q\neq i} \frac{1}{c_{iq}} \tilde{V}\left(\frac{\mu_i - \mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right)\right).
\end{aligned}$$

To update σ, similar to (59), we have

$$\begin{aligned}
\frac{\partial}{\partial z_i}\left(\frac{\partial f/\partial z_i}{f}\right) = & \sum_{q:r(q)<r(i)} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{qi}/\partial z_i}{f_{qi}}\right) + \sum_{q:r(q)>r(i)} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right) \\
& + \sum_{q:r(q)=r(i), q\neq i} \frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right).
\end{aligned}$$

Using (93) and the fact that $d\phi(x)/dx = -x\phi(x)$

$$\begin{aligned}
& \frac{\partial}{\partial z_i}\frac{\partial f_{iq}/\partial z_i}{f_{iq}} \\
= & \frac{\sigma_i}{c_{iq}} \frac{\partial(\phi/\Phi)}{\partial \theta_i} \cdot \frac{\partial \theta_i}{\partial z_i} = \frac{\sigma_i^2}{c_{iq}} \frac{\Phi \frac{d\phi}{d\theta_i} - \phi \frac{d\Phi}{d\theta_i}}{\Phi^2} \\
= & \frac{\sigma_i^2}{c_{iq}} \left(-\left(\frac{\theta_i - \theta_q - \varepsilon}{c_{iq}}\right) \cdot V\left(\frac{\theta_i - \theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right) \frac{1}{c_{iq}} - \frac{1}{c_{iq}} V\left(\frac{\theta_i - \theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right)^2 \right) \\
= & -\frac{\sigma_i^2}{c_{iq}^2} W\left(\frac{\theta_i - \theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right),
\end{aligned} \tag{95}$$

where the function W is defined in (67). Similarly,

$$\frac{\partial}{\partial z_i}\frac{\partial f_{qi}/\partial z_i}{f_{qi}} = -\frac{\sigma_i^2}{c_{iq}} W\left(\frac{\theta_q - \theta_i}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right). \tag{96}$$

If $r(i) = r(q)$, then we use (92) and (94) to calculate

$$\frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right) = \frac{-\sigma_i^2}{c_{iq}} \frac{A-B}{\left(\Phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \Phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})\right)^2},$$

where

$$A = \frac{1}{c_{iq}}\left(\Phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \Phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})\right) \times \left(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}\phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}}\phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})\right)$$

and

$$B = \frac{-1}{c_{iq}}\left(\phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})\right)^2.$$

Hence

$$\begin{aligned}
&\frac{\partial}{\partial z_i}\left(\frac{\partial f_{iq}/\partial z_i}{f_{iq}}\right) \\
=&\frac{-\sigma_i^2}{c_{iq}^2}\left(\frac{\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}\phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}}\phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})}{\Phi(\frac{\varepsilon-(\theta_i-\theta_q)}{c_{iq}}) - \Phi(\frac{-\varepsilon-(\theta_i-\theta_q)}{c_{iq}})} + \tilde{V}\left(\frac{\theta_i-\theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right)^2\right) \qquad (97)\\
=&\frac{-\sigma_i^2}{c_{iq}^2}\tilde{W}\left(\frac{\theta_i-\theta_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}\right),
\end{aligned}$$

where the function $\tilde{W}$ is defined in (69). Combining (95), (96), and (97), the update rule for σ_i^2 is

$$\sigma_i^2 \leftarrow \sigma_i^2\left(1 - \left(\sum_{q:r(q)<r(i)} \frac{\sigma_i^2}{c_{iq}^2} W(\frac{\mu_q-\mu_i}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) + \sum_{q:r(q)>r(i)} \frac{\sigma_i^2}{c_{iq}^2} W(\frac{\mu_i-\mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}}) \right.\right.$$
$$\left.\left. \sum_{q:r(q)=r(i), q\neq i} \frac{\sigma_i^2}{c_{iq}^2} \tilde{W}(\frac{\mu_i-\mu_q}{c_{iq}}, \frac{\varepsilon}{c_{iq}})\right)\right).$$

Appendix F. Derivations of Update Rules for the Plackett-Luce Model

Using $f(\mathbf{z})$ and $f_q(\mathbf{z})$ defined in (72),

$$f_q(\mathbf{z}) = \left(\frac{e^{\theta_q/c}}{\sum_{s\in C_q} e^{\theta_s/c}}, \right)^{1/A_q},$$

so

$$\begin{aligned}
\frac{\partial f_q/\partial z_i}{f_q} =& \frac{\partial \log f_q}{\partial z_i} = \frac{1}{A_q}\left(\frac{\partial(\theta_q/c)}{\partial \theta_i} - \frac{\partial \log(\sum_{s\in C_q} e^{\theta_s/c})}{\partial \theta_i}\right)\frac{\partial \theta_i}{\partial z_i}\\
=& \frac{\sigma_i}{cA_q}\begin{cases} 1 - \frac{e^{\theta_i/c}}{\sum_{s\in C_q} e^{\theta_s/c}} & \text{if } q = i,\\ -\frac{e^{\theta_i/c}}{\sum_{s\in C_q} e^{\theta_s/c}} & \text{if } r(q) \le r(i), q \neq i,\\ 0 & \text{if } r(q) > r(i). \end{cases} \qquad (98)
\end{aligned}$$

From (38), the update rule is

$$\mu_i \leftarrow \mu_i + \Omega_i,$$

where

$$\begin{aligned}\Omega_i =& \sigma_i \sum_{q=1}^{k} \left.\frac{\partial f_q(\mathbf{z})/\partial z_i}{f_q(\mathbf{z})}\right|_{\mathbf{z}=\mathbf{0}} \\ =& \frac{\sigma_i^2}{c}\left(\frac{1}{A_i}\left(1-\frac{e^{\mu_i/c}}{\sum_{s\in C_i} e^{\mu_s/c}}\right) + \sum_{q:q\neq i, r(q)\leq r(i)} -\frac{1}{A_q}\frac{e^{\mu_i/c}}{\sum_{s\in C_q} e^{\mu_s/c}}\right).\end{aligned}$$

To update σ, similar to (59), we must calculate

$$\frac{\partial}{\partial z_i}\left(\frac{\partial f_q/\partial z_i}{f_q}\right), \forall q. \tag{99}$$

From (98), if $i \in C_q$, then

$$\begin{aligned}(99) =& -\frac{\sigma_i}{cA_q}\left(\frac{\partial}{\partial \theta_i}\frac{e^{\theta_i/c}}{\sum_{s\in C_q} e^{\theta_s/c}}\right)\cdot\frac{\partial \theta_i}{\partial z_i} = \frac{\sigma_i^2}{c^2 A_q}\frac{(\sum_{s\in C_q} e^{\theta_s/c})e^{\theta_i/c} - (e^{\theta_i/c})^2}{(\sum_{s\in C_q} e^{\theta_s/c})^2} \\ =& \frac{\sigma_i^2}{c^2 A_q}\frac{e^{\theta_i/c}}{\sum_{s\in C_q} e^{\theta_s/c}}\left(1-\frac{e^{\theta_i/c}}{\sum_{s\in C_q} e^{\theta_s/c}}\right).\end{aligned}$$

The update rule for σ_i^2 is

$$\sigma_i^2 \leftarrow \sigma_i^2\left(1-\sum_{q:r(q)\leq r(i)}\frac{1}{c^2 A_q}\frac{e^{\mu_i/c}}{\sum_{s\in C_q} e^{\mu_s/c}}\left(1-\frac{e^{\mu_i/c}}{\sum_{s\in C_q} e^{\mu_s/c}}\right)\right).$$

References

John Aitchison and Colin B. Begg. Statistical diagnosis when basic cases are not classified with certainty. *Biometrika*, 63:1–12, 1976.

Ralph A. Bradley and Milton E. Terry. The rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39:324–345, 1952.

Arpad E. Elo. *The Rating of Chessplayers, Past and Present*. Arco Publishing, New York, 2nd edition, 1986.

Mark E. Glickman. *Paired Comparison Models with Time-varying Parameters*. PhD thesis, Department of Statistics, Harvard University, 1993.

Mark E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics*, 48:377–394, 1999.

John Guiver and Edward Snelson. Bayesian inference for Plackett-Luce ranking model. In *Proceedings of the Twenty Sixth International Conference on Machine Learning (ICML)*, 2009.

Ralf Herbrich, Tom Minka, and Thore Graepel. TrueSkill^{TM}: A Bayesian skill rating system. In *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

Jim C. Huang and Brendan J. Frey. Cumulative distribution networks and the derivative-sum-product algorithm. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI)*, 2008.

Tzu-Kuo Huang, Ruby C. Weng, and Chih-Jen Lin. Generalized Bradley-Terry models and multi-class probability estimates. *Journal of Machine Learning Research*, 7:85–115, 2006. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/generalBT.pdf`.

Willard D. James and Charles M. Stein. Estimation with quadratic loss. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 361–379, 1961.

Robert E. Kass and Adrian E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

John I. Marden. *Analyzing and Modeling Rank Data*. Chapman & Hall, London, 1995.

Joshua E. Menke and Tony R. Martinez. A Bradley-Terry artificial neural network model for individual ratings in group competitions. *Neural Computing and Applications*, 17:175–186, 2008.

Thomas Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, MIT, 2001.

Charles M. Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9:1135–1151, 1981.

Louis L. Thurstone. A law of comparative judgement. *Psychological Reviews*, 34:273–286, 1927.

Luke Tierney and Joseph B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81:82–86, 1986.

Ruby C. Weng. A Bayesian Edgeworth expansion by Stein's identity. *Bayesian Analysis*, 5(4): 741–764, 2010.

Ruby C. Weng and Michael Woodroofe. Integrable expansions for posterior distributions for multi-parameter exponential families with applications to sequential confidence levels. *Statistica Sinica*, 10:693–713, 2000.

Michael Woodroofe. Very weak expansions for sequentially designed experiments: linear models. *The Annals of Statistics*, 17:1087–1102, 1989.

Michael Woodroofe and D. Stephen Coad. Corrected confidence sets for sequentially designed experiments. *Statistica Sinica*, 7:53–74, 1997.

Ernst Zermelo. Die berechnung der Turnier-Ergebnisse als ein maximumproblem der wahrscheinlichkeitsrechnung. *Mathematische Zeitschrift*, 29:436–460, 1929.

Journal of Machine Learning Research 12 (2011) 301-348 Submitted 10/09; Revised 8/10; Published 1/11

Cumulative Distribution Networks and the Derivative-sum-product Algorithm: Models and Inference for Cumulative Distribution Functions on Graphs

Jim C. Huang JIMHUA@MICROSOFT.COM
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA

Brendan J. Frey FREY@PSI.TORONTO.EDU
University of Toronto
10 King's College Rd.
Toronto, ON M5S 3G4, Canada

Editor: Tommi Jaakkola

Abstract

We present a class of graphical models for directly representing the joint cumulative distribution function (CDF) of many random variables, called *cumulative distribution networks* (CDNs). Unlike graphs for probability density and mass functions, for CDFs the marginal probabilities for any subset of variables are obtained by computing limits of functions in the model, and conditional probabilities correspond to computing mixed derivatives. We will show that the conditional independence properties in a CDN are distinct from the conditional independence properties of directed, undirected and factor graphs, but include the conditional independence properties of bi-directed graphs. In order to perform inference in such models, we describe the 'derivative-sum-product' (DSP) message-passing algorithm in which messages correspond to derivatives of the joint CDF. We will then apply CDNs to the problem of learning to rank players in multiplayer team-based games and suggest several future directions for research.

Keywords: graphical models, cumulative distribution function, message-passing algorithm, inference

1. Introduction

Probabilistic graphical models provide a pictorial means of specifying a joint probability density function (PDF) defined over many continuous random variables, the joint probability mass function (PMF) of many discrete random variables, or a joint probability distribution defined over a mixture of continuous and discrete variables. Each variable in the model corresponds to a node in a graph and edges between nodes in the graph convey statistical dependence relationships between the variables in the model. The graphical formalism allows one to obtain the independence relationships between random variables in a model by inspecting the corresponding graph, where the separation of nodes in the graph implies a particular conditional independence relationship between the corresponding variables.

A consequence of representing independence constraints between subsets of variables using a graph is that the joint probability often factors into a product of functions defined over subsets of

©2011 Jim C. Huang and Brendan J. Frey.

neighboring nodes in the graph. Typically, this allows us to decompose a large multivariate distribution into a product of simpler functions, so that the task of inference and estimation of such models can also be simplified and efficient algorithms for performing these tasks can be implemented. Often, a complex distribution over observed variables can be constructed using a graphical model with latent variables introduced, where the joint probability over the observed variables is obtained by marginalization over the latent variables. The model with additional latent variables has the advantage of having a more compact factorized form as compared to that for the joint probability over the observed variables. However, this often comes at the cost of a significantly higher computational cost for estimation and inference, as additional latent variables often require one to either approximate intractable marginalization operations (Minka, 2001) or to sample from the model using Markov Chain Monte Carlo (MCMC) methods (Neal, 1993). Furthermore, there is also the problem that there are possibly an infinite number of latent variable models associated with any given model defined over observable variables, so that adding latent variables for any given application can often present difficulties in terms of model identifiability, which may be desirable when model parameters are to be interpreted. These issues may hamper the applicability of graphical models for many real-world problems in the presence of latent variables.

Another possible limitation of many graphical models is that the joint PDF/PMF itself might not be appropriate as a probability model for certain applications. For example, in learning to rank, the *cumulative distribution function* (CDF) is a probabilistic representation that arises naturally as a probability of inequality events of the type $\{X \leq x\}$. The joint CDF lends itself to such problems that are easily described in terms of inequality events in which statistical dependence relationships also exist among events. An example of this type of problem is that of predicting multiplayer game outcomes with a team structure (Herbrich, Minka and Graepel, 2007). In contrast to the canonical problems of classification or regression, in learning to rank we are required to learn some mapping from inputs to inter-dependent output variables so that we may wish to model both stochastic orderings between variable states and statistical dependence relationships between variables.

Given the above, here we present a class of graphical models called *cumulative distribution networks* (CDN) in which we represent the joint CDF of a set of observed variables. As we will show, CDNs can be viewed as providing a means to construct multivariate distributions over observed variables without the need to explicitly introduce latent variables and then marginalize. The resulting model consists of a factorized form for the joint CDF, where the principal operations required for answering probabilistic queries and for marginalization consist of differentiation and computing limits respectively, in contrast to summation/integration in graphical models for PDFs with latent variables. Furthermore, the parametrization of the model as a joint CDF has the advantage that the global normalization constraint can be enforced locally for each function in the CDN, unlike the case of undirected graphical models for PDF/PMFs. We will present the basic properties of CDNs and show that the rules for ascertaining conditional independence relationships among variables in a CDN are distinct from the rules in directed, undirected and factor graphs (Pearl, 1988; Lauritzen, 1996; Kschischang, Frey and Loeliger, 2001). We will show that the conditional independence properties in a CDN include, but are not limited to, the conditional independence properties for bi-directed graphs (Drton and Richardson, 2008; Richardson and Spirtes, 2002; Richardson, 2003).

We will then discuss the problem of performing inference under CDNs in which the principal challenge is to compute the derivatives of the joint CDF. To this end we will describe a message-passing algorithm for inference in CDNs called the *derivative-sum-product algorithm* based on previous work (Huang and Frey, 2008; Huang, 2009). To demonstrate the applicability of CDNs,

we will use the message-passing algorithm for inference in order to apply CDNs to the problem of learning to rank, where we will show that CDFs arise naturally as a probability models in which it is easy to specify stochastic ordering constraints among variables in the model.

1.1 Notation

Before we proceed, we will establish some notation to be used throughout the paper. We will denote bipartite graphs as $\mathcal{G} = (V, S, E)$ where V, S are two disjoint sets of nodes and $E \subseteq \{V \times S, S \times V\}$ is a set of edges that correspond to ordered pairs (α, s) or (s, α) for $\alpha \in V$ and $s \in S$. We will denote neighboring sets $\mathcal{N}(\alpha)$ and $\mathcal{N}(s)$ as

$$\mathcal{N}(\alpha) = \{s \in S : (\alpha, s) \in E\},$$
$$\mathcal{N}(s) = \{\alpha \in V : (\alpha, s) \in E\}.$$

Furthermore, let $\mathcal{N}(A) = \cup_{\alpha \in A} \mathcal{N}(\alpha)$.

Throughout the paper we will use boldface notation to denote vectors and/or matrices. Scalar and vector random variables will be denoted as X_α and $\mathbf{X}_A$ respectively where α is a node in a graph $\mathcal{G}$ and A denotes a set of nodes in $\mathcal{G}$. The notation $|A|, |\mathbf{x}|, |\mathbf{X}|$ will denote the cardinality, or number of elements, in set A and vectors $\mathbf{x}, \mathbf{X}$ respectively. We will also denote the mixed partial derivative/finite difference as $\partial_{\mathbf{x}_A}\left[\cdot\right]$, where the mixed derivative here is taken with respect to arguments $x_\alpha \; \forall \; \alpha \in A$. Throughout the paper we assume hat sets consist of unique elements such that for any set A and for any element $\alpha \in A$, $A \cap \alpha = \alpha$, so that $\partial_{\mathbf{x}_A}\left[\cdot\right]$ consists of the mixed derivative with respect to unique variable arguments $X_\alpha \in \mathbf{X}_A$. For example, $\partial_{x_{1,2,3}}\left[F(x_1, x_2, x_3)\right] \equiv \frac{\partial^3 F}{\partial x_1 \partial x_2 \partial x_3}$.

1.2 Cumulative Distribution Functions

Here we provide a brief definition for the joint CDF $F(\mathbf{x})$ defined over random variables $\mathbf{X}$, denoted individually as X_α. The joint *cumulative distribution function* $F(\mathbf{x})$ is then defined as the function $F : \mathbb{R}^{|\mathbf{X}|} \mapsto [0, 1]$ such that

$$F(\mathbf{x}) = \mathbb{P}\left[\bigcap_{X_\alpha \in \mathbf{X}} \left\{X_\alpha \leq x_\alpha\right\}\right] \equiv \mathbb{P}\left[\mathbf{X} \leq \mathbf{x}\right].$$

Thus the CDF is a probability of events $\{X_\alpha \leq x_\alpha\}$. Alternately, the CDF can be defined in terms of the joint probability density function (PDF) or probability mass function (PMF) $P(\mathbf{x})$ via

$$F(\mathbf{x}) = \int_{-\infty}^{\mathbf{x}} P(\mathbf{u}) \, d\mathbf{u},$$

where $P(\mathbf{x})$, if it exists, satisfies $P(\mathbf{x}) \geq 0$, $\int_{\mathbf{x}} P(\mathbf{x}) \, d\mathbf{x} = 1$ and $P(\mathbf{x}) = \partial_{\mathbf{x}}\left[F(\mathbf{x})\right]$ where $\partial_{\mathbf{x}}\left[\cdot\right]$ denotes the higher-order mixed derivative operator $\partial_{x_1, \cdots, x_K}\left[\cdot\right] \equiv \dfrac{\partial^K}{\partial x_1 \cdots \partial x_K}$ for $\mathbf{x} = [x_1 \; \cdots \; x_K] \in \mathbb{R}^K$.

A function F is a CDF for some probability $\mathbb{P}$ if and only if F satisfies the following conditions:

1. The CDF $F(\mathbf{x})$ converges to unity as all of its arguments tend to ∞, or

$$F(\infty) \equiv \lim_{\mathbf{x} \to \infty} F(\mathbf{x}) = 1.$$

2. The CDF $F(\mathbf{x})$ converges to 0 as any of its arguments tends to $-\infty$, or

$$F(-\infty, \mathbf{x} \setminus x_\alpha) \equiv \lim_{x_\alpha \to -\infty} F(x_\alpha, \mathbf{x} \setminus x_\alpha) = 0 \quad \forall\, X_\alpha \in \mathbf{X}.$$

3. The CDF $F(\mathbf{x})$ is monotonically non-decreasing, so that

$$F(\mathbf{x}) \leq F(\mathbf{y}) \;\forall\, \mathbf{x} \leq \mathbf{y},\; \mathbf{x}, \mathbf{y} \in \mathbb{R}^{|\mathbf{X}|}.$$

   where $\mathbf{x} \leq \mathbf{y}$ denotes element-wise inequality of all the elements in vectors $\mathbf{x}, \mathbf{y}$.

4. The CDF $F(\mathbf{x})$ is right-continuous, so that

$$\lim_{\boldsymbol{\epsilon} \to \mathbf{0}^+} F(\mathbf{x} + \boldsymbol{\epsilon}) \equiv F(\mathbf{x}) \;\forall\, \mathbf{x} \in \mathbb{R}^{|\mathbf{X}|}.$$

A proof of forward implication in the above can be found in Wasserman (2004) and Papoulis and Pillai (2001).

Proposition 1 *Let $F(\mathbf{x}_A, \mathbf{x}_B)$ be the joint CDF for variables $\mathbf{X}$ where $\mathbf{X}_A, \mathbf{X}_B$ for a partition of the set of variables $\mathbf{X}$. The joint probability of the event $\{\mathbf{X}_A \leq \mathbf{x}_A\}$ is then given in terms of $F(\mathbf{x}_A, \mathbf{x}_B)$ as*

$$F(\mathbf{x}_A) \equiv \mathbb{P}\Big[\mathbf{X}_A \leq \mathbf{x}_A\Big] = \lim_{\mathbf{x}_B \to \infty} F(\mathbf{x}_A, \mathbf{x}_B).$$

□

The above proposition follows directly from the definition of a CDF in which

$$\lim_{\mathbf{x}_B \to \infty} F(\mathbf{x}_A, \mathbf{x}_B) = \mathbb{P}\left[\left[\bigcap_{\alpha \in A} \{X_\alpha \leq x_\alpha\}\right] \cap \left[\bigcap_{\beta \in B} \{X_\beta \leq \infty\}\right]\right] = \mathbb{P}\left[\bigcap_{\alpha \in A} \{X_\alpha \leq x_\alpha\}\right] = F(\mathbf{x}_A).$$

Thus, marginal CDFs of the form $F(\mathbf{x}_A)$ can be computed from the joint CDF by computing limits.

1.3 Conditional Cumulative Distribution Functions

In the sequel we will be making use of the concept of a conditional CDF for some subset of variables $\mathbf{X}_A$ conditioned on event M. We formally define the conditional CDF below.

Definition 2 *Let M be an event with $\mathbb{P}[M] > 0$. The conditional CDF $F(\mathbf{x}_A \mid M)$ conditioned on event M is defined as*

$$F(\mathbf{x}_A \mid M) \equiv \mathbb{P}\Big[\mathbf{X}_A \leq \mathbf{x}_A \mid M\Big] = \frac{\mathbb{P}\Big[\{\mathbf{X}_A \leq \mathbf{x}_A\} \cap M\Big]}{\mathbb{P}\big[M\big]}.$$

□

We will now find the above conditional CDF for different types of events M.

Lemma 3 *Let $F(\mathbf{x}_C)$ be a marginal CDF obtained from the joint CDF $F(\mathbf{x})$ as given by Proposition 1 for some $\mathbf{X}_C \subseteq \mathbf{X}$. Consider some variable set $\mathbf{X}_A \subseteq \mathbf{X}$ where $\mathbf{X}_A \bigcap \mathbf{X}_C = \emptyset$. Let $M = \omega(\mathbf{x}_C) \equiv \{\mathbf{X}_C \leq \mathbf{x}_C\}$ for $\mathbf{X}_C \subset \mathbf{X}$. If $F(\mathbf{x}_C) > 0$, then $F(\mathbf{x}_A | \omega(\mathbf{x}_C)) \equiv F(\mathbf{x}_A | \mathbf{X}_C \leq \mathbf{x}_C) = \dfrac{F(\mathbf{x}_A, \mathbf{x}_C)}{F(\mathbf{x}_C)}$.* □

Thus a conditional CDF of the form $F(\mathbf{x}_A|\omega(\mathbf{x}_C))$ can be obtained by taking ratios of joint CDFs, which consists of computing limits to obtain the required marginal CDFs. It follows from Lemma 3 that marginalization over variables $\mathbf{X}_C$ can be viewed as a special case of conditioning on $\mathbf{X}_C < \infty$.

To compute conditional CDFs of the form $F(\mathbf{x}_A|x_\beta)$ where we instead condition on x_β, we need to *differentiate* the joint CDF, as we now show.

Lemma 4 *Consider some variable set $\mathbf{X}_A \subseteq \mathbf{X}$. Let $M = \{x_\beta < X_\beta \leq x_\beta + \varepsilon\}$ with $\varepsilon > 0$ for some scalar random variable $X_\beta \notin \mathbf{X}_A$. If $F(x_\beta)$ and $F(\mathbf{x}_A, x_\beta)$ are differentiable with respect to x_β so that $\partial_{x_\beta}\Big[F(x_\beta)\Big]$ and $\partial_{x_\beta}\Big[F(\mathbf{x}_A, x_\beta)\Big]$ exist with $\partial_{x_\beta}\Big[F(x_\beta)\Big] > 0$, then the conditional CDF $F(\mathbf{x}_A|x_\beta) \equiv \lim_{\varepsilon\to 0^+} F(\mathbf{x}_A|x_\beta < X_\beta < x_\beta + \varepsilon) = \lim_{\varepsilon\to 0^+} \dfrac{\mathbb{P}\Big[\{\mathbf{X}_A \leq \mathbf{x}_A\} \cap \{x_\beta < X_\beta \leq x_\beta + \varepsilon\}\Big]}{\mathbb{P}\Big[x_\beta < X_\beta \leq x_\beta + \varepsilon\Big]}$ is given by*

$$F(\mathbf{x}_A|x_\beta) = \frac{\partial_{x_\beta}\Big[F(\mathbf{x}_A, x_\beta)\Big]}{\partial_{x_\beta}\Big[F(x_\beta)\Big]} \propto \partial_{x_\beta}\Big[F(\mathbf{x}_A, x_\beta)\Big].$$

Proof We can write

$$\begin{aligned} F(\mathbf{x}_A|x_\beta < X_\beta \leq x_\beta + \varepsilon) &= \frac{\mathbb{P}\Big[\{\mathbf{X}_A \leq \mathbf{x}_A\} \cap \{x_\beta < X_\beta \leq x_\beta + \varepsilon\}\Big]}{\mathbb{P}\Big[x_\beta < X_\beta \leq x_\beta + \varepsilon\Big]} \\ &= \frac{\frac{1}{\varepsilon}\mathbb{P}\Big[\{\mathbf{X}_A \leq \mathbf{x}_A\} \cap \{x_\beta < X_\beta \leq x_\beta + \varepsilon\}\Big]}{\frac{1}{\varepsilon}\mathbb{P}\Big[x_\beta < X_\beta \leq x_\beta + \varepsilon\Big]} = \frac{\frac{F(\mathbf{x}_A, x_\beta+\varepsilon) - F(\mathbf{x}_A, x_\beta)}{\varepsilon}}{\frac{F(x_\beta+\varepsilon) - F(x_\beta)}{\varepsilon}}. \end{aligned}$$

Taking limits, and given differentiability of both $F(x_\beta)$ and $F(\mathbf{x}_A, x_\beta)$ with respect to x_β, the conditional CDF $F(\mathbf{x}_A|x_\beta)$ is given by

$$F(\mathbf{x}_A|x_\beta) \equiv \frac{\lim\limits_{\varepsilon\to 0^+} \dfrac{F(\mathbf{x}_A, x_\beta+\varepsilon) - F(\mathbf{x}_A, x_\beta)}{\varepsilon}}{\lim\limits_{\varepsilon\to 0^+} \dfrac{F(x_\beta+\varepsilon) - F(x_\beta)}{\varepsilon}} = \frac{\partial_{x_\beta}\Big[F(\mathbf{x}_A, x_\beta)\Big]}{\partial_{x_\beta}\Big[F(x_\beta)\Big]} \propto \partial_{x_\beta}\Big[F(\mathbf{x}_A, x_\beta)\Big],$$

where the proportionality constant does not depend on $\mathbf{x}_A$. ■

The generalization of the above lemma to conditioning on sets of variables $\mathbf{X}_C \subseteq \mathbf{X}$ can be found in the Appendix.

2. Cumulative Distribution Networks

Graphical models allow us to simplify the computations required for obtaining conditional probabilities of the form $P(\mathbf{x}_A|\mathbf{x}_B)$ or $P(\mathbf{x}_A)$ by allowing us to model conditional independence constraints in terms of graph separation constraints. However, for many applications it may be desirable to compute other conditional and marginal probabilities such as probabilities of events of the type $\{\mathbf{X} \leq \mathbf{x}\}$. Here we will present the cumulative distribution network (CDN), which is a graphical

framework for directly modeling the joint cumulative distribution function, or CDF. With the CDN, we can thus expand the set of possible probability queries so that in addition to formulating queries as conditional/marginal probabilities of the form $P(\mathbf{x}_A)$ and $P(\mathbf{x}_A|\mathbf{x}_B)$, we can also compute probabilities of the form $F(\mathbf{x}_A|\omega(\mathbf{x}_B)), F(\mathbf{x}_A|\mathbf{x}_B), P(\mathbf{x}_A|\omega(\mathbf{x}_B))$ and $F(\mathbf{x}_A)$, where $F(\mathbf{u}) \equiv \mathbb{P}\Big[\mathbf{U} \leq \mathbf{u}\Big]$ is a CDF and we denote the inequality event $\{\mathbf{U} \leq \mathbf{u}\}$ using $\omega(\mathbf{x}_U)$. Examples of this new type of query could be "Given that the drug dose was less than 1 mg, what is the probability of the patient living at least another year?", or "Given that a person prefers one brand of soda over another, what is the probability of that person preferring one type of chocolate over another?". A significant advantage with CDNs is that the graphical representation of the joint CDF may naturally allow for queries which would otherwise be difficult, if not intractable, to compute under directed, undirected and factor graphical models for PDFs/PMFs.

Here we provide a formal definition of the CDN and we will show that the conditional independence properties in such graphical models are distinct from the properties for directed, undirected and factor graphs. We will then show that the conditional independence properties in CDNs include the properties of bi-directed graphs (Drton and Richardson, 2008; Richardson, 2003). Finally, we will show that CDNs provide a tractable means of parameterizing models for learning to rank in which we can construct multivariate CDFs from a product of CDFs defined over subsets of variables.

Definition 5 *The cumulative distribution network (CDN) is an undirected bipartite graphical model consisting of a bipartite graph $\mathcal{G} = (V, S, E)$, where V denotes variable nodes and S denotes factor nodes, with edges in E connecting factor nodes to variable nodes. The CDN also includes a specification of functions $\phi_s(\mathbf{x}_s)$ for each function node $s \in S$, where $\mathbf{x}_s \equiv \mathbf{x}_{\mathcal{N}(s)}$, $\cup_{s \in S} \mathcal{N}(s) = V$ and each function $\phi_s : \mathbb{R}^{|\mathcal{N}(s)|} \mapsto [0,\ 1]$ satisfies the properties of a CDF. The joint CDF over the variables in the CDN is then given by the product over CDFs $\phi_s : \mathbb{R}^{|\mathcal{N}(s)|} \mapsto [0,\ 1]$, or*

$$F(\mathbf{x}) = \prod_{s \in S} \phi_s(\mathbf{x}_s),$$

where each CDF ϕ_s is defined over neighboring variable nodes $\mathcal{N}(s)$. □

An example of a CDN defined over three variable nodes with four CDN function nodes is shown in Figure 1, where the joint CDF over three variables X, Y, Z is given by

$$F(x, y, z) = \phi_a(x, y)\phi_b(x, y, z)\phi_c(y, z)\phi_d(z).$$

In the CDN, each function node (depicted as a diamond) corresponds to one of the functions $\phi_s(\mathbf{x}_s)$ in the model for the joint CDF $F(\mathbf{x})$. Thus, one can think of the CDN as a factor graph for modeling the joint CDF instead of the joint PDF. However, as we will see shortly, this leads to a different set of conditional independence properties as compared to the conditional independence properties of directed, undirected and factor graphs.

Since the CDN is a graphical model for the joint CDF, the functions in the CDN must be such that $F(\mathbf{x})$ is a CDF for some probability $\mathbb{P}$. The following lemma establishes a sufficient condition that the CDN functions ϕ_s be themselves CDFs in order for F to be a CDF.

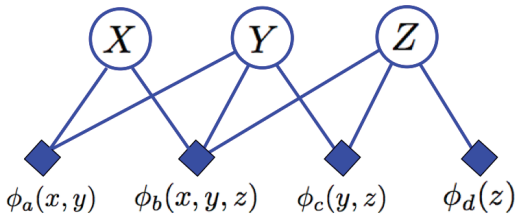


Figure 1: A cumulative distribution network (CDN) defined over three variables and four functions.

Lemma 6 *If all functions $\phi_s(\mathbf{x}_s)$ satisfy the properties of a CDF, then the product $\prod_{s\in S}\phi_s(\mathbf{x}_s)$ also satisfies the properties of a CDF.* □

Proof If for all $s \in S$, we have $\lim_{\mathbf{x}_s\to\infty}\phi_s(\mathbf{x}_s) = 1$, then $\lim_{\mathbf{x}\to\infty}\prod_{s\in S}\phi_s(\mathbf{x}_s) = 1$. Furthermore, if for any given $\alpha \in V$ and for $s \in \mathcal{N}(\alpha)$, we have $\lim_{x_\alpha\to-\infty}\phi_s(\mathbf{x}_s) = 0$, then $\lim_{x_\alpha\to-\infty}\prod_{s\in S}\phi_s(\mathbf{x}_s) = 0$.

To show that the product of monotonically non-decreasing functions is monotonically non-decreasing, we note that $\mathbf{x}_s < \mathbf{y}_s$ for all $s \in S$ if and only if $\mathbf{x} < \mathbf{y}$, since $\cup_{s\in S}\mathcal{N}(s) = V$. Thus if we have $\phi_s(\mathbf{x}_s) \le \phi_s(\mathbf{y}_s)\ \forall\ \mathbf{x}_s < \mathbf{y}_s$ for all $s \in S$, we can then write

$$F(\mathbf{x}) = \prod_{s\in S}\phi_s(\mathbf{x}_s) \le \prod_{s\in S}\phi_s(\mathbf{y}_s) = F(\mathbf{y}).$$

Finally, a product of right-continuous functions is also right-continuous. Thus if all of the functions $\phi_s(\mathbf{x}_s)$ satisfy the properties of a CDF, then the product of such functions also satisfies the properties of a CDF. ■

Although the condition that each of the ϕ_s functions be a CDF is sufficient for the overall product to satisfy the properties of a CDF, we emphasize that it is not a necessary condition, as one could construct a function that satisfies the properties of a CDF from a product of functions that are not CDFs. The sufficient condition above ensures, however, that we can construct CDNs by multiplying together CDFs to obtain another CDF. Furthermore, the above definition and theorem do not assume differentiability of the joint CDF or of the CDN functions: the following proposition shows that differentiability and non-negativity of the derivatives of functions ϕ_s with respect to all neighboring variables in $\mathcal{N}(s)$ imply both differentiability and monotonicity of the joint CDF $F(\mathbf{x})$. In the sequel we will assume that whenever CDN functions are differentiable, derivatives are invariant to the order in which they are computed (Schwarz' Theorem).

Proposition 7 *If the mixed derivatives $\partial_{\mathbf{x}_A}\left[\phi_s(\mathbf{x}_s)\right]$ satisfy $\partial_{\mathbf{x}_A}\left[\phi_s(\mathbf{x}_s)\right] \ge 0$ for all $s \in S$ and $A \subseteq \mathcal{N}(s)$, then*

- *$\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right] \ge 0$ for all $C \subseteq V$,*
- *$F(\mathbf{x}) \le F(\mathbf{y})$ for all $\mathbf{x} < \mathbf{y}$,*
- *$F(\mathbf{x})$ is differentiable.*

□

Proof A product of differentiable functions is differentiable and so $F(\mathbf{x})$ is differentiable. To show that $\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right] \geq 0 \; \forall \, C \subseteq V$, we can group the functions $\phi_s(\mathbf{x}_s)$ arbitrarily into two functions $g(\mathbf{x})$ and $h(\mathbf{x})$ so that $F(\mathbf{x}) = g(\mathbf{x})h(\mathbf{x})$. The goal here will be to show that if all derivatives $\partial_{\mathbf{x}_A}\left[g(\mathbf{x})\right]$ and $\partial_{\mathbf{x}_A}\left[h(\mathbf{x})\right]$ are non-negative, then $\partial_{\mathbf{x}_A}\left[F(\mathbf{x})\right]$ must also be non-negative. For all $C \subseteq V$, applying the product rule to $F(\mathbf{x}) = g(\mathbf{x})h(\mathbf{x})$ yields

$$\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right] = \sum_{A \subseteq C} \partial_{\mathbf{x}_A}\left[g(\mathbf{x})\right] \partial_{\mathbf{x}_{C \setminus A}}\left[h(\mathbf{x})\right],$$

so if $\partial_{\mathbf{x}_A}\left[g(\mathbf{x})\right], \partial_{\mathbf{x}_{C \setminus A}}\left[h(\mathbf{x})\right] \geq 0$ for all $A \subseteq C$ then $\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right] \geq 0$. By recursively applying this rule to each of the functions $g(\mathbf{x}), h(\mathbf{x})$ until we obtain sums over terms involving $\partial_{\mathbf{x}_A}\left[\phi_s(\mathbf{x}_s)\right] \; \forall \, A \subseteq \mathcal{N}(s)$, we see that if $\partial_{\mathbf{x}_A}\left[\phi_s(\mathbf{x}_s)\right] \geq 0$, then $\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right] \geq 0 \; \forall \, C \subseteq V$.

Now, $\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right] \geq 0$ for all $C \subseteq V$ implies that $\partial_{x_\alpha}\left[F(\mathbf{x})\right] \geq 0$ for all $\alpha \in V$. By the Mean Value Theorem for functions of several variables, it then follows that if $\mathbf{x} < \mathbf{y}$, then

$$F(\mathbf{y}) - F(\mathbf{x}) = \sum_{\alpha \in V} \partial_{z_\alpha}\left[F(\mathbf{z})\right](y_\alpha - x_\alpha) \geq 0,$$

and so $F(\mathbf{x})$ is monotonic. ■

The above ensures differentiability and monotonicity of the joint CDF through constraining the derivatives of each of the CDN functions. We note that although it is merely sufficient for the first-order derivatives to be non-negative in order for $F(\mathbf{x})$ to be monotonic, the condition that the higher-order mixed derivatives $\partial_{\mathbf{x}_C}\left[F(\mathbf{x})\right]$ of the functions $\phi_s(\mathbf{x}_s)$ be non-negative also implies non-negativity of the first-order derivatives. Thus in the sequel, whenever we assume differentiability of CDN functions, we will assume that for all $s \in S$, all mixed derivatives of $\phi_s(\mathbf{x}_s)$ with respect to any and all subsets of argument variables are non-negative.

Having described the above conditions on CDN functions, we will now provide some examples of CDNs constructed from a product of CDFs.

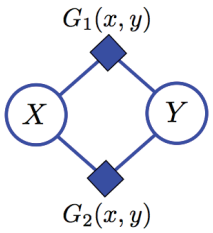
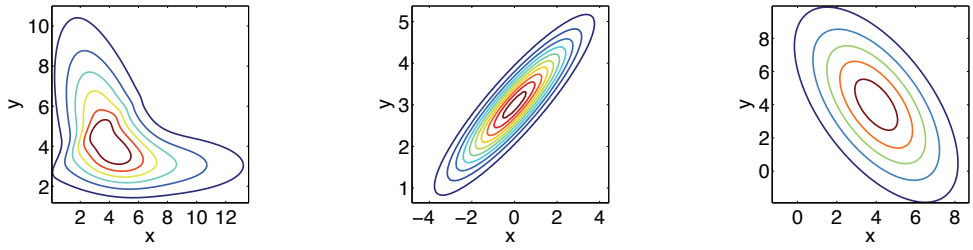


Figure 2: A CDN defined over two variables X and Y with functions $G_1(x,y), G_2(x,y)$.

Example 1 (Product of bivariate Gaussian CDFs) *As a simple example of a CDN, consider two random variables X and Y with joint CDF modeled by the CDN in Figure 2, so that $F(x,y) =$*



(a) (b) (c)

Figure 3: a) Joint probability density function $P(x,y)$ corresponding to the distribution function $F(x,y)$ using bivariate Gaussian CDFs as CDN functions; b),c) The PDFs corresponding to $\partial_{x,y}\Big[G_1(x,y)\Big]$ and $\partial_{x,y}\Big[G_2(x,y)\Big]$.

$G_1(x,y)G_2(x,y)$ *with*

$$G_1(x,y) = \Phi\left(\begin{bmatrix}x\\y\end{bmatrix};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1\right), \quad \boldsymbol{\mu}_1 = \begin{bmatrix}\mu_{x,1}\\\mu_{y,1}\end{bmatrix}, \qquad \boldsymbol{\Sigma}_1 = \begin{bmatrix}\sigma^2_{x,1} & \rho_1\sigma_{x,1}\sigma_{y,1}\\\rho_1\sigma_{x,1}\sigma_{y,1} & \sigma^2_{y,1}\end{bmatrix},$$
$$G_2(x,y) = \Phi\left(\begin{bmatrix}x\\y\end{bmatrix};\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2\right), \quad \boldsymbol{\mu}_2 = \begin{bmatrix}\mu_{x,2}\\\mu_{y,2}\end{bmatrix}, \qquad \boldsymbol{\Sigma}_2 = \begin{bmatrix}\sigma^2_{x,2} & \rho_2\sigma_{x,2}\sigma_{y,2}\\\rho_2\sigma_{x,2}\sigma_{y,2} & \sigma^2_{y,2}\end{bmatrix},$$

where $\Phi(\cdot;\mathbf{m},\mathbf{S})$ *is the multivariate Gaussian CDF with mean vector* $\mathbf{m}$ *and covariance* $\mathbf{S}$. *Taking derivatives, the density* $P(x,y)$ *is given by*

$$\begin{aligned}P(x,y) &= \partial_{x,y}\Big[F(x,y)\Big] = \partial_{x,y}\Big[G_1(x,y)G_2(x,y)\Big]\\ &= G_1(x,y)\partial_{x,y}\Big[G_2(x,y)\Big] + \partial_x\Big[G_1(x,y)\Big]\partial_y\Big[G_2(x,y)\Big]\\ &+ \partial_y\Big[G_1(x,y)\Big]\partial_x\Big[G_2(x,y)\Big] + \partial_{x,y}\Big[G_1(x,y)\Big]G_2(x,y).\end{aligned}$$

As functions G_1, G_2 *are Gaussian CDFs, the above derivatives can be expressed in terms of Gaussian CDF and PDFs. For example,*

$$\begin{aligned}\partial_x\Big[G_1(x,y)\Big] &= \int_{-\infty}^{y} Gaussian\left(\begin{bmatrix}x\\t\end{bmatrix};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1\right)dt\\ &= Gaussian(x;\mu_{x,1},\sigma^2_{x,1})\int_{-\infty}^{y} Gaussian(t;\mu_{y|x,1},\sigma^2_{y|x,1})\,dt\\ &= Gaussian(x;\mu_{x,1},\sigma^2_{x,1})\Phi(y;\mu_{y|x,1},\sigma^2_{y|x,1}),\end{aligned}$$

where

$$\mu_{y|x,1} = \mu_{y,1} + \rho_1\frac{\sigma_{y,1}}{\sigma_{x,1}}(x-\mu_{x,1}),$$
$$\sigma^2_{y|x,1} = (1-\rho_1^2)\sigma^2_{y,1}.$$

Other derivatives can be obtained similarly. The resulting joint PDF $P(x,y)$ obtained by differentiating the CDF is shown in Figure 3(a), where the CDN function parameters are given by $\mu_{x,1}=0,\mu_{x,2}=4,\mu_{y,1}=3,\mu_{y,2}=4,\sigma_{x,1}=\sqrt{3},\sigma_{x,2}=\sqrt{5},\sigma_{y,1}=1,\sigma_{y,2}=\sqrt{10},\rho_1=0.9,\rho_2=-0.6$. The PDFs corresponding to $\partial_{x,y}\Big[G_1(x,y)\Big]$ and $\partial_{x,y}\Big[G_2(x,y)\Big]$ are shown in Figures 3(b) and 3(c). □

The next example provides an illustration of the use of copula functions for constructing multivariate CDFs under the framework of CDNs.

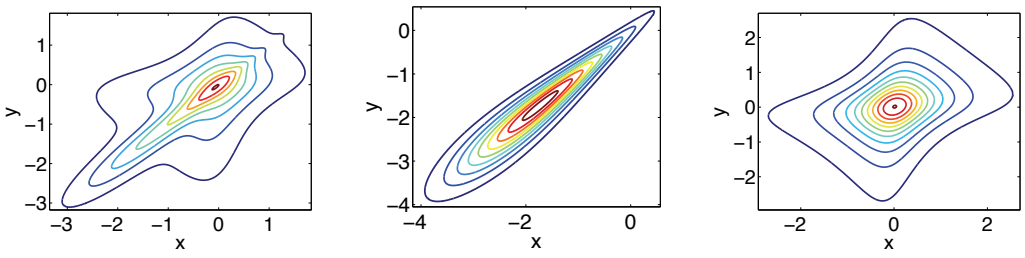


Figure 4: a) Joint probability density function $P(x,y)$ corresponding to the distribution function $F(x,y)$ using bivariate Gumbel copulas as CDN functions, with Student's-t and Gaussian marginal input CDFs; b),c) The PDFs corresponding to $\partial_{x,y}\Big[G_1(x,y)\Big]$ and $\partial_{x,y}\Big[G_2(x,y)\Big]$.

Example 2 (Product of copulas) *We can repeat the above for the case where each CDN function consists of a copula function (Nelsen, 1999). Copula functions provide a flexible means to construct CDN functions ϕ_s whose product yields a joint CDF under Lemma 6. Copula functions allow one to construct a multivariate CDF ϕ_s from marginal CDFs $\{F(x_\alpha)\}_{\alpha\in\mathcal{N}(s)}$ so that*

$$\phi_s(\mathbf{x}_s)=\zeta_s\Big(\{F(x_\alpha)\}_{\alpha\in\mathcal{N}(s)}\Big),$$

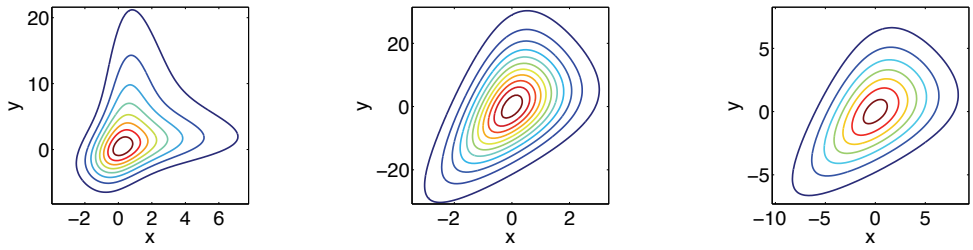
where ζ_s is a copula defined over variables $X_\alpha,\alpha\in\mathcal{N}(s)$. For the CDN shown in Figure 2, we can set the CDN functions G_1,G_2 to Gumbel copulas so that

$$G_1(x,y)=\zeta_1(H_{1,x}(x),H_{1,y}(y))=\exp\left(-\Big((-\log H_{1,x}(x))^{\frac{1}{\theta_1}}+(-\log H_{1,y}(y))^{\frac{1}{\theta_1}}\Big)^{\theta_1}\right),$$

$$G_2(x,y)=\zeta_2(H_{2,x}(x),H_{2,y}(y))=\exp\left(-\Big((-\log H_{2,x}(x))^{\frac{1}{\theta_2}}+(-\log H_{2,y}(y))^{\frac{1}{\theta_2}}\Big)^{\theta_2}\right),$$

with $H_{1,x},H_{2,x}$ set to univariate Gaussian CDFs with parameters $\mu_{1,x},\mu_{2,x},\sigma_{1,x},\sigma_{2,x}$ and $H_{1,y},H_{2,y}$ set to univariate Student's-t CDFs with parameters $\nu_{1,y},\nu_{2,y}$. One can then verify that the functions G_1,G_2 satisfy the properties of a copula function (Nelsen, 1999) and so the product of G_1,G_2 yields the CDF $F(x,y)$. An example of the resulting joint probability density $P(x,y)$ obtained by differentiation of $F(x,y)$ for parameters $\mu_{1,x}=\mu_{2,x}=-2,\sigma_{1,x}=\sigma_{2,x}=1,\nu_{1,y}=\nu_{2,y}=0.5,\theta_1=0.25,\theta_2=0.5$

is shown in Figure 4(a), with the PDFs corresponding to $\partial_{x,y}\Big[G_1(x,y)\Big]$ *and* $\partial_{x,y}\Big[G_2(x,y)\Big]$ *shown in Figures 4(b) and 4(c).* □



(a) (b) (c)

Figure 5: a) Joint probability density function $P(x,y)$ corresponding to the distribution function $F(x,y)$ using bivariate sigmoidal functions as CDN functions; b),c) The PDFs corresponding to $\partial_{x,y}\Big[G_1(x,y)\Big]$ and $\partial_{x,y}\Big[G_2(x,y)\Big]$.

Example 3 (Product of bivariate sigmoids) *As another example of a probability density function constructed using a CDN, consider the case in which functions* $G_1(x,y)$ *and* $G_1(x,y)$ *in the CDN of Figure 2 are set to be multivariate sigmoids of the form*

$$G_1(x,y) = \frac{1}{1+\exp(-w_x^1 x)+\exp(-w_y^1 y)},$$
$$G_2(x,y) = \frac{1}{1+\exp(-w_x^2 x)+\exp(-w_y^2 y)},$$

with $w_x^1, w_y^1, w_x^2, w_y^2$ *non-negative. An example of the resulting joint probability density* $P(x,y)$ *obtained by differentiation of* $F(x,y) = G_1(x,y)G_2(x,y)$ *for parameters* $w_x^1 = 12.5, w_y^1 = 0.125, w_x^2 = 0.4, w_y^2 = 0.5$ *is shown in Figure 5(a), with the PDFs corresponding to* $\partial_{x,y}\Big[G_1(x,y)\Big]$ *and* $\partial_{x,y}\Big[G_2(x,y)\Big]$ *shown in Figures 5(b) and 5(c).* □

The above examples demonstrate that one can construct multivariate CDFs by taking a product of CDFs defined over subsets of variables in the graph.

2.1 Conditional and Marginal Independence Properties of CDNs

In this section, we will derive the marginal and conditional independence properties for a CDN, which we show to be distinct from those of Bayesian networks, Markov random fields or factor graphs. As with these graphical models, marginal and conditional independence relationships can be gleaned by inspecting whether variables are separated with respect to the graph. In a bipartite graph $\mathcal{G} = (V,S,E)$, a (undirected) path of length K between two variable nodes $\alpha, \beta \in V$ consists of a sequence of distinct variable and function nodes $\alpha_0, s_0, \alpha_1, s_1, \cdots, s_K, \alpha_K$ such that $\alpha_0 = \alpha, \alpha_K = \beta$ and $(\alpha_k, s_k) = (s_k, \alpha_k) \in E$ for all $k = 0, \cdots, K$. A set $C \subseteq V$ is said to separate two variable nodes

$\alpha, \beta \in V \setminus C$ with respect to $\mathcal{G}$ if all paths from α to β intersect C. Two variable nodes $\alpha, \beta \in V$ are said to be separated if there exists any non-empty set C that separates them with respect to $\mathcal{G}$. Similarly, a set $C \subseteq V$ is said to separate two variable node sets $A, B \subseteq V \setminus C$ with respect to $\mathcal{G}$ if all paths from any variable node $\alpha \in A$ to any variable node $\beta \in B$ intersect C. Disjoint variable sets $A, B \in V$ are said to be separated if all pairs of nodes (α, β) for $\alpha \in A, \beta \in B$ are separated.

Having defined graph separation for bipartite graphs, we begin with the conditional inequality independence property of CDNs, from which other marginal and conditional independence properties for a CDN will follow.

Theorem 8 (Conditional inequality independence in CDNs) *Let $\mathcal{G} = (V, S, E)$ be a CDN and let $A, B \subseteq V$ be disjoint sets of variable nodes. If A and B are separated with respect to $\mathcal{G}$, then for any $W \subseteq V \setminus (A \cup B)$ $A \perp\!\!\!\perp B | \omega(\mathbf{x}_W)$ where $\omega(\mathbf{x}_W) \equiv \{\mathbf{X}_W \leq \mathbf{x}_W\}$.*

Proof If A and B are separated with respect to $\mathcal{G}$, then we can write

$$F(\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_{V \setminus (A \cup B)}) = g(\mathbf{x}_A, \mathbf{x}_{V \setminus (A \cup B)}) h(\mathbf{x}_B, \mathbf{x}_{V \setminus (A \cup B)})$$

for some functions g, h that satisfy the conditions of Lemma 6. This means that $F(\mathbf{x}_A, \mathbf{x}_B | \omega(\mathbf{x}_W))$ is given by

$$\begin{aligned} F(\mathbf{x}_A, \mathbf{x}_B | \omega(\mathbf{x}_W)) &= \frac{\lim_{\mathbf{x}_{V \setminus (A \cup B \cup W)} \to \infty} F(\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_{V \setminus (A \cup B)})}{\lim_{\mathbf{x}_{V \setminus W} \to \infty} F(\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_{V \setminus (A \cup B)})} \\ &\propto F(\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_W) = g(\mathbf{x}_A, \mathbf{x}_W) h(\mathbf{x}_B, \mathbf{x}_W), \end{aligned}$$

which implies $A \perp\!\!\!\perp B | \omega(\mathbf{x}_W)$. ■

We show that if a CDF $F(\mathbf{x})$ satisfies the conditional independence property of Theorem 8 for a given CDN, then F can be written as a product over functions $\phi_s(\mathbf{x}_s)$.

Theorem 9 (Factorization property of a CDN) *Let $\mathcal{G} = (V, S, E)$ be a bipartite graph and let the CDF $F(\mathbf{x})$ satisfy the conditional independence property implied by the CDN described by $\mathcal{G}$, so that graph separation of A and B by $V \setminus (A \cup B)$ with respect to $\mathcal{G}$ implies $A \perp\!\!\!\perp B | \omega(\mathbf{x}_W)$ for any $W \subseteq V \setminus (A \cup B)$ and for any $\mathbf{x}_W \in \mathbb{R}^{|W|}$. Then there exist functions $\phi_s(\mathbf{x}_s), s \in S$ that satisfy the properties of a CDF such that the joint CDF $F(\mathbf{x})$ factors as $\prod_{s \in S} \phi_s(\mathbf{x}_s)$.*

Proof The proof here parallels that for the Hammersley-Clifford theorem for undirected graphical models (Lauritzen, 1996). We begin our proof by defining $\psi_U(\mathbf{x}), \zeta_U(\mathbf{x})$ as functions that depend only on variable nodes in some set $U \subseteq V$ and that form a Möbius transform pair

$$\begin{aligned} \psi_U(\mathbf{x}) &= \sum_{W \subseteq U} \zeta_W(\mathbf{x}), \\ \zeta_U(\mathbf{x}) &= \sum_{W \subseteq U} (-1)^{|U \setminus W|} \psi_W(\mathbf{x}), \end{aligned}$$

where we take $\psi_U(\mathbf{x}) \equiv \log F(\mathbf{x}_U)$. Now, we note that $F(\mathbf{x})$ can always be written as a product of functions $\prod_{U \subseteq V} \phi_U(\mathbf{x})$ where each function ϕ_U satisfies the properties of a CDF: a trivial example

of this is to set $\phi_V(\mathbf{x}) = F(\mathbf{x})$ and $\phi_U(\mathbf{x}) = 1$ for all $U \subset V$. Since by hypothesis F satisfies all of the conditional independence properties implied by the CDN described by $\mathcal{G}$, if we take $\phi_U(\mathbf{x}) = \exp\big(\zeta_U(\mathbf{x})\big)$, then it suffices to show that $\zeta_U(\mathbf{x}) \equiv 0$ for subsets of variable nodes U for which any two non-neighboring variable nodes $\alpha, \beta \in U$ are separated such that $\alpha \perp\!\!\!\perp \beta | \omega(\mathbf{x}_W)$ for any $W \subseteq U \setminus (\alpha, \beta)$. Observe that we can write $\zeta_U(\mathbf{x})$ as

$$\begin{aligned}\zeta_U(\mathbf{x}) &= \sum_{W \subseteq U} (-1)^{|U \setminus W|} \psi_W(\mathbf{x}) \\ &= \sum_{W \subseteq U \setminus (\alpha \cup \beta)} (-1)^{|U \setminus W|} \Big(\psi_W(\mathbf{x}) - \psi_{W \cup \alpha}(\mathbf{x}) - \psi_{W \cup \beta}(\mathbf{x}) + \psi_{W \cup \alpha \cup \beta}(\mathbf{x}) \Big).\end{aligned}$$

If $\alpha, \beta \in U$ are separated and $W \subseteq U \setminus (\alpha \cup \beta)$, then $\alpha \perp\!\!\!\perp \beta | \omega(\mathbf{x}_W)$ and

$$\begin{aligned}\psi_{W \cup \alpha \cup \beta}(\mathbf{x}) - \psi_{W \cup \alpha}(\mathbf{x}) &= \log \frac{F(x_\alpha, x_\beta, \mathbf{x}_W)}{F(x_\alpha, \mathbf{x}_W)} = \log \frac{F(x_\alpha | \omega(\mathbf{x}_W)) F(x_\beta | \omega(\mathbf{x}_W)) F(\mathbf{x}_W)}{F(x_\alpha | \omega(\mathbf{x}_W)) F(\mathbf{x}_W)} \\ &= \log \frac{F(x_\beta | \omega(\mathbf{x}_W)) F(\mathbf{x}_W)}{F(\mathbf{x}_W)} \\ &= \log F(x_\beta, \mathbf{x}_W) - \log F(\mathbf{x}_W) \\ &= \psi_{W \cup \beta}(\mathbf{x}) - \psi_W(\mathbf{x}).\end{aligned}$$

Thus if U is any set where nodes $\alpha, \beta \in U$ are separated, then for all $W \subseteq U \setminus (\alpha \cup \beta)$ we must have $\psi_W(\mathbf{x}) - \psi_{W \cup \alpha}(\mathbf{x}) - \psi_{W \cup \beta}(\mathbf{x}) + \psi_{W \cup \alpha \cup \beta}(\mathbf{x}) \equiv 0$ and so $\zeta_U(\mathbf{x}) = 0$. Since $F(\mathbf{x}) = \exp(\psi_V(\mathbf{x})) = \exp\Big(\sum_U \zeta_U(\mathbf{x}) \Big) = \prod_U \phi_U(\mathbf{x})$ where the product is taken over subsets of variable nodes U that are not separated. Now, we note that for any U that is not separated, we must have $U \subseteq \mathcal{N}(s)$ (as $U = \mathcal{N}(s) \cup A$ for some A with $\mathcal{N}(s) \cap A = \emptyset$ implies that U is not separated) for some $s \in S$ and so we can write $F(\mathbf{x}) = \prod_U \phi_U(\mathbf{x}) = \prod_{s \in S} \prod_{U \subseteq \mathcal{N}(s)} \phi_U(\mathbf{x}) = \prod_{s \in S} \phi_s(\mathbf{x}_s)$, where $\phi_s(\mathbf{x}_s) = \prod_{U \subseteq \mathcal{N}(s)} \phi_U(\mathbf{x})$ satisfies the properties of a CDF given that functions $\phi_U(\mathbf{x})$ each satisfy the properties of a CDF. Thus we can write $F(\mathbf{x}) = \prod_{s \in S} \phi_s(\mathbf{x}_s)$, where each function ϕ_s is defined over the set of variable nodes $\mathcal{N}(s)$. ■

Thus, if $F(\mathbf{x})$ satisfies the conditional independence property where graph separation of A and B with respect to $\mathcal{G}$ implies $A \perp\!\!\!\perp B | \omega(\mathbf{x}_W)$ for any $W \subseteq V \setminus (A, B)$, then F can be written as a product of functions of the form $\prod_{s \in S} \phi_s(\mathbf{x}_s)$. The above theorem then demonstrates equivalence between the conditional independence property $A \perp\!\!\!\perp B | \omega(\mathbf{x}_W)$ and the factored form for $F(\mathbf{x})$.

The conditional inequality independence property for CDNs then implies that variables that are separated in the CDN are marginally independent. An example of the marginal independence property for a three-variable CDN in Figure 6, where variables X and Y are separated by variable Z with respect to graph $\mathcal{G}$, and so are marginally independent. In a CDN, variables that share no neighbors in the CDN graph are marginally independent: we formalize this with the following theorem.

Theorem 10 (Marginal Independence) *Let $\mathcal{G} = (V, S, E)$ be a CDN and let $A, B \subseteq V$ be disjoint sets of variables. Then $A \perp\!\!\!\perp B$ if $\mathcal{N}(A) \bigcap \mathcal{N}(B) = \emptyset$.* □

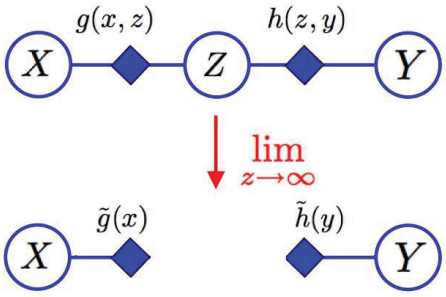
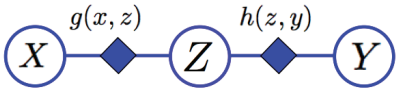


Figure 6: Marginal independence property of CDNs: if two variables X and Y share no common function nodes, they are marginally independent.

Proof Follows from Theorem 8 with $\mathbf{x}_W \to \infty$. ■

Note that the converse to the above does not generally hold: if disjoint sets A and B do share functions in S, they can still be marginally independent, as one can easily construct a bipartite graph in which variable nodes are not separated in the graph but the function nodes connecting A to B correspond to factorized functions so that $A \perp\!\!\!\perp B$. Given the above marginal independence property in a CDN, we now consider the conditional independence property of a CDN. To motivate this, we first present a toy example in Figure 7 in which we are given CDNs for variables X, Y, Z, W and we condition on variable Z. Here the separation of X and Y by unobserved variable W implies $X \perp\!\!\!\perp Y|Z$, but separation of X and Y by Z only implies the marginal independence relationship $X \perp\!\!\!\perp Y$. In general, variable sets that are separated in a CDN by unobserved variables will be conditionally independent given all other variables: thus, as long as two variables are separated by some unobserved variables they are independent, irrespective of the fact that other variables may be observed as well. We formalize this conditional independence property with the following theorem.



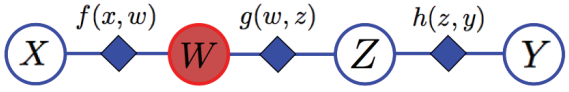


Figure 7: Conditional independence in CDNs. Two variables X and Y that are separated with respect to the graph are marginally independent (*top*). When an unobserved variable W (shaded to denote its unobserved status) separates X from Y, X, Y are conditionally independent given Z (*bottom*). The bottom graph thus implies $X \perp\!\!\!\perp Y, X \perp\!\!\!\perp Z$, $W \perp\!\!\!\perp Y$, $X \perp\!\!\!\perp Y|W$ and $X \perp\!\!\!\perp Y|Z$.

Theorem 11 (Conditional independence in CDNs) *Let $\mathcal{G} = (V,S,E)$ be a CDN. For all disjoint sets of $A,B,C \subseteq V$, if C separates A from B relative to graph $\mathcal{G}$ then*

$$A \perp\!\!\!\perp B | V \setminus (A \cup B \cup C).$$

□.

Proof If C separates A from B, then marginalizing out variables in C yields two disjoint subgraphs with variable sets A', B', with $A \subseteq A', B \subseteq B', A' \cup B' = V \setminus C$ and $\mathcal{N}(A') \bigcap \mathcal{N}(B') = \emptyset$. From Theorem 10, we therefore have $A' \perp\!\!\!\perp B'$. Now consider the set $V \setminus (A \cup B \cup C)$ and let $\tilde{A}, \tilde{B}$ denote a partition of the set so that

$$\begin{aligned} &\tilde{A} \cup \tilde{B} = V \setminus (A \cup B \cup C), \quad \tilde{A} \cap \tilde{B} = \emptyset, \\ &\tilde{A} \cap B' = \emptyset, \quad \tilde{B} \cap A' = \emptyset. \end{aligned}$$

From the semi-graphoid axioms (Lauritzen, 1996; Pearl, 1988), $A' \perp\!\!\!\perp B'$ implies $A \perp\!\!\!\perp B | V \setminus (A \cup B \cup C)$ since $\tilde{A} \subset A'$ and $\tilde{B} \subset B'$. ■

An illustration of the proof is provided in Figures 8(a) and 8(b). The above conditional independence property is distinct from that described in Theorem 8, as in the latter we condition on inequality events of the type $\omega(\mathbf{x}_W)$, whereas in the former we condition on observations $\mathbf{x}_W$ themselves.

In addition to the above, both the conditional independence properties of Theorem 11 and 8 are closed under marginalization, which consists of computing limits of CDN functions. Thus if $\mathcal{G}$ is a CDN model for $F(\mathbf{x})$, then the graph for CDN for CDF $F(\mathbf{x}_A) = \lim_{\mathbf{x}_{V \setminus A} \to \infty} F(\mathbf{x}_A, \mathbf{x}_{V \setminus A})$ is given by a subgraph of $\mathcal{G}$ which then implies only a subset of the independence properties of $\mathcal{G}$. The next proposition formalizes this.

Proposition 12 *Let $\mathcal{G} = (V,S,E)$ be a CDN and let $A,B,C \subset V$ be disjoint sets of nodes with $C \subseteq V \setminus (A \cup B)$ separating A from B with respect to $\mathcal{G}$. Let $\mathcal{G}' = (V',S',E')$ be a subgraph of $\mathcal{G}$ with $V' \subseteq V, S' \subseteq S, E' \subseteq E$. Similarly, let $A' = A \cap V', B' = B \cap V', C' = C \cap V'$ be disjoint sets of nodes. Then C' separates A' from B' with respect to $\mathcal{G}'$.* □

The above proposition is illustrated in Figures 9(a) and 9(b). As a result, the conditional independence relation $A' \perp\!\!\!\perp B' | V' \setminus (A' \cup B' \cup C')$ must also hold in the subgraph $\mathcal{G}'$, such that $\mathcal{G}'$ implies a subset of the independence constraints implied by $\mathcal{G}$. The above closure property under marginalization is a property that also holds for Markov random fields, but not for Bayesian networks (see Richardson and Spirtes, 2002 for an example). The above closure and conditional independence properties for CDNs have also been previously shown to hold for bi-directed graphs as well, which we will now describe.

2.2 The Relationship Between Cumulative Distribution Networks and Bi-directed Graphs

Graphical models with some of the independence properties of CDNs have in fact been studied previously in the statistics literature. The marginal independence property for CDNs is in fact identical to the global Markov property of Richardson and Spirtes (2002), which was derived in the context of bi-directed graphical models (Drton and Richardson, 2008; Richardson and Spirtes, 2002; Richardson, 2003). A bi-directed graph $G = (V,E)$ consists of nodes $\alpha \in V$ and *bi-directed* edges $e \in E$

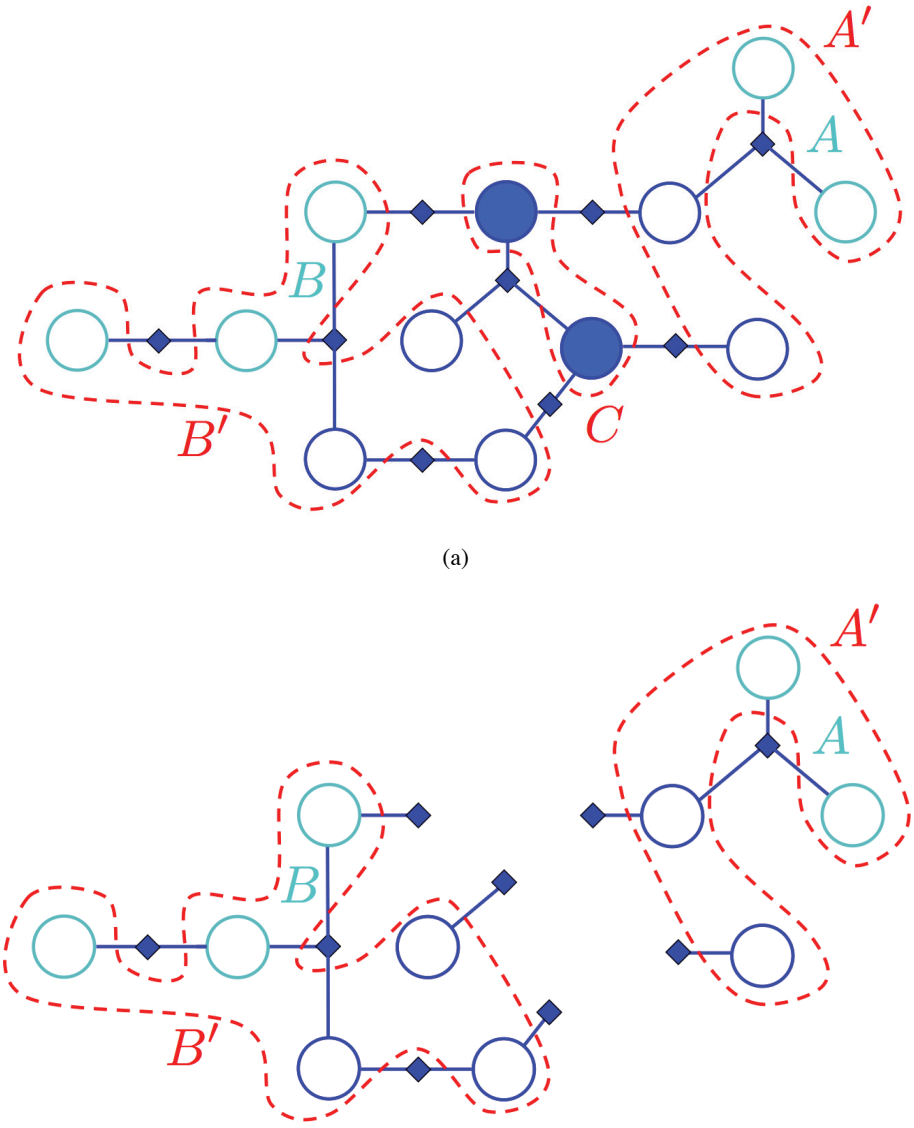


Figure 8: Example of conditional independence due to graph separation in a CDN. a) Given bipartite graph $\mathcal{G} = (V, S, E)$, node set C separates set A from B (nodes in light blue) with respect to $\mathcal{G}$. Furthermore, we have for A', B' (nodes in red dotted line) $A \subseteq A', B \subseteq B'$, $A' \cup B' = V \setminus C$ and $\mathcal{N}(A') \bigcap \mathcal{N}(B') = \emptyset$ as shown. b) Marginalizing out variables corresponding to nodes in C yields two disjoint subgraphs of $\mathcal{G}$ and so $A \perp\!\!\!\perp B | V \setminus (A \cup B \cup C)$.

corresponding to unordered pairs of nodes α, β, denoted by (α, β). Alternately, we denote edges in

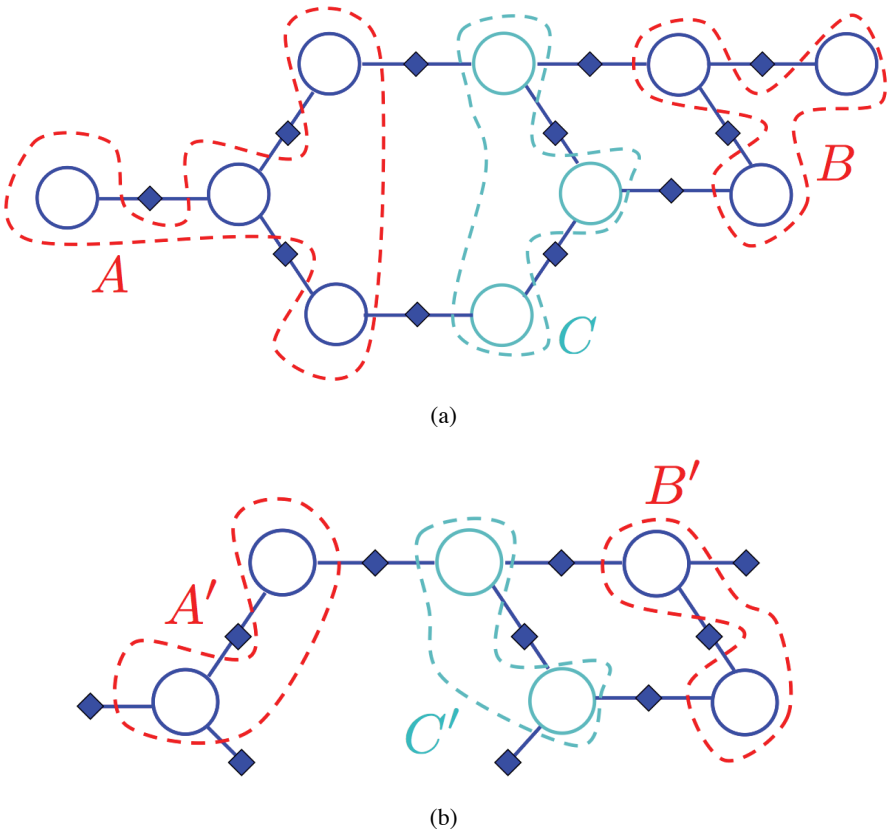


Figure 9: Example of closure under marginalization in a CDN. a) Given CDN $\mathcal{G} = (V,S,E)$, node set C separates set A from B (nodes in light blue) with respect to $\mathcal{G}$. b) For subgraph $\mathcal{G}' = (V',S',E')$ with $A' \subseteq A, B' \subseteq B, C' \subseteq C, C'$ separates A' from B' with respect to $\mathcal{G}'$.

a bi-directed graph as $(\alpha,\beta) \equiv \alpha \leftrightarrow \beta$.[1] In a bi-directed graph G, the *global Markov property* corresponds to two disjoint variable sets $A,B \subseteq V$ satisfying the marginal independence constraint $A \perp\!\!\!\perp B$ if there are no paths between any $\alpha \in A$ and any $\beta \in B$. It can be shown (Richardson and Spirtes, 2002) that any bi-directed graphical model corresponds to a directed graphical model with latent variables marginalized out. In particular, we define the *canonical* directed acyclic graph (DAG) for the bi-directed graph G as a directed graph $\tilde{G}$ with additional latent variables such that if $\alpha \leftrightarrow \beta$ in G, then $\alpha \leftarrow u_{\alpha,\beta} \rightarrow \beta$ in $\tilde{G}$ for some latent variable $u_{\alpha,\beta}$. Thus bi-directed graphical models can be viewed as models obtained from a corresponding canonical DAG with latent variables marginalized out, such that independence constraints between neighboring variable nodes in G can be viewed as arising from the absence of any shared latent variables in the canonical DAG $\tilde{G}$. This suggests the usefulness of bi-directed graphical models for problems where we cannot discount the presence of unobserved variables but we either A) do not have sufficient domain knowledge to specify distributions for latent variables, and/or B) we wish to avoid marginalizing over these latent variables.

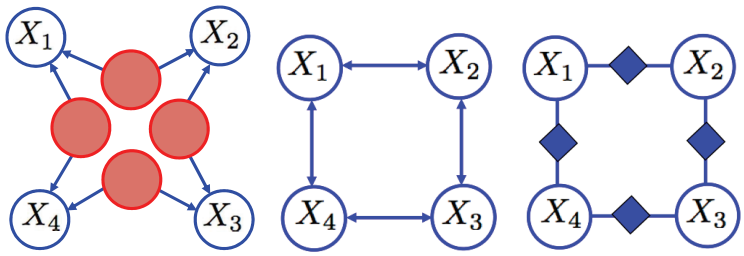
1. Note that $\alpha \leftrightarrow \beta$ is *not* equivalent to having both directed edges $\alpha \rightarrow \beta$ and $\alpha \leftarrow \beta$.

In such cases, one can instead attempt to parameterize a probability defined on observed variables using a bi-directed graphical model in which independence constraints among variables are implied by both the corresponding canonical DAG and bi-directed graphs. Examples of a canonical DAG and corresponding bi-directed graph that imply the same set of independence constraints among observed variables are shown in Figures 10(a) and 10(b). Several parameterizations had been previously proposed for bi-directed graphical models. Covariance graphs (Kauermann, 1996) were proposed in which variables are jointly Gaussian with zero pairwise covariance if there is no edge connecting the two variables in the bi-directed graph. In addition, Silva and Ghahramani (2009a) proposed a mixture model with latent variables in which dependent variables in the bi-directed graph can be explained by the causal influence of common components in the mixture model. For bi-directed graphical models defined over binary variables, a parametrization was proposed based on joint probabilities over connected components of the bi-directed graph so that the joint probability of any subset of variables could be obtained by Möbius inversion (Drton and Richardson, 2008).

Suppose now that we are given a bi-directed graph G and a CDN $\mathcal{G}$ defined over the same variables nodes V. Let G and $\mathcal{G}$ have the same connectivity, such that for any pair of variable nodes $\alpha, \beta \in V$, a path between α, β exists both in G and $\mathcal{G}$. Then both G and $\mathcal{G}$ imply the same set of marginal independence constraints, as we have shown above that in a CDN, two nodes that do not share any function nodes in common are marginally independent (Theorem 10). An example of a bi-directed graph and CDN that imply the same set of marginal independence constraints is shown in Figures 10(b) and 10(c). In addition to implying the same marginal independence constraints as a bi-directed graphical model, the conditional independence property given in Theorem 11 for CDNs corresponds to the dual global Markov property of Kauermann (1996) for bi-directed graphical models, which we now present.

Theorem 13 *Let $G = (V, E)$ be a bi-directed graphical model and let $A, B, C \subseteq V$ be three disjoint node sets so that $V \setminus (A \cup B \cup C)$ separates A from B with respect to G. Then $A \perp\!\!\!\perp B | C$.* □

Note that this is identical to the conditional independence property of Theorem 11 where the separating set is set to $V \setminus (A \cup B \cup C)$ instead of C.



(a) (b) (c)

Figure 10: Graphical models over four variables X_1, X_2, X_3, X_4 in which graph separation of variable nodes imply the marginal independence relations $X_1 \perp\!\!\!\perp X_3, X_2 \perp\!\!\!\perp X_4$. a) A canonical directed acyclic graphical model with additional latent variables, shown as shaded nodes; b)A bi-directed graph; b) A corresponding CDN.

While the conditional and marginal independence constraints implied by both a bi-directed graph and a CDN of the same connectivity are identical, Theorem 8 shows that conditional independence constraints of the form $A \perp\!\!\!\perp B|\omega(\mathbf{x}_W)$ are implied in a CDN which are not included in the definition for a bi-directed graph of the same connectivity. As a result of these *additional* constraints, CDNs model a subset of the distributions that satisfy the independence constraints of a corresponding bi-directed graph with the same connectivity. In general, CDNs do not model the full set of the probability distributions that can be modeled by bi-directed graphical models with the same connectivity. The following example illustrates how the conditional inequality independence property of CDNs is in general not implied by a bi-directed graphical model with the same graph topology.

Example 4 *Consider a 3-variable covariance graph model consisting of the bi-directed graph $X_1 \leftrightarrow X_2 \leftrightarrow X_3$, where X_1, X_2, X_3 are jointly Gaussian with zero mean and covariance matrix $\mathbf{\Sigma}$. The proposed covariance graph model imposes the marginal independence constraint $X_1 \perp\!\!\!\perp X_3$, as there is no edge between variables X_1, X_3. Denoting σ_{ij} as element (i, j) of $\mathbf{\Sigma}$, this is equivalent to the constraint $\sigma_{13} = \sigma_{31} = 0$. Now suppose further that the conditional inequality independence property $X_1 \perp\!\!\!\perp X_3|\omega(x_2)$ is also implied by the covariance graph model. By Theorem 9, this implies that the joint CDF $F(x_1, x_2, x_3)$ factors as*

$$F(x_1, x_2, x_3) = \Phi\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}; \mathbf{0}, \mathbf{\Sigma} \right) = g(x_1, x_2)h(x_2, x_3),$$

where $\Phi(\mathbf{x}; \boldsymbol{\mu}, \mathbf{\Sigma})$ is the multivariate Gaussian CDF with mean zero and covariance matrix $\mathbf{\Sigma}$, and $g(x_1, x_2), h(x_2, x_3)$ are functions that satisfy the properties of a CDF. The constraints on functions $g(x_1, x_2), h(x_2, x_3)$ are given by marginalization with respect to subsets of variables:

$$\begin{aligned} F(x_1, x_2, \infty) &= g(x_1, x_2)h(x_2, \infty), \\ F(\infty, x_2, x_3) &= g(\infty, x_2)h(x_2, x_3), \\ F(\infty, x_2, \infty) &= g(\infty, x_2)h(x_2, \infty), \end{aligned}$$

so that and so multiplying $F(x_1, x_2, \infty)$ and $F(\infty, x_2, x_3)$, we obtain

$$F(x_1, x_2, x_3)F(\infty, x_2, \infty) = F(x_1, x_2, \infty)F(\infty, x_2, x_3). \tag{1}$$

Thus, if the conditional inequality independence constraint $X_1 \perp\!\!\!\perp X_3|\omega(x_2)$ is also implied by the covariance graph model for the joint Gaussian CDF $F(x_1, x_2, x_3)$, then the above equality should hold for all $(x_1, x_2, x_3) \in \mathbb{R}^3$ and for any positive-definite covariance matrix $\mathbf{\Sigma}$ for which $\sigma_{13} = \sigma_{31} = 0$. Let $x_1 = x_2 = x_3 = 0$ and let $\mathbf{\Sigma}$ be given by

$$\mathbf{\Sigma} = \begin{bmatrix} 1 & \frac{\sqrt{2}}{2} & 0 \\ \frac{\sqrt{2}}{2} & 1 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 1 \end{bmatrix},$$

so that $\rho_{12} = \frac{\sqrt{2}}{2}, \rho_{23} = -\frac{1}{2}$ are the pairwise correlations between X_1, X_2 and X_2, X_3. From Stuart and Ord (1994), we can analytically evaluate joint Gaussian CDFs at the origin as a function of

correlation parameters, so that

$$
\begin{aligned}
F(\infty,0,\infty) &= \frac{1}{2},\\
F(0,0,\infty) &= \Phi\left(\begin{bmatrix}0\\0\end{bmatrix};\mathbf{0},\mathbf{\Sigma}_{12}\right) = \frac{1}{4}+\frac{1}{2\pi}\sin^{-1}\rho_{12} = \frac{3}{8},\\
F(\infty,0,0) &= \Phi\left(\begin{bmatrix}0\\0\end{bmatrix};\mathbf{0},\mathbf{\Sigma}_{23}\right) = \frac{1}{4}+\frac{1}{2\pi}\sin^{-1}\rho_{23} = \frac{1}{6},\\
F(0,0,0) &= \Phi\left(\begin{bmatrix}0\\0\\0\end{bmatrix};\mathbf{0},\mathbf{\Sigma}\right) = \frac{1}{8}+\frac{1}{4\pi}(\sin^{-1}\rho_{12}+\sin^{-1}\rho_{23}+\sin^{-1}\rho_{13})\\
&= \frac{1}{8}+\frac{1}{4\pi}(\sin^{-1}\rho_{12}+\sin^{-1}\rho_{23}) = \frac{7}{48},
\end{aligned}
$$

where $\mathbf{\Sigma}_{ij}$ is the sub-matrix consisting of rows and columns (i,j) in $\mathbf{\Sigma}$, ρ_{ij} is the correlation coefficient between variables i,j, and $\rho_{13}=0$ is implied by the covariance graph. From Equation (1)*, we must have*

$$
F(0,0,0)F(\infty,0,\infty) = F(0,0,\infty)F(\infty,0,0) \Leftrightarrow \frac{7\cdot 1}{48\cdot 2} = \frac{3\cdot 1}{8\cdot 6},
$$

so that the equality does not hold. Thus, the conditional inequality independence constraint $X_1 \perp\!\!\!\perp X_3|\omega(x_2)$ is not implied by the covariance graph model. It can also be verified that the expression for $F(x_1,x_2,x_3)$ given in Equation (1) *does not in general correspond to a proper PDF when $F(x_1,x_2),F(x_2,x_3),F(x_2)$ are Gaussians, as $\partial_{x_1,x_2,x_3}\left[F(x_1,x_2,x_3)\right]$ is not non-negative for all $(x_1,x_2,x_3)\in\mathbb{R}^3$.*

The previous example shows that while graph separation of variable node sets A,B with respect to both bi-directed graphical models and CDNs of the same connectivity implies the same set of marginal independence constraints, in CDNs we have the *additional* constraint of $A \perp\!\!\!\perp B|\omega(\mathbf{x}_C)$, a constraint that is not implied by the corresponding bi-directed graphical model. The above example shows how such additional constraints can then impose constraints on the joint probabilities that can be modeled by CDNs. However, for probabilities that can be modeled by any of CDN, bi-directed graph or corresponding canonical DAG models, CDNs can provide closed-form parameterizations where other types of probability models might not.

In the case of CDNs defined over discrete variables taking values in an ordered set $\mathcal{X} = \{r_1,\cdots,r_K\}$, the conditional independence property $A \perp\!\!\!\perp B|\omega(\mathbf{x}_W)$ for $W \subseteq V\setminus(A\cup B)$ (Theorem 8) implies that conditioning on the event $\mathbf{X}_C = r_1\mathbf{1}$ yields conditional independence between disjoint sets $A,B,C\subseteq V$ in which C separates A,B with respect to $\mathcal{G}$. We define the corresponding *min-independence* property below.

Definition 14 (Min-independence) *Let $\mathbf{X}_A,\mathbf{X}_B,\mathbf{X}_C$ be sets of ordinal discrete variables that take on values in the totally ordered alphabet $\mathcal{X}$ with minimum element $r_1\in\mathcal{X}$ defined as $r_1 \prec \alpha\ \forall\alpha\neq r_1,\alpha\in\mathcal{X}$. $\mathbf{X}_A$ and $\mathbf{X}_B$ are said to be min-independent given $\mathbf{X}_C$ if*

$$
\mathbf{X}_A \perp\!\!\!\perp \mathbf{X}_B|\mathbf{X}_C = r_1\mathbf{1},
$$

where $r_1\mathbf{1} = [r_1\ r_1\ \cdots\ r_1]^T$. □

Theorem 15 (Min-independence property of CDNs) *Let $\mathcal{G} = (V,S,E)$ be a CDN defined over ordinal discrete variables that take on values in the totally ordered alphabet $\mathcal{X}$ with minimum element $r_1 \in \mathcal{X}$ defined as $r_1 \prec \alpha \; \forall \alpha \neq r_1, \alpha \in \mathcal{X}$. Let $A,B,C \subseteq V$ be arbitrary disjoint subsets of V, with C separating A,B with respect to $\mathcal{G}$. Then $\mathbf{X}_A$ and $\mathbf{X}_B$ are min-independent given $\mathbf{X}_C$.* □

Proof This follows directly from Theorem 8 with $\mathbf{x}_c = r_1\mathbf{1}$. ■

Thus, in the case of a CDN defined over discrete variables where each variable can have values in the totally ordered alphabet $\mathcal{X}$, a finite difference with respect to variables $\mathbf{X}_C$, when evaluated at the vector of minimum elements $\mathbf{X}_C = r_1\mathbf{1}$ is equivalent to directly evaluating the CDF at $\mathbf{X}_C = r_1\mathbf{1}$. This means that in the case of models defined over ordinal discrete variables, the particular set of conditional independence relationships amongst variables in the model is determined as a function of the ordering over possible labels for each variable in the model, so that one must exercise care in how such variables are labeled and what ordering is satisfied by such labels.

2.3 Stochastic Orderings in a Cumulative Distribution Network

The CDN, as a graphical model for the joint CDF over many random variables, also allows one to easily specify *stochastic ordering* constraints between subsets of variables in the model. Informally, a stochastic ordering relationship $X \preceq Y$ holds between two random variables X,Y if samples of Y tend to be larger than samples of X. We will focus here on first-order stochastic ordering constraints (Lehmann, 1955; Shaked and Shanthikumar, 1994) of the form $X \preceq Y$ and how one can specify such constraints in terms of the CDN functions in the model. We note that such constraints are not a necessary part of the definition for a CDN or for a multivariate CDF, so that the graph for the CDN alone does not allow one to inspect stochastic ordering constraints based on graph separation of variables. However, the introduction of stochastic ordering constraints, in combination with separation of variables with respect to the graph, do impose constraints on the products of CDN functions, as we will now show. We will define below the concept of first-order stochastic orderings among random variables, as this is the primary definition for a stochastic ordering that we will use. We refer the reader to Lehmann (1955) and Shaked and Shanthikumar (1994) for additional definitions.

Definition 16 *Consider two scalar random variables X and Y with marginal CDFs $F_X(x)$ and $F_Y(y)$. Then X and Y are said to satisfy the first-order stochastic ordering constraint $X \preceq Y$ if $F_X(t) \geq F_Y(t)$ for all $t \in \mathbb{R}$.* □

The above definition of stochastic ordering is stronger than the constraint $\mathbb{E}[X] \leq \mathbb{E}[Y]$ which is often used and one can show that $X \preceq Y$ implies the former constraint. Note that the converse is not true: $\mathbb{E}[X] \leq \mathbb{E}[Y]$ does not necessarily imply $X \preceq Y$. For example, consider two Gaussian random variables X and Y for which $\mathbb{E}[X] \leq \mathbb{E}[Y]$ but $Var[X] \gg Var[Y]$. The definition of a stochastic ordering can also be extended to disjoint sets of variables $\mathbf{X}_A$ and $\mathbf{X}_B$.

Definition 17 *Let $\mathbf{X}_A$ and $\mathbf{X}_B$ be disjoint sets of variables so that $\mathbf{X}_A = \{X_{\alpha_1}, \cdots, X_{\alpha_K}\}$ and $\mathbf{X}_B = \{X_{\beta_1}, \cdots, X_{\beta_K}\}$ for some strictly positive integer K. Let $F_{\mathbf{X}_A}(\mathbf{t})$ and $F_{\mathbf{X}_B}(\mathbf{t})$ be the CDFs of $\mathbf{X}_A$ and $\mathbf{X}_B$. Then $\mathbf{X}_A, \mathbf{X}_B$ are said to satisfy the stochastic ordering relationship $\mathbf{X}_A \preceq \mathbf{X}_B$ if*

$$F_{\mathbf{X}_A}(\mathbf{t}) \geq F_{\mathbf{X}_B}(\mathbf{t})$$

for all $\mathbf{t} \in \mathbb{R}^K$. □

Having defined stochastic orderings, we will now present the corresponding constraints on CDN functions which are implied by the above definitions.

Proposition 18 *Let $\mathcal{G} = (V,S,E)$ be a CDN, with $A,B \subset V$ so that $A = \{\alpha_1,\cdots,\alpha_K\}$ and $B = \{\beta_1,\cdots,\beta_K\}$ for some strictly positive integer K. Let $\mathbf{t} \in \mathbb{R}^K$. Then A,B satisfy the stochastic ordering relationship $\mathbf{X}_A \preceq \mathbf{X}_B$ if and only if*

$$\prod_{s\in\mathcal{N}(A)} \lim_{\mathbf{u}_{\mathcal{N}(s)\setminus A}\to\infty} \phi_s(\mathbf{u}_{\mathcal{N}(s)\setminus A}, \mathbf{t}_{\mathcal{N}(s)\cap A}) \geq \prod_{s\in\mathcal{N}(B)} \lim_{\mathbf{u}_{\mathcal{N}(s)\setminus B}\to\infty} \phi_s(\mathbf{u}_{\mathcal{N}(s)\setminus B}, \mathbf{t}_{\mathcal{N}(s)\cap B})$$

for all $\mathbf{t} \in \mathbb{R}^K$. □

The above can be readily obtained by marginalizing over variables in $V \setminus A, V \setminus B$ respectively to obtain expressions for $F(\mathbf{x}_A), F(\mathbf{x}_B)$ as products of CDN functions. The corresponding ordering then holds from Definition 17 if and only if $F_{\mathbf{X}_A}(\mathbf{t}) \geq F_{\mathbf{X}_B}(\mathbf{t})$ for all $\mathbf{t} \in \mathbb{R}^K$.

2.4 Discussion

We have presented the CDN and sufficient conditions on the functions in the CDN in order for the CDN to model to a CDF. We have shown that the conditional independence relationships that follow from graph separation in CDNs are different from the relationships implied by graph separation in Bayesian networks, Markov random fields and factor graph models. We have shown that the conditional independence properties of CDNs include, but are not limited to, the marginal independence properties of bi-directed graphs, such that CDNs model a subset of all probability distributions that could be modeled by bi-directed graphs.

As we have shown, performing marginalization in a CDN consists of computing limits, unlike marginalization in models for probability densities. Furthermore, conditioning on observations in a CDN consists of computing derivatives. In the next section, we show how these two operations can be performed efficiently for tree-structured CDNs using message-passing, where messages being passed in the graph for the CDN correspond to mixed derivatives of the joint CDF with respect to variables in subtrees of the graph.

3. The Derivative-sum-product Algorithm

In the previous section, we showed that for a joint CDF, we could compute conditional probabilities of the forms $F(\mathbf{x}_A|\omega(\mathbf{x}_B)), F(\mathbf{x}_A|\mathbf{x}_B), P(\mathbf{x}_A|\omega(\mathbf{x}_B))$ and $P(\mathbf{x}_A|\mathbf{x}_B)$, in addition to probabilities of the type $P(\mathbf{x}_A), F(\mathbf{x}_A)$. In directed, undirected or factor graphs, computing and evaluating such conditional CDFs/PDFs would generally require us to integrate over several variables. In a CDN, computing and evaluating such conditionals corresponds to *differentiating* the joint CDF and then evaluating the total mixed derivative for any given vector of observations $\mathbf{x}$. In this section we will show that if we model the joint CDF using a CDN with a tree-structured graph, then we can derive a class of message-passing algorithms called *derivative-sum-product* (DSP) for efficiently computing and evaluating derivatives in CDNs. Since that the CDF factorizes for a CDN, the global mixed derivative can then be decomposed into a series of local mixed derivative computations, where each function $s \in S$ and its derivatives is evaluated for observations $\mathbf{x}_s$. Throughout this section, we will assume that the sufficient conditions for the CDN functions $\phi_s(\mathbf{x}_s)$ hold in order for the CDN to model a valid joint CDF (Lemma 6). We will further assume that the derivatives/ finite differences

of CDN functions $\phi_s(\mathbf{x}_s)$ with respect to all subsets of argument variables exist and that the order of differentiation does not affect the computation of any mixed derivatives. In the case where we are differentiating with respect to a set of variables $\mathbf{X}_C$ that are observed with values $\mathbf{x}_C$, we assume that the resulting derivative/finite difference is evaluated at the observed values $\mathbf{x}_C$. In the case where we are given a function $G(x)$ defined over a single ordinal discrete variable $x \in \mathcal{X}$ where $\mathcal{X} = \{r_0, r_1, \cdots, r_{N-1}\}$ and $r_0 < r_1 \cdots < r_{N-1}, r_i \in \mathbb{R}$ are N real-valued scalars, we define the finite difference of G with respect to x, evaluated at x as

$$\partial_x\Big[G(x)\Big] = \begin{cases} G(r_0) & \text{if } x = r_0, \\ G(r_i) - G(r_{i-1}) & \text{if } x = r_i,\ i = 1, \cdots, N-1. \end{cases}$$

3.1 Differentiation in Cumulative Distribution Networks

We first consider the problem of computing the marginal CDF $F(x_\alpha)$ for particular variable X_α. We note that in the CDN, marginalization corresponds to taking limits with respect to the variables in the model, so if we let

$$F(\mathbf{x}) = F(x_\alpha, \mathbf{x}_{V\setminus\alpha}) = \prod_{s\in\mathcal{N}(\alpha)} \phi_s(x_\alpha, \mathbf{x}_{\mathcal{N}(s)\setminus\alpha}) \prod_{s\notin\mathcal{N}(\alpha)} \phi_s(\mathbf{x}_s),$$

then the marginal CDF for X_α is given by

$$F(x_\alpha) = \lim_{\mathbf{x}_{V\setminus\alpha}\to\infty} F(x_\alpha, \mathbf{x}_{V\setminus\alpha}) = \prod_{s\in\mathcal{N}(\alpha)} \phi_s(x_\alpha, \infty) \prod_{s\notin\mathcal{N}(\alpha)} \phi_s(\infty) = \prod_{s\in\mathcal{N}(\alpha)} \phi_s(x_\alpha, \infty).$$

Thus for any x_α, we can obtain any distribution of the type $F(\mathbf{x}_A)$ in time $O(|S||V|)$ by taking the product of limits of functions $\lim_{\mathbf{x}_{\mathcal{N}(s)\setminus\alpha}\to\infty} \phi_s(x_\alpha, \mathbf{x}_{\mathcal{N}(s)\setminus\alpha}) = \phi_s(x_\alpha, \infty)$. Furthermore, we can compute any conditional cumulative distribution of the type $F(\mathbf{x}_A|\omega(\mathbf{x}_B))$ in the same fashion by marginalizing the joint CDF over variables in $V \setminus (A \cup B)$ and computing

$$F(\mathbf{x}_A|\omega(\mathbf{x}_B)) = \frac{F(\mathbf{x}_A, \mathbf{x}_B)}{F(\mathbf{x}_B)} = \frac{\lim_{\mathbf{x}_{V\setminus(A\cup B)}\to\infty} F(\mathbf{x})}{\lim_{\mathbf{x}_{V\setminus B}\to\infty} F(\mathbf{x})}.$$

Note that the above marginalization contrasts with the problem of exact inference in density models, where local marginalization corresponds to computing integrals or sums of the joint PDF/PMF over variable states.

Although obtaining marginals in the CDN is relatively simple, computing and evaluating probability distributions of the form $F(\mathbf{x}_A|\mathbf{x}_B), P(\mathbf{x}_A|\omega(\mathbf{x}_B)), P(\mathbf{x}_A|\mathbf{x}_B)$ and $P(\mathbf{x}_A)$ is more involved. We have seen previously that in order to compute conditional CDFs, we must compute corresponding higher-order derivatives with respect to these observed variables. In particular, given observed data we may wish to numerically evaluate probabilities under the model, such that computing derivatives for each function ϕ_s requires that we store only the numerical value for the derivatives. Provided that the CDN functions are chosen to be themselves tractable to evaluate and differentiate, computing derivatives of these functions will consist of tractable function evaluations.

Since the factorization of the joint CDF modeled by a CDN consists of a product of functions $\phi_s(\mathbf{x}_s)$, the intuition here is that we can distribute the differentiation operation such that at each

function node in the CDN, we compute the derivatives with respect to local variables and pass the result to its neighbors. The resulting algorithm consists of passing messages $\mu_{\alpha\to s}(\mathbf{x}), \mu_{s\to\alpha}(\mathbf{x})$ from variable nodes to function nodes and from function nodes to variable nodes, analogous to the operation of the sum-product algorithm in factor graphs. In the Appendix, we present the derivation of the algorithm in the setting where we wish to compute the mixed derivative of the CDF $F(\mathbf{x})$ modeled by a tree-structured CDN: the derivation is analogous to the derivation for the sum-product algorithm, but with the summation operator replaced by the differentiation operator. To illustrate the corresponding message-passing algorithm, consider the following toy example.

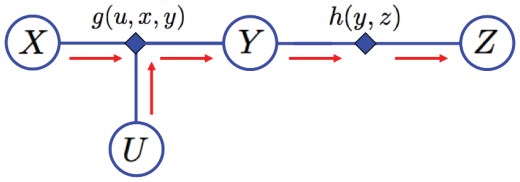


Figure 11: Flow of messages in the toy example of CDN defined over variables X, Y, Z, U.

Example 5 *Consider the CDN over four random variables U, X, Y, Z from Figure 11. The joint CDF is given by $F(u,x,y,z) = g(u,x,y)h(y,z)$. Let Z be the root node so that X and U are leaf nodes. Then the messages from leaf variable nodes to the root are given by*

$$\begin{aligned}
&\mu_{X\to g}(x) = 1,\\
&\mu_{U\to g}(u) = 1,\\
&\mu_{g\to Y}(y;u,x) = \partial_{u,x}\Big[g(u,x,y)\mu_{X\to g}(x)\mu_{U\to g}(u)\Big],\\
&\mu_{Y\to h}(y;u,x) = \mu_{g\to Y}(y;u,x),\\
&\mu_{h\to Z}(z;u,x,y) = \partial_y\Big[h(y,z)\mu_{Y\to h}(y;u,x)\Big].
\end{aligned}$$

Figure 11 shows the flow of the above messages.

Once we have propagated messages from the leaf nodes to the root node, we can evaluate the joint probability $P(u,x,y,z) = \partial_z\Big[\mu_{h\to Z}(z;u,x,y)\Big]$ at the root node as

$$\begin{aligned}
P(u,x,y,z) = \partial_z\Big[\mu_{h\to Z}(z;u,x,y)\Big] &= \partial_z\Big[\partial_y\Big[h(y,z)\mu_{Y\to h}(y;u,x)\Big]\Big]\\
&= \partial_z\Big[\partial_y\Big[h(y,z)\mu_{g\to Y}(y;u,x)\Big]\Big]\\
&= \partial_z\Big[\partial_y\Big[h(y,z)\partial_{u,x}\Big[g(u,x,y)\mu_{X\to g}(x)\mu_{U\to g}(u)\Big]\Big]\Big]\\
&= \partial_{x,y,z,u}\Big[g(u,x,y)h(y,z)\Big]\\
&= \partial_{x,y,z,u}\Big[F(u,x,y,z)\Big].
\end{aligned}$$

The above example illustrates the fact that if the graph topology is a tree, then the message-passing algorithm yields the correct mixed derivatives with respect to each variable in the CDN so that we

obtain the joint probability $P(\mathbf{x}) = \partial_{\mathbf{x}}\Big[F(\mathbf{x})\Big]$ at the root node of the tree by multiplying all incoming messages to the root.

The above example also illustrates a potential source for complexity: each message consists of a symbolic expression that is a sum of products of derivatives of CDN functions. For larger graphs, it is easy to see that such a message-passing scheme would grow in complexity as the symbolic expression for each message would grow in size as we pass from leaf nodes to the root. However, for practical purposes in which we wish to obtain numerical values for probabilities at the observed data, we are interested in *evaluating* derivatives corresponding to marginal/conditional probabilities for observed data $\mathbf{x}$, with unobserved variables marginalized out by taking limits. As the message-passing algorithm allows us to decompose the total mixed derivative computation into a series of local computations, each term in this decomposition consists of a derivative that can be "clamped" to the observed values for its arguments. Moreover, this "clamping" need only be performed locally for each CDN function as we evaluate each outgoing message. In the above example, given observed values u^*, x^* the message $\mu_{g\to Y}(y;u,x)$ consists of computing a derivative with respect to u,x, *followed by evaluation* of the derivative at u^*, x^*. Thus by "clamping" to observed values, messages in the above scheme will not increase in size, regardless of the functional forms chosen for the CDN functions. By evaluating each derivative in the example for u^*, x^*, y^*, z^*, we can obtain a numerical value for the probability $P(u^*, x^*, y^*, z^*)$ by multiplying messages at the root node.

3.2 Inference in Cumulative Distribution Networks

Thus far we have presented a message-passing scheme for computing derivatives of the joint CDF in order to obtain the joint PDF/PMF $P(\mathbf{x})$. Here we will demonstrate the correspondence between computing higher-order derivatives and the problem of inference in a CDN. The relation between differentiation and inference in CDNs is analogous to the relation between marginalization and inference in factor graphs. Thus, in analogy to how the sum-product algorithm allows one to compute distributions of the type $P(\mathbf{x}_A|\mathbf{x}_B)$, message-passing in a CDN allows us to compute conditional distributions of the form $F(\mathbf{x}_A|\mathbf{x}_B)$ and $P(\mathbf{x}_A|\mathbf{x}_B)$ for disjoint sets $A, B \subset V$. In order to compute conditional distributions of the above types, we will assume that when computing a conditional distribution such as $F(\mathbf{x}_A|\mathbf{x}_B)$ or $P(\mathbf{x}_A|\mathbf{x}_B)$, we have $P(\mathbf{x}_B) = \partial_{\mathbf{x}_B}\Big[F(\mathbf{x}_B)\Big] > 0$. Now consider the problem of computing the quantity $F(\mathbf{x}_A|\mathbf{x}_B)$. We can write this as

$$F(\mathbf{x}_A|\mathbf{x}_B) = \frac{\partial_{\mathbf{x}_B}\Big[F(\mathbf{x}_A,\mathbf{x}_B)\Big]}{\partial_{\mathbf{x}_B}\Big[F(\mathbf{x}_B)\Big]} = \frac{\lim\limits_{\mathbf{x}_{V\setminus(A\cup B)}\to\infty}\partial_{\mathbf{x}_B}\Big[F(\mathbf{x})\Big]}{\lim\limits_{\mathbf{x}_{V\setminus B}\to\infty}\partial_{\mathbf{x}_B}\Big[F(\mathbf{x})\Big]} = \frac{\partial_{\mathbf{x}_B}\left[\lim\limits_{\mathbf{x}_{V\setminus(A\cup B)}\to\infty}F(\mathbf{x})\right]}{\partial_{\mathbf{x}_B}\left[\lim\limits_{\mathbf{x}_{V\setminus B}\to\infty}F(\mathbf{x})\right]}$$

$$\propto \partial_{\mathbf{x}_B}\left[\lim_{\mathbf{x}_{V\setminus(A\cup B)}\to\infty}F(\mathbf{x})\right],$$

so that by combining the operations of taking limits and computing derivatives/finite differences, we can compute any conditional probability of the form $F(\mathbf{x}_A|\mathbf{x}_B)$. To compute the conditional CDF for any variable node in the network, we can pass messages from leaf nodes to root and then from the root node back to the leaves. For any given variable node, we can then multiply all incoming

messages to obtain the conditional CDF for that variable, up to a scaling factor. We will now demonstrate this principle using the previous toy example CDN.

Example 6 *Consider the toy example of a CDN over four random variables U,X,Y,Z from Figure 11. Suppose we wish to compute $F(y|x,z) = \lim_{u\to\infty} F(u,y|x,z)$. This is equivalent to message-passing in a CDN defined over variables X,Y,Z with U marginalized out (Figure 12) so that $\tilde{g}(x,y) = \lim_{u\to\infty} g(u,x,y)$. Thus the message updates are given by*

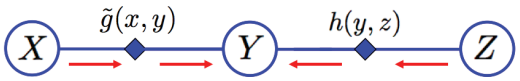


Figure 12: Flow of messages in the toy example CDN of Figure 11 with variable U marginalized out in order to compute the conditional CDF $F(y|x,z)$. Messages are here passed from all observed variable nodes to the root node.

$$\mu_{X\to\tilde{g}}(x) = 1,\ \mu_{\tilde{g}\to Y}(y;x) = \partial_x\Big[\tilde{g}(x,y)\mu_{X\to\tilde{g}}(x)\Big] = \partial_x\Big[\tilde{g}(x,y)\Big],$$
$$\mu_{Z\to h}(z) = 1,\ \mu_{h\to Y}(y;z) = \partial_z\Big[h(y,z)\mu_{Z\to h}(z)\Big] = \partial_z\Big[h(y,z)\Big].$$

Once we have computed the above messages, we can evaluate the conditional CDF $F(y|x,z)$ at node Y as

$$F(y|x,z) = \frac{\mu_{\tilde{g}\to Y}(y;x)\mu_{h\to Y}(y;z)}{\mathcal{Z}} = \frac{\partial_z\Big[h(y,z)\Big]\partial_x\Big[\tilde{g}(x,y)\Big]}{\mathcal{Z}}.$$

Note that the normalizing constant $\mathcal{Z}$ can be readily obtained by computing

$$\mathcal{Z} = \lim_{y\to\infty} \partial_z\Big[h(y,z)\Big]\partial_x\Big[\tilde{g}(x,y)\Big] = \partial_{x,z}\Big[\lim_{y\to\infty} h(y,z)\tilde{g}(x,y)\Big],$$

so that

$$F(y|x,z) = \frac{\mu_{\tilde{g}\to Y}(y;x)\mu_{h\to Y}(y;z)}{\mathcal{Z}} = \frac{\partial_z\Big[h(y,z)\Big]\partial_x\Big[\tilde{g}(x,y)\Big]}{\partial_{x,z}\Big[\lim_{y\to\infty} h(y,z)\tilde{g}(x,y)\Big]} = \frac{\lim_{u\to\infty}\partial_z\Big[h(y,z)\Big]\partial_x\Big[\tilde{g}(u,x,y)\Big]}{\partial_{x,z}\Big[\lim_{u,y\to\infty} h(y,z)\tilde{g}(u,x,y)\Big]}$$
$$= \frac{\partial_{x,z}\Big[\lim_{u\to\infty} F(u,x,y,z)\Big]}{\partial_{x,z}\Big[\lim_{u,y\to\infty} F(u,x,y,z)\Big]}.$$

Note that in the above, if we were to observe $X = x^*, Z = z^*$, we could then evaluate $F(y|x^*,z^*)$ given any candidate value y for variable Y.

The above example shows that the message-passing algorithm can be used to compute conditional CDFs of the form $F(\mathbf{x}_A|\mathbf{x}_B)$, up to a normalizing constant $\mathcal{Z}$. Messages are passed once from all

variable nodes on which we are conditioning to the root node: in the example, messages are passed from variable nodes X, Z to variable node Y in order to compute $F(y|x,z)$. If we wished to compute, say, $F(x|y,z)$, then messages would be passed from variable nodes Y, Z to variable node X.

To obtain distributions of the type $P(\mathbf{x}_A|\mathbf{x}_B)$ from $F(\mathbf{x}_A|\mathbf{x}_B)$, we first compute $\partial_{\mathbf{x}_A}\Big[F(\mathbf{x}_A|\mathbf{x}_B)\Big]$ using the above message-passing scheme and then multiply messages together to obtain conditional PDFs. We note that computing the normalizing constant $\mathcal{Z}$ can be viewed as the result of message-passing in a CDN in which the variables $\mathbf{X}_A$ have been marginalized out in addition to variables $\mathbf{X}_{V\setminus(A\cup B)}$ and then evaluating the resulting messages at the observed values $\mathbf{x}_B$. Equivalently, one can compute $\mathcal{Z} = \lim_{\mathbf{x}_A\to\infty} \partial_{\mathbf{x}_B}\Big[F(\mathbf{x}_A,\mathbf{x}_B)\Big]$ after message-passing with only variables in $V\setminus(A\cup B)$ marginalized out.

3.3 Derivative-sum-product: A Message-passing Algorithm for Inference in Cumulative Distribution Networks

- **Input: A tree-structured CDN $\mathcal{G} = (V,S,E)$, root node $\alpha \in V$ and a vector x of observations**
- **Output: The probability mass function (PMF) $P(\mathbf{x})$**
- For each leaf variable node α' and for all function nodes $s \in \mathcal{N}(\alpha')$, propagate $\mu_{\alpha'\to s}(\mathbf{x}) = 1$. For each leaf function node with function $\phi_s(x_{\alpha'})$, send the messages $\mu_{s\to\alpha'}(\mathbf{x}) = \phi_s(x_{\alpha'})$.
- For each non-leaf variable node α and neighboring function nodes $s \in \mathcal{N}(\alpha)$,
$$\mu_{\alpha\to s}(\mathbf{x}) = \prod_{s'\in\mathcal{N}(\alpha)\setminus s} \mu_{s'\to\alpha}(\mathbf{x}).$$
- For each non-leaf function node s and neighboring variable nodes $\alpha \in \mathcal{N}(s)$,
$$\mu_{s\to\alpha}(\mathbf{x}) = \partial_{\mathbf{x}_{\mathcal{N}(s)\setminus\alpha}}\left[\phi_s(\mathbf{x}_s)\prod_{\beta\in\mathcal{N}(s)\setminus\alpha}\mu_{\beta\to s}(\mathbf{x})\right].$$
- Repeat the 2^{nd} and 3^{rd} steps towards the root node α.

Table 1: The derivative-sum-product (DSP) algorithm for computing the probability mass function $P(\mathbf{x})$ in a CDN defined over discrete variables.

Given that the fundamental operations required for message-passing consist of differentiation/ finite differences, sums and products, we will refer to the above class of message-passing algorithms as the derivative-sum-product (DSP) algorithm. For CDNs defined over discrete ordinal variables, the DSP algorithm is shown in Table 1. As can be seen, for graphs defined over discrete variables, the DSP algorithm is analogous to the sum-product algorithm with the summation operation replaced by a finite difference operation. For graphs defined over discrete ordinal variables that take

on one of K values, for an observed $\mathbf{x}$, each message $\mu_{\alpha\to s},\mu_{s\to\alpha}$ consists of a K-vector, analogous to messages in the sum-product algorithm. To see this, we note that each time we compute a finite difference with respect to variables in $\mathcal{N}(s)\setminus\alpha$, we also evaluate the result at $\mathbf{x}_{\mathcal{N}(s)\setminus\alpha}$, ensuring that each message is a K-vector.

In contrast to the DSP algorithm for discrete variables, the required complexity increases for CDNs defined over continuous variables. For such models, we are required to invoke the product rule of differential calculus in order to express these messages in terms of the derivatives of CDN functions and combinations thereof. To this end, we need to define two additional sets of messages $\lambda_{\alpha\to s}(\mathbf{x})$ and $\lambda_{s\to\alpha}(\mathbf{x})$ which correspond to $\partial_{x_\alpha}\Big[\mu_{\alpha\to s}(\mathbf{x})\Big]$ and $\partial_{x_\alpha}\Big[\mu_{s\to\alpha}(\mathbf{x})\Big]$ respectively. We first derive the expression for $\lambda_{\alpha\to s}(\mathbf{x})$ by applying the product rule of differential calculus to the message $\mu_{\alpha\to s}(\mathbf{x})$, bearing in mind that each of the messages $\mu_{s\to\alpha}(\mathbf{x})$ depends on variable X_α. This yields

$$\lambda_{\alpha\to s}(\mathbf{x})=\partial_{x_\alpha}\Big[\mu_{\alpha\to s}(\mathbf{x})\Big]=\partial_{x_\alpha}\left[\prod_{s'\in\mathcal{N}(\alpha)\setminus s}\mu_{s'\to\alpha}(\mathbf{x})\right]=\mu_{\alpha\to s}(\mathbf{x})\sum_{s'\in\mathcal{N}(\alpha)\setminus s}\frac{\lambda_{s'\to\alpha}(\mathbf{x})}{\mu_{s'\to\alpha}(\mathbf{x})}.$$

In order to derive the general expressions for $\mu_{s\to\alpha}(\mathbf{x})$, $\lambda_{s\to\alpha}(\mathbf{x})$, we first note that for any two differentiable multivariate functions $f(\mathbf{x}),g(\mathbf{x})$, the product rule for computing the higher-order derivative of a product of functions is given by

$$\partial_{\mathbf{y}}\Big[f(\mathbf{y})g(\mathbf{y})\Big]=\sum_{\mathbf{y}_A\subseteq\mathbf{y}}\partial_{\mathbf{y}_A}\Big[f(\mathbf{y})\Big]\partial_{\mathbf{y}\setminus\mathbf{y}_A}\Big[g(\mathbf{y})\Big].$$

The key observation we make here is that to evaluate the above derivative for observed $\mathbf{y}$, we can evaluate each term in the summation for the observed $\mathbf{y}$ such that the above is merely a sum of products of scalars. Thus, given a vector of observed variable values $\mathbf{x}$, the messages in the DSP algorithm for continuous variables will all consist of scalars, allowing us to obtain numerical values for probabilities under the model.

To compute messages $\mu_{s\to\alpha}(\mathbf{x}),\lambda_{s\to\alpha}(\mathbf{x})$ from $\mu_{s\to\alpha}(\mathbf{x})$, applying the above product rule yields

$$\begin{aligned}
\mu_{s\to\alpha}(\mathbf{x})&=\partial_{\mathbf{x}_{\mathcal{N}(s)\setminus\alpha}}\left[\phi_s(x_\alpha,\mathbf{x}_{\mathcal{N}(s)\setminus\alpha})\prod_{\beta\in\mathcal{N}(s)\setminus\alpha}\mu_{\beta\to s}(\mathbf{x})\right]\\
&=\sum_{B\subseteq\mathcal{N}(s)\setminus\alpha}\partial_{\mathbf{x}_B}\Big[\phi_s(\mathbf{x}_s)\Big]\prod_{\beta\in B}\mu_{\beta\to s}(\mathbf{x})\prod_{\beta\in\mathcal{N}(s)\setminus(\alpha\cup B)}\lambda_{\beta\to s}(\mathbf{x}),\\
\lambda_{s\to\alpha}(\mathbf{x})&=\partial_{x_\alpha}\Big[\mu_{s\to\alpha}(\mathbf{x})\Big]\\
&=\sum_{B\subseteq\mathcal{N}(s)\setminus\alpha}\partial_{\mathbf{x}_B,x_\alpha}\Big[\phi_s(\mathbf{x}_s)\Big]\prod_{\beta\in B}\mu_{\beta\to s}(\mathbf{x})\prod_{\beta\in\mathcal{N}(s)\setminus(\alpha\cup B)}\lambda_{\beta\to s}(\mathbf{x}),
\end{aligned}$$

where we have made use of the tree-structure of the CDN to write the derivative of a product of messages as a product of derivatives of the messages. The above updates then define the DSP algorithm for CDNs defined over continuous variables, with a total of four sets of messages defined solely in terms of the CDN functions, their derivatives and linear combinations thereof. The message-passing algorithm for continuous CDNs is summarized in Table 2 and is illustrated in Figure 13.

We see from Table 2 that the DSP algorithm grows exponentially in complexity as the number of neighboring variable nodes for any given function increases, as the updates at function nodes

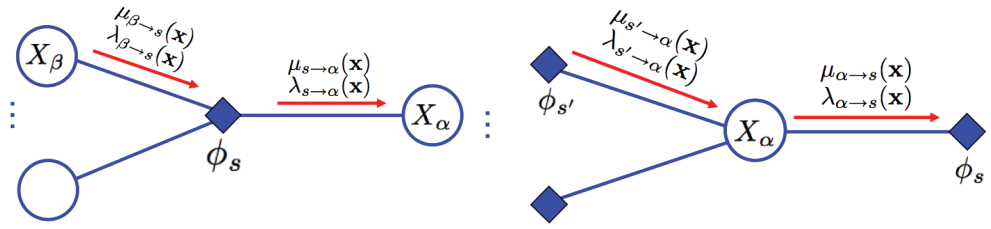


Figure 13: The DSP algorithm. a) Computation of the message from a function node s to a variable node α; b) Computation of the message from a variable node α to a function node s.

- **Input: A tree-structured CDN $\mathcal{G} = (V,S,E)$, root node $\alpha \in V$ and a vector x of observations**
- **Output: The probability density function (PDF)** $P(\mathbf{x})$
- For each leaf variable node α' and for all function nodes $s \in \mathcal{N}(\alpha')$, propagate $\mu_{\alpha'\to s}(\mathbf{x}) = 1, \lambda_{\alpha'\to s}(\mathbf{x}) = 0$. For each leaf function node with function $\phi_s(x_{\alpha'})$, send the messages $\mu_{s\to\alpha'}(\mathbf{x}) = \phi_s(x_{\alpha'}), \lambda_{s\to\alpha'}(\mathbf{x}) = \partial_{x_{\alpha'}}\left[\phi_s(x_{\alpha'})\right]$.
- For each non-leaf variable node α and neighboring function nodes $s \in \mathcal{N}(\alpha)$,

$$\mu_{\alpha\to s}(\mathbf{x}) = \prod_{s'\in\mathcal{N}(\alpha)\setminus s} \mu_{s'\to\alpha}(\mathbf{x}),$$
$$\lambda_{\alpha\to s}(\mathbf{x}) = \partial_{x_\alpha}\left[\mu_{\alpha\to s}(\mathbf{x})\right] = \mu_{\alpha\to s}(\mathbf{x}) \sum_{s'\in\mathcal{N}(\alpha)\setminus s} \frac{\lambda_{s'\to\alpha}(\mathbf{x})}{\mu_{s'\to\alpha}(\mathbf{x})}.$$

- For each non-leaf function node s and neighboring variable nodes $\alpha \in \mathcal{N}(s)$,

$$\mu_{s\to\alpha}(\mathbf{x}) = \sum_{B\subseteq\mathcal{N}(s)\setminus\alpha} \partial_{\mathbf{x}_B}\left[\phi_s(\mathbf{x}_s)\right] \prod_{\beta\in B}\mu_{\beta\to s}(\mathbf{x}) \prod_{\beta\in\mathcal{N}(s)\setminus\{\alpha\cup B\}} \lambda_{\beta\to s}(\mathbf{x}),$$
$$\lambda_{s\to\alpha}(\mathbf{x}) = \partial_{x_\alpha}\left[\mu_{s\to\alpha}(\mathbf{x})\right]$$
$$= \sum_{B\subseteq\mathcal{N}(s)\setminus\alpha} \partial_{\mathbf{x}_B,x_\alpha}\left[\phi_s(\mathbf{x}_s)\right] \prod_{\beta\in B}\mu_{\beta\to s}(\mathbf{x}) \prod_{\beta\in\mathcal{N}(s)\setminus\{\alpha\cup B\}} \lambda_{\beta\to s}(\mathbf{x}).$$

- Repeat the 2^{nd} and 3^{rd} steps towards root node α.

Table 2: The derivative-sum-product (DSP) algorithm for computing the joint probability density function $P(\mathbf{x})$ in a CDN defined over continuous variables.

require one to perform a sum over all subsets of neighboring variables. However, in many cases the computational complexity will be tractable for sparser graphs, as demonstrated by the following example.

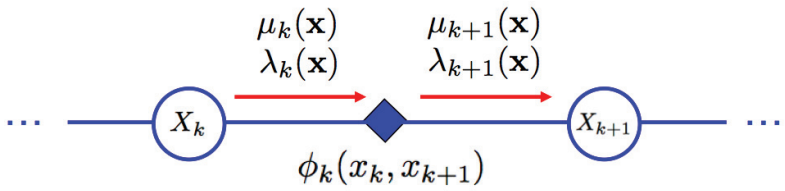


Figure 14: The DSP algorithm for a chain-structured CDN.

Example 7 (Derivative-sum-product on a linear first-order chain CDN) *Consider the CDN defined over K variables such that the joint CDF over these variables is given by*

$$F(\mathbf{x}) = \prod_{k=1}^{K-1} \phi_k(x_k, x_{k+1}),$$

so that the variable nodes are connected in the chain-structured graph shown in Figure 14. In this case, the DSP messages can be written as

$$\begin{aligned}
\mu_{k+1}(\mathbf{x}) &\equiv \mu_{\phi_k \to X_{k+1}}(\mathbf{x}) \\
&= \partial_{x_k}\Big[\phi_k(x_k, x_{k+1})\Big]\mu_k(\mathbf{x}) + \phi_k(x_k, x_{k+1})\lambda_k(\mathbf{x}), \\
\lambda_{k+1}(\mathbf{x}) &\equiv \lambda_{\phi_k \to X_{k+1}}(\mathbf{x}) \\
&= \partial_{x_k, x_{k+1}}\Big[\phi_k(x_k, x_{k+1})\Big]\mu_k(\mathbf{x}) + \partial_{x_{k+1}}\Big[\phi_k(x_k, x_{k+1})\Big]\lambda_k(\mathbf{x}), \quad k = 1, \cdots, K-1.
\end{aligned}$$

Example 8 (Sampling from a cumulative distribution network) *We can further take advantage of the DSP algorithm for generating samples from the CDF modeled by a CDN. We can proceed as follows: arbitrarily select a variable in the model, say X_1. Then, generate a sample x_1^* from its marginal CDF $F(x_1)$ (which we obtain by marginalizing over all other variables). Given x_1^*, we can then proceed to generate samples for its children by marginalizing out all other unobserved variables and then sampling from the conditional distribution $F(x_2|x_1^*)$. We can continue this way until we have sampled a complete configuration $\mathbf{x}^* = [x_1^*, \cdots, x_K^*]$. The algorithm for sampling from the joint CDF modeled by a CDN is then given by*

- *Pick a sampling ordering $X_1, X_2, \cdots, X_K$,*
- *For variable $X_k, k = 1, \cdots, K$, compute*

$$F(x_1, \cdots, x_k) = \lim_{x_{k+1}, \cdots, x_K \to \infty} F(x_1, \cdots, x_k, x_{k+1} \cdots, x_K).$$

- *Sample x_i^* from*

$$F(x_k|x_1, \cdots, x_{k-1}) = \frac{\partial_{x_1, \cdots, x_{k-1}}\Big[F(x_1, \cdots, x_k)\Big]}{\lim_{x_k \to \infty} \partial_{x_1, \cdots, x_{k-1}}\Big[F(x_1, \cdots, x_k)\Big]}.$$

From the above we see that if the CDN has a tree structure, then we can compute the conditional CDFs $F(x_k|x_1,\cdots,x_{k-1})$ exactly via DSP. In the case of a CDN with cycles, we can always convert it to one with a tree structure by clustering variables and corresponding function nodes (Lauritzen and Spiegelhalter, 1988). This generally incurs an increase in function node complexity, but with the benefit of being able to sample from the joint CDF modeled by the CDN.

3.4 Discussion

We have presented the derivative-sum-product (DSP) algorithm for computing derivatives in tree-structured CDNs. For graphs defined over continuous variables, the DSP algorithm can be implemented through two sets of messages in order to compute the higher-order derivatives of the joint CDF. While we have presented the DSP algorithm for computing derivatives given a set of CDN functions, we have not addressed here the issue of how to learn these CDN functions from data. A possible method would be to run DSP to obtain the joint PDF and then maximize this with respect to model parameters for a particular $\mathbf{x}$. Another issue we have not addressed is how to perform inference in graphs with cycles: an interesting future direction would be to investigate exact or approximate methods for doing so and connections to methods in the literature (Minka, 2001; Neal, 1993) for doing this in traditional graphical models. We will further discuss these issues in the concluding section.

Having defined the CDN and having described the DSP algorithm, we will now proceed to apply both of these to the general problem of learning to rank from examples. As we will see, the ability to model a joint CDF using a graphical framework will yield advantages in both representation and computation for this class of problems.

4. Learning to Rank in Multiplayer Team-based Games with Cumulative Distribution Networks

In this section, we will apply CDNs and the DSP algorithm to the problem of structured ranking learning in which the goal is to learn a model for ranking players in a multiplayer game. For this problem, we observe the scores achieved by several players over many games $t = 1\cdots,T$ in which players interactively compete in groups, or teams, which change with each game. For any given game, players compete in teams so that at the end of each game, each player will have achieved a score as a result of actions taken by all players during the game. For example, these player scores could correspond to the number of targets destroyed or the number of flags stolen, so that a higher player score reflects a better performance for that player. Here we will define a game Γ_t as a triplet $(\mathcal{P}_t,\mathcal{T}_t,O_t)$, where $\mathcal{P}_t \subset \mathcal{P}$ is a subset of the set $\mathcal{P}$ of all players and $\mathcal{T}_t$ is a partition of $\mathcal{P}_t$ into sets corresponding to teams for game Γ_t, so that if $\mathcal{T}_t = \{\mathcal{T}_t^1,\cdots,\mathcal{T}_t^N\}$ then there are N teams for game Γ_t and player $k \in \mathcal{P}_t$ is assigned to team n for game Γ_t if and only if $k \in \mathcal{T}_t^n$. For example, a game involving six players labeled $1,2,3,4,5,6$ organized into three teams of two players each could correspond to $\mathcal{P}_t = \{1,2,3,4,5,6\}$ and $\mathcal{T}_t = \{\{1,2\},\{3,4\},\{5,6\}\}$. Without loss of generality we will label the teams in a game by $n = 1,\cdots,N$ where each team corresponds to a set in the partition $\mathcal{T}_t$.

In addition to the above, we will denote by O_t the *outcome* of a game that consists of the pair $(\mathbf{x}_{\mathcal{P}_t},\mathbf{r}_{\mathcal{T}_t})$, where $\mathbf{x}_{\mathcal{P}_t} \in \mathbb{R}^{|\mathcal{P}_t|}$ is a vector of player scores for game Γ_t and the set $\mathbf{r}_{\mathcal{T}_t}$ is defined as a partially ordered set of team performances, or set of ranks for each team. Such ranks are obtained by first computing the sum of the player scores for each team $n = 1,\cdots,N$, and then ranking the

teams by sorting the resulting sums. We will refer to these sums in the sequel as the team scores t_n. An example of this for the previous example is $\mathbf{x}_{\mathcal{P}_t} = [30\ 12\ 15\ 25\ 100\ 23]^T$, so that $\mathbf{r}_{\mathcal{T}_t} = \{2,\ 1,\ 3\}$ is the corresponding partially ordered set of team rankings. We will also denote by $\mathbf{x}_n \in \mathbb{R}^{|\mathcal{T}_t^n|}$ the vector of player scores for team n in game Γ_t. Games can also be classified into various types, such that the sizes and/or number of teams are constrained in different ways for different game types. For example, a "SmallTeam" game type would consist of two teams with at most two players per team, whereas a "FreeForAll" game type would constrain the number of teams to be at most eight, with one player per team. Furthermore, the team rankings are a function of unweighed sums of player scores: although there is no reason *a priori* to weigh the scores of players differently for determining the rank of a team, one could extend the above scheme for determining team rankings to weigh player scores according to player type or player-specific features.

Given the above, the goal is to construct a model that will allow us to predict the outcome O_t of the new game before it begins, given $\mathcal{P}_t$ and previous game outcomes $O_1, \cdots, O_{t-1}$. In particular, we wish to construct a model that will minimize the number of mis-ordered teams based on the set of team performances $\mathbf{r}_{\mathcal{T}_t}$ for game Γ_t. Here, the probability model for the given game should account for the team-based structure of games, such that team performances are determined by individual player scores and a game outcome is determined by the ordering of team scores. We will demonstrate here that the graphical framework of CDNs makes it straightforward to model both stochastic orderings of variables in the model as well as statistical independence relationships among these variables. In particular, the model we will construct here will be amenable to exact inference via the DSP algorithm.

Our model will be similar in design to the TrueSkill™ model of Herbrich et al. (2007) for skill rating in Halo 2™, whereby each player $k \in \mathcal{P}_t$ is assigned a probability distribution over latent skill variables S_k, which is then inferred from individual player scores over multiple games using the expectation propagation algorithm for approximate inference (Minka, 2001). Inference in the TrueSkill™ model thus consists of applying expectation propagation to a factor graph for a given game in order to update probabilities over player skills. An example of such a factor graph is shown in Figure 15. In TrueSkill™, the factors connecting team-specific nodes to one another dictate a constraint on relative differences in the total player scores between teams, while factors connecting player nodes to their team-specific nodes enforce the constraint that the team score is determined by the sum of player scores. Finally, for teams $n, n+1$, there is a difference variable $H_{n,n+1}$ and a corresponding factor which declares a tied rank between two teams if the difference between the two team scores is below some threshold parameter.

4.1 A Cumulative Distribution Network for Modeling Multiplayer Game Outcomes

Here we will examine a model for multiplayer game outcomes that will be modeled using a CDN. The model will be designed on a game-by-game basis where the team assignments of players for a given game determines the connectivity of the graph $\mathcal{G}$ for the CDN. In our model the team variables will correspond to the ranks of teams: we will call such variables *team performances* and denote these as R_n for team n in order to contrast these with the team score variables T_n in the TrueSkill model. Our model will account for player scores X_k for each player $k \in \mathcal{P}_t$ in the game, the team performances R_n for each team $n = 1, \cdots, N$ in the game and each player's skill function $s_k(x_k)$, which is a CDF specific to each player. For any given game, R_n will be determined as the sum of the player scores for team n, and then sorting the resulting sums so that R_n corresponds to the rank of

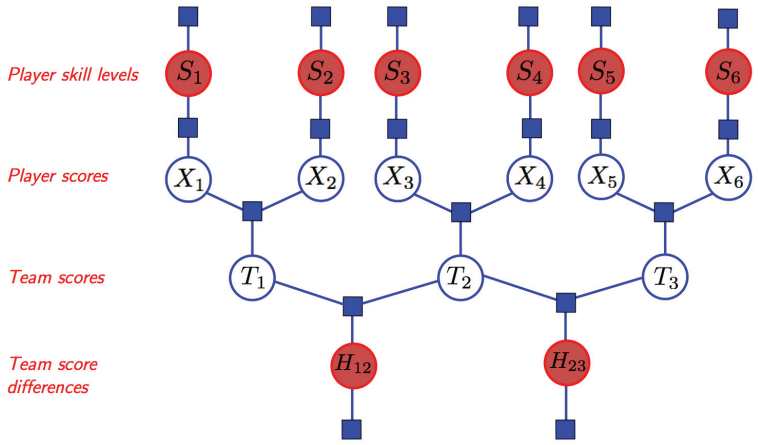


Figure 15: The TrueSkill$^{\text{TM}}$ factor graph for a particular Halo 2$^{\text{TM}}$ game involving three teams with two players each with the team scores $T_1 = t_1, T_2 = t_2, T_3 = t_3$ with $t_1 < t_2 < t_3$ so that team 3 here achieved the highest total of player scores. The variables H_{12}, H_{23} correspond to differences in team scores which determine the ranking of teams, so that teams n and $n+1$ are tied in their rankings if the difference in their team scores is below a threshold parameter. Here, $\mathcal{P}_t = \{1,2,3,4,5,6\}$ and $\mathcal{T}_t = \{\{1,2\},\{3,4\},\{5,6\}\}$. Unobserved variables correspond to nodes shaded in red and observed variables correspond to unshaded variable nodes. Each player $k = 1,2,3,4,5,6$ is assigned a skill function that reflects the distribution of that player's skill level S_k given past game outcomes. Each player then achieves score X_k in any given game and team scores $T_n, n = 1,2,3$ are then determined as the sum of player scores for each team.

team n. The set of observed team performances $\mathbf{r}_{\mathcal{T}_t}$ will be given by the joint configuration of the R_n variables for that game. The goal will then be to adapt player skill functions $s_k(x_k)$ for each game as a function of game outcome. We will design our model according to two principles. First, the relationship between player scores and team performances is modeled as being stochastic, as both player scores and team assignments vary from one game to the next, so that given knowledge of the players in that game and their team assignments, there is some uncertainty in how a team will rank once the game is over. Second, team performance variables depend on those of other teams in the game, so that each team's performance should be linked to that of other teams in a game.

The CDN framework allows us to satisfy both desiderata in the form of modeling constraints on the marginal CDFs for variables in the model. To address the first point, we will require a set of CDN functions that connect player scores to team performances. Here we will make use of the cumulative model for ordinal regression (see Appendix) that relates a linear function $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ on inputs $\mathbf{x}$ to a single output ordinal variable $y \in \{r_1, \cdots, r_L\}$ so that $\mathbb{P}[y = r_l] = \mathbb{P}[\theta(r_{l-1}) < f(\mathbf{x}) + \varepsilon \leq \theta(r_l)] = F_\varepsilon(\theta(r_l) - f(\mathbf{x})) - F_\varepsilon(\theta(r_{l-1}) - f(\mathbf{x}))$, where ε is an additive noise variable and $\theta(r_0), \cdots, \theta(r_L)$ are the cutpoint parameters of the model with $\theta(r_0) = -\infty, \theta(r_L) = \infty$. Equivalently, we can write $\mathbb{P}[y \leq r_l] = \mathbb{P}[\varepsilon \leq \theta(r_l) - f(\mathbf{x})]$. In the context of multiplayer games, we perform separate ordinal regressions for different game types, as the cutpoints that are learned for a given game type may vary between different game types due to differing team sizes between game types. For a given game

type, we treat the set of all games as a bag of pairs of player score vectors $\mathbf{x}_n$ and team performances r_n from which cutpoints in an ordinal regression model can be learned. Thus, we learn a set of cutpoints $\theta(r_0) < \cdots < \theta(r_L)$ once using all of the games in the training data set for a given game type. Team performances are treated as being independent: thus, we can use the CDN framework to augment the above parametric model in order to account for statistical dependencies between multiple team performances in any given game.

We will model multiplayer games using a CDN in which players are grouped into teams and teams compete with one another. To model dependence between player scores and team performance, we will combine the above cumulative model for ordinal regression with prior player score distributions under the assumptions that players contribute equally to team performance and that players perform equally well on average. To do this, we will use functions g_n where if there are N teams for any given game, then we can assign a CDN function g_n for each team such that

$$g_n(\mathbf{x}_n, r_n) = \int_{-\infty}^{\mathbf{x}_n} F\left(\theta(r_n); \mathbf{1}^T\mathbf{u}, \sigma^2\right) P\left(\mathbf{u}\right) d\mathbf{u}, \tag{2}$$

where $F\left(\theta(r_n); \mathbf{1}^T\mathbf{u}, \sigma^2\right)$ is a cumulative model relating input player scores to output team performance and $\mathbf{x}_n, r_n$ are the player scores and team performance for team n. The regression function in the cumulative model is given by $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ with $\mathbf{w}$ set to the vector of ones $\mathbf{1}$, as we weigh the contributions of players on a team equally. Furthermore, $\theta(r_n)$ are the cutpoints that define contiguous intervals in which r_n is the ranking for team n based on that team's performance and $P(\mathbf{u})$ is a probability density over a vector of latent player scores $\mathbf{u}$. Once the cutpoints have been estimated by ordinal regression, we will model the distributions $F\left(\theta(r_n); \mathbf{1}^T\mathbf{u}, \sigma^2\right), P\left(\mathbf{u}\right)$ in Equation (2) as

$$\begin{aligned} F\left(\theta(r_n); \mathbf{1}^T\mathbf{u}, \sigma^2\right) &= \Phi\left(\theta(r_n); \mathbf{1}^T\mathbf{u}, \sigma^2\right), \\ P(\mathbf{u}) &= \mathit{Gaussian}(\mathbf{u}; \mu\mathbf{1}, \sigma^2\mathbf{I}). \end{aligned}$$

By combining functions g_n (which assume equal player skills on average) with individual player skills whilst accounting for the dependence between players' skills and team performances, we can update each player's skill function s_k conditioned on the outcome of a game.

To address the fact that teams compete in any given game, we model ordinal relationships between team performance using the notion of stochastic orderings (Section 2.3), so that for two teams with team performances R_X, R_Y, $R_X \preceq R_Y$ if $F_{R_X}(t) \geq F_{R_Y}(t) \forall\, t \in \mathbb{R}$, where $F_{R_X}(\cdot), F_{R_Y}(\cdot)$ are the marginal CDFs of R_X, R_Y. This then allows us to design models in which we can express differences in team performances in the form of pairwise constraints on their marginal CDFs. We note at this juncture that while it is possible to model such stochastic ordering constraints between variables using directed, undirected or factor graphs, doing so introduces additional constraints that are likely to increase the difficulty of performing inference under such models. In contrast, the CDN framework here allows us to explicitly specify such stochastic ordering constraints, in addition to allowing for tractable computations in the resulting model. Thus, although each of the R_n variables is a deterministic function of the sum of player scores, we can nevertheless model them as being stochastic using the framework of CDNs to specify orderings among the R_n variables. By contrast, it will generally be more difficult in terms of computation and representation to enforce constraints of the type $\left[R_n \preceq R_{n+1}\right]$ in a directed/undirected/factor graph model.

For the proposed CDN model, given N ranked teams, we can thus define $N-1$ functions $h_{n,n+1}$ so that

$$h_{n,n+1}(r_n, r_{n+1}) = \Phi\left(\begin{bmatrix} r_n \\ r_{n+1} \end{bmatrix}; \begin{bmatrix} \tilde{r}_n \\ \tilde{r}_{n+1} \end{bmatrix}, \boldsymbol{\Sigma} \right),$$

where

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma^2 & \rho\sigma^2 \\ \rho\sigma^2 & \sigma^2 \end{bmatrix},$$

and $\tilde{r}_n \leq \tilde{r}_{n+1}$ are chosen without loss of generality such that $\tilde{r}_n = n$ so as to enforce $R_n \preceq R_{n+1}$ in the overall model. Finally, we will use a *skill function* $s_k(x_k)$ for each player k to model that player's distribution over game scores given previous game outcomes. The player performance nodes in the CDN will then be connected to the team performance nodes via the above CDN functions g_n and team performance variable nodes R_n are linked to one another via the above CDN functions $h_{n,n+1}$. The joint CDF for a given game Γ_t with N teams is then given by

$$F(\mathbf{x}_{\mathcal{P}_t}, \mathbf{r}_{\mathcal{T}_t}) = \prod_{n=1}^{N} g(\mathbf{x}_n, r_n) \prod_{n=1}^{N-1} h_{n,n+1}(r_n, r_{n+1}) \prod_{k \in \mathcal{P}_t} s_k(x_k).$$

The above functions and model variables jointly define the CDN for modeling multiplayer games. An example is given in Figure 16 for a game with three teams and six players. One can readily verify from the CDN of Figure 16 using Proposition 18 that for the above model and for any given game, the stochastic ordering relationship $R_1 \preceq R_2 \preceq \cdots \preceq R_N$ as defined above can be enforced by marginalizing over all player scores in the CDN and having selected appropriate cutpoints that satisfy $\theta(r_1) < \theta(r_2) < \theta(r_3)$ and parameters $\tilde{r}_1 < \tilde{r}_2 < \tilde{r}_3$, so that we have $F(r_1) \geq F(r_2) \geq F(r_3)$.

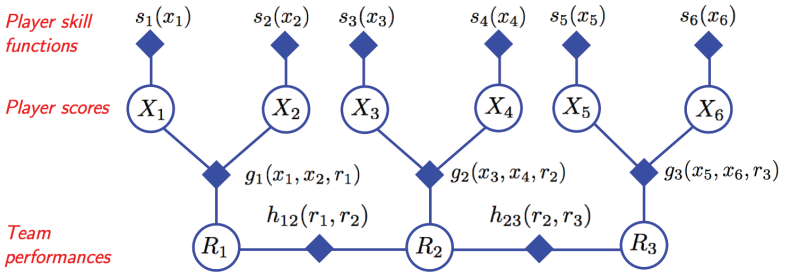


Figure 16: The CDN for the player scores and team performances in a game of Halo 2™ for a game with three teams with two players each. Each player $k = 1,2,3,4,5,6$ achieves score X_k in a match and team performances $R_n, n = 1,2,3$ are determined based on the sum of player scores for each team.

Having presented the CDN for modeling multiplayer games, we will now proceed to describe a method for predicting game outcomes in which we update player skill functions after each game using message-passing.

4.2 Ranking Players in Multiplayer Games Using the Derivative-sum-product Algorithm

Here we will apply the DSP algorithm in the context of ranking players in multiplayer games with a team structure, where the problem consists of jointly predicting multiple ordinal output variables. It should be noted that while it may be possible to construct similar models using a directed, undirected or factor graph, the CDN allows us to simultaneously specify both ordinal and statistical independence relationships among model variables while allowing for a tractable inference algorithm.

In order to compute the DSP messages using the above CDN functions, we must compute the derivatives of all CDN functions. Since all of our functions are themselves Gaussian CDFs, the derivatives $\partial_{\mathbf{x}_A}\left[\phi_s(\mathbf{x}_s)\right]$ can be easily evaluated with respect to variables $\mathbf{X}_A$ as

$$\partial_{\mathbf{x}_A}\Big[\Phi\big(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}\big)\Big] = \mathit{Gaussian}\Big(\mathbf{x}_A;\boldsymbol{\mu}_A,\boldsymbol{\Sigma}_A\Big)\Phi\Big(\mathbf{x}_B;\tilde{\boldsymbol{\mu}}_B,\tilde{\boldsymbol{\Sigma}}_B\Big),$$

where

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_A \\ \mathbf{x}_B \end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_A \\ \boldsymbol{\mu}_B \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_A & \boldsymbol{\Sigma}_{A,B} \\ \boldsymbol{\Sigma}_{A,B}^T & \boldsymbol{\Sigma}_B \end{bmatrix},$$

$$\tilde{\boldsymbol{\mu}}_B = \boldsymbol{\mu}_B + \boldsymbol{\Sigma}_{A,B}^T\boldsymbol{\Sigma}_A^{-1}(\mathbf{x}_A - \boldsymbol{\mu}_A),$$
$$\tilde{\boldsymbol{\Sigma}}_B = \boldsymbol{\Sigma}_B - \boldsymbol{\Sigma}_{A,B}^T\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\Sigma}_{A,B}.$$

The message computations in the CDN are given in the Appendix. We ensure that each message is properly normalized by locally computing the constant $\mathcal{Z}$ as described in Section 3.2 for each message and multiplying each message pair μ,λ by $\mathcal{Z}^{-1}$.

Given the above CDN model for multiplayer games, we would like to then estimate the player skill functions $s_k(x_k)$ for each player k from previous games played by that player. Let $T_k \subseteq \{1,\cdots,T\}$ be the set of games in which player k participated. We then seek to estimate $s_k(x_k)$ for player k given previous team performances $\mathbf{r}_{\mathcal{T}_t}, t \in T_k$ and player scores for *all other* players $\mathbf{x}_{\mathcal{P}_t\setminus k}$ for all games $t \in T_k$ in which player k participated. Denote by O_t^{-k} the outcome of a game with the player score for player k removed from $\mathbf{x}_{\mathcal{P}_t}$. We will define the skill function $s_k(x_k)$ for a player to be given by

$$s_k(x_k) = F\left(x_k|\{O_t^{-k}\}_{t\in T_k}\right) = \prod_{t\in T_k} F(x_k|O_t^{-k}).$$

The above expression for the skill function $s_k(x_k)$ for player k corresponds to the conditional distribution $F\left(x_k|\{O_t^{-k}\}_{t\in T_k}\right)$ given all past games played by player k with the assumption that team performances and player scores are independently drawn from CDFs $F(\mathbf{r}_{\mathcal{T}_t},\mathbf{x}_{\mathcal{P}_t})$ for $t = 1,\cdots,T$. The skill function s_k can then be readily estimated by the DSP algorithm, since each game outcome is modeled by a tree-structured CDN. More precisely, we first initialize $s_k(x_k)$ to the Gaussian CDF $\Phi(x_k;\mu,\beta^2)$ evaluated at many values for x_k. For each game Γ_t we can perform message-passing to obtain the conditional CDF $F(x_k|O_t^{-k}) = \mu_{g_n\to X_k}(\mathbf{r}_{\mathcal{T}_t},\mathbf{x}_{\mathcal{P}_t\setminus k})$ for player k (assuming the message $\mu_{g_n\to X_k}$ has been properly normalized as described above) and then perform a multiplicative update $s_k(x_k) \leftarrow s_k(x_k)\mu_{g_n\to X_k}(\mathbf{r}_{\mathcal{T}_t},\mathbf{x}_{\mathcal{P}_t\setminus k})$. The updates consist of pointwise multiplications of $s_k(x_k)$ and $\mu_{g_n\to X_k}$ for different values of x_k. The skill function $s_k(x_k)$ can then be used to make predictions for player k's scores in future games. We will proceed in the next section to apply the model and the above inference procedure to the problem of modeling Halo 2™ games.

4.3 The Halo 2™ Beta Data Set

The Halo 2™ Beta data set (v1.1)[2] consists of player scores for four game types ("HeadToHead", "FreeForAll", "SmallTeams" and "LargeTeams") over a total of 6,465 players. The descriptions for each of the four game modes are given below.

- HeadToHead: 6227 games/1672 players, one player competing against another player
- FreeForAll: 60022 games/5943 players, up to eight players playing against each other
- SmallTeams: 27539 games/4992 players, up to four players per team, two competing teams
- LargeTeams: 1199 games/2576 players, up to eight players per team, two competing teams

To construct the above CDN model, we set the cutpoints $\theta(r_n)$ in the above cumulative model using ordinal regression of team ranks on team performances for all games in the training set. We initialized all player skill functions to $s_k(x_k) = \Phi(x_k; \mu, \beta^2)$. The set of parameters $\{\mu, \rho, \beta, \sigma\}$ in the CDN model was set to $\{25, -0.95, 20, 0.25\}$ for "HeadToHead", $\{50, -0.2, 10, 0.2\}$ for "FreeForAll", $\{20, -0.1, 10, 0.027\}$ for "SmallTeams" and $\{1, -0.9, 1, 0.01\}$ for "LargeTeams" game modes.[3] For each of these game modes, we applied the DSP algorithm as described above in order to obtain updates for the player skill functions $s_k(x_k)$. An example of such an update at the end of a game with four competing players is shown in Figure 17.

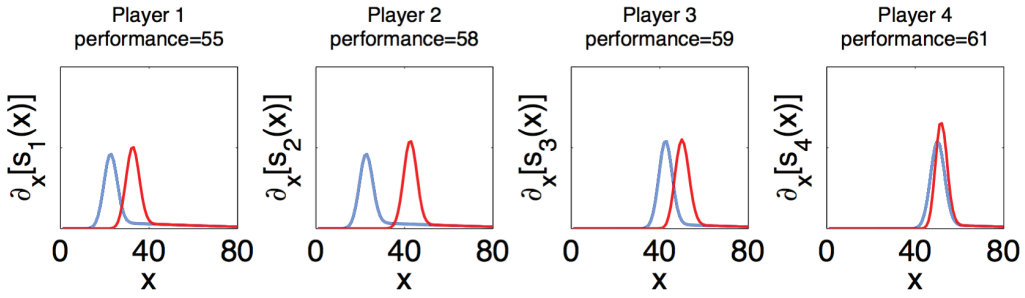


Figure 17: An example of derivative-sum-product updates for a four-player free-for-all game, with the derivative of the skill functions before the updates (light blue) and afterwards (red).

Before each game, we can predict the team performances using the player skills learned thus far via the rule $x_k^* = \arg\max_{x_k} \partial_{x_k} \left[s_k(x_k) \right]$. For each game, the set of team performances is then defined by the ordering of teams once the game is over, where we add the predicted player scores x_k^* together for each team and sorting the resulting sums in ascending order. For any predicted set of team performances, an error is incurred for that game if two teams for that game were mis-ranked such that the number of errors for a given game is $\sum_{n=1}^{N-1} \sum_{m>n}^{N} [R_n \leq R_m] \wedge [R_n^{true} > R_m^{true}]$. One can then compute an error rate over the entire set of games for which we make predictions about team performances.

2. Credits for the use of the Halo 2™ Beta Data Set are given to Microsoft Research Ltd. and Bungie.
3. These parameter settings were selected using a validation set and differ from those of Huang and Frey (2008).

A plot showing the average prediction error rate obtained for the above CDN models over five runs of DSP is shown in Figure 18. It is worth noting that our choice of multivariate Gaussian CDFs as CDN functions in the above model requires that we use a rejection sampling method in order to evaluate the CDN functions, so that the error bars over the five runs are shown. In addition, Figure 18 also shows the error rates reported by Herbrich et al. (2007) for TrueSkill™ and ELO (Elo, 1978), which is a statistical rating system used in chess. Here, we see that the ability to specify both ordinal relationships and statistical dependence relationships between model variables using a CDN allows us to achieve higher predictive accuracy than either TrueSkill™or the ELO method.

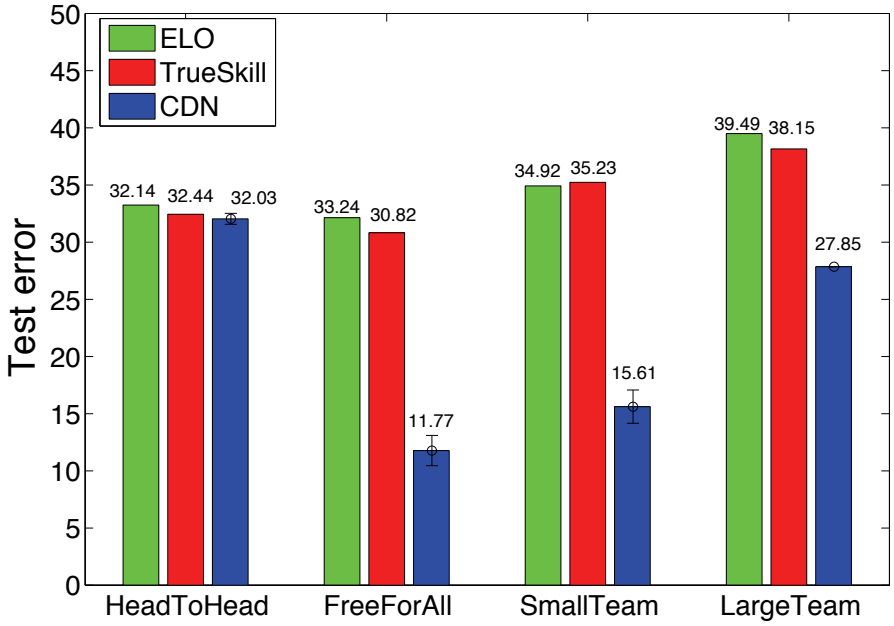


Figure 18: Prediction error on the Halo 2™ Beta data set (computed as the fraction of team predicted incorrectly before each game) for DSP, ELO (Elo, 1978) and TrueSkill™ (Herbrich, Minka and Graepel, 2007) methods. Error bars over five runs of DSP are shown.

4.4 Discussion

In this section we presented a model and method for learning to rank in the context of multiplayer team-based games such as Halo 2™. Our model represent both statistical dependence relationships and stochastic orderings of variables in the model such as team performances and individual player scores. We then used the DSP algorithm to compute conditional CDFs for each player's score. Comparisons to the TrueSkill™ and ELO methods for factor graph models show that our model and method allows both for fast estimation and improved test error on the Halo 2™ Beta data set.

While the above method has the advantage of providing a flexible probabilistic model and allowing for tractable inference, the choice of multivariate Gaussian CDFs for CDN functions requires

the use of sampling methods in order to evaluate DSP messages. Future work could focus on finding closed-form parameterizations of CDN functions for which computing derivatives is tractable.

5. Conclusion

We have proposed the CDN as a graphical model for joint CDFs over many variables. We have shown that the conditional independence properties of a CDN are distinct from the independence properties of directed, undirected and factor graphs. However, these properties include, but are not limited to, those for bi-directed graphs. We have then demonstrated that inference in a CDN corresponds to computing derivatives/finite differences. We described the DSP algorithm for computing such derivatives/finite differences by passing messages in the CDN where each message corresponds to local derivatives of the joint CDF.

We used the graphical framework provided by CDNs to formulate models and methods for learning to rank in a structured setting in which we must account for statistical dependence relationships between model variables. We first applied the DSP algorithm to the problem of ranking in multiplayer gaming where players compete in teams. The DSP algorithm allowed us to compute distributions over player scores given previous game outcomes while accounting for the team-based structure of the games, whereby we were able to show improved results over previous methods. The CDN framework was then used to construct loss functionals for structured ranking learning where we wish to account for statistical dependence relationships which arise in ranking a set of objects. We showed that many probability models for rank data can be viewed as particular CDNs with different connectivities between pairwise object preferences. Based on the work and results presented, we can recommend future directions of research pertaining to the methods presented in this manuscript.

5.1 Future Work

While we presented a framework for constructing a graphical model for a joint CDF, there may be applications in which one may wish to instead optimize the log-probability density $\log P(\mathbf{x}|\boldsymbol{\theta})$. We presented the DSP algorithm for both discrete and continuous-variable networks and we showed how DSP could be used to compute the probability density $P(\mathbf{x}|\boldsymbol{\theta})$ from the joint CDF $F(\mathbf{x}|\boldsymbol{\theta})$ modeled by the CDN. In order to perform maximum likelihood learning in which we wish to maximize the log-likelihood $\mathcal{L}(\mathbf{x};\boldsymbol{\theta}) = \log P(\mathbf{x}|\boldsymbol{\theta})$ with respect to a parameter vector $\boldsymbol{\theta}$ for a given set of observed variables $\mathbf{x}$, one can use modified versions of DSP messages in order to compute the gradient $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x};\boldsymbol{\theta})$ of the log-likelihood. The guiding principle here is that the gradient operator can be distributed amongst local functions in the CDF, much like the differentiation operation in DSP, so that by modifying DSP messages appropriately we can obtain the gradient $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathbf{x};\boldsymbol{\theta})$. Once computed, the gradient vector can then be used in a gradient-descent algorithm to optimize the log-likelihood. Future research in this direction could be directed at establishing what class of graphs can yield tractable gradient computations, as well as the complexity/accuracy tradeoffs involved in computing gradients in graphs with cycles.

We have shown that our message-passing algorithm leads to the correct set of derivatives of the joint CDF provided that the underlying graph is a tree. As with the sum-product algorithm for factor graphs, if the graph contains cycles, then the derivative-sum-product is no longer guaranteed to yield the correct mixed derivatives of the joint CDF, so that messages may begin to 'oscillate' as they propagate around cycles in the graph. One important direction to pursue is to establish

conditions under which the presence of cycles will not lead to oscillations in messages: one could resort to a similar methodology as that employed by Weiss and Freeman (2001), where a graph with cycles is "unwrapped" and the resulting messages are analyzed.

We showed that for graphs defined over continuous variables, the complexity of computing DSP message updates at a given function node increased exponentially with the number of neighboring variable nodes, as one has to sum over products of messages incoming from all subsets of variables connected to the function node. However, it may be possible to approximate messages using simpler, tractable forms such as conditional univariate Gaussian CDFs. Future work here would be to establish tractable methods for performing such approximations and gauge the performance of such an approximate scheme for inference in CDNs on both synthetic and real-world data.

As we have demonstrated, the graph separation criterion for assessing conditional independence in CDNs includes those of bi-directed graphs (Richardson and Spirtes, 2002). As such graphs are a special case of mixed graphs containing undirected, directed and bi-directed edges, a future avenue of research would be to investigate whether one can tractably approximate such mixed graphical models using a hybrid graphical formulation combining the CDN model with that of factor graphs for joint probability density/mass functions. The Bayesian learning approach adopted by Silva and Ghahramani (2009b) could provide a framework with which to qualitatively and quantitatively compare the use of CDNs for constructing such mixed graphical models.

Acknowledgments

Part of this work was done while JCH was a graduate student at the University of Toronto. We wish to thank Zoubin Ghahramani, Nebojsa Jojic, Frank Kschischang, Christopher Meek, Tom Minka, Radford Neal and Thomas Richardson for valuable comments and feedback that have contributed significantly to the content of the manuscript.

Appendix A.

Lemma 19 *Let* $M = \{\mathbf{x}_C < \mathbf{X}_C \leq \mathbf{x}_C + \boldsymbol{\epsilon}\} \equiv \bigcap_{\gamma \in C} \{x_\gamma < X_\gamma \leq x_\gamma + \varepsilon\}$ *with* $\varepsilon > 0$ *for* $\mathbf{X}_C \subset \mathbf{X}$ *and* $\boldsymbol{\epsilon} = [\varepsilon \ \cdots \ \varepsilon]^T \in \mathbb{R}^{|\mathbf{X}_C|}$. *Consider the set of random variables* $\mathbf{X}_A \subset \mathbf{X}$ *with* $\mathbf{X}_C \cap \mathbf{X}_A = \emptyset$. *If both* $\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_C)\Big]$ *and* $\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_A, \mathbf{x}_C)\Big]$ *exist for all* $\mathbf{x}_C$ *with* $\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_C)\Big] > 0$, *then the conditional CDF* $F(\mathbf{x}_A|\mathbf{x}_C) \equiv \lim_{\boldsymbol{\epsilon}\to\mathbf{0}^+} F(\mathbf{x}_A|\mathbf{x}_C < \mathbf{X}_C \leq \mathbf{x}_C + \boldsymbol{\epsilon}) = \lim_{\boldsymbol{\epsilon}\to\mathbf{0}^+} \dfrac{\mathbb{P}\Big[\{\mathbf{X}_A \leq \mathbf{x}_A\} \cap \{\mathbf{x}_C < \mathbf{X}_C \leq \mathbf{x}_C + \boldsymbol{\epsilon}\}\Big]}{\mathbb{P}\Big[\mathbf{x}_C < \mathbf{X}_C \leq \mathbf{x}_C + \boldsymbol{\epsilon}\Big]}$ *is given by*

$$F(\mathbf{x}_A|\mathbf{x}_C) = \frac{\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_A, \mathbf{x}_C)\Big]}{\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_C)\Big]} \propto \partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_A, \mathbf{x}_C)\Big],$$

where $\partial_{\mathbf{x}_C}\Big[\cdot\Big]$ *is a mixed derivative operator with respect to* $\{x_\gamma, \gamma \in C\}$.

Proof We can proceed by induction on variable set $\mathbf{X}_C$ with the base case given by Lemma 3. Let $\mathbf{X}_C = \mathbf{X}_{C'} \cup X_\beta$ with $X_\beta \notin \mathbf{X}_{C'} \cup \mathbf{X}_A$. Let $M' \equiv M'(\boldsymbol{\xi}) = \{\mathbf{x}_{C'} \leq \mathbf{X}_{C'} \leq \mathbf{x}_{C'} + \boldsymbol{\xi}\} = \bigcap_{\gamma \in C'} \{x_\gamma < X_\gamma \leq x_\gamma + \xi\}$ and $M \equiv M(\boldsymbol{\xi}, \varepsilon) = M' \cap \{x_\beta < X_\beta \leq x_\beta + \varepsilon\}$ with $\boldsymbol{\epsilon} = [\boldsymbol{\xi}^T \ \varepsilon]^{\mathrm{T}}$. Suppose that $\partial_{\mathbf{x}_{C'}}\Big[F(\mathbf{x}_{C'})\Big] > 0$ and we have computed

$$F(\mathbf{x}_A, x_\beta | \mathbf{x}_{C'}) \equiv \lim_{\boldsymbol{\xi} \to 0^+} F\big(\mathbf{x}_A, x_\beta | M'(\boldsymbol{\xi})\big) = \frac{\partial_{\mathbf{x}_{C'}}\Big[F(\mathbf{x}_A, x_\beta, \mathbf{x}_{C'})\Big]}{\partial_{\mathbf{x}_{C'}}\Big[F(\mathbf{x}_{C'})\Big]},$$

and

$$F(x_\beta | \mathbf{x}_{C'}) \equiv \lim_{\boldsymbol{\xi} \to 0^+} F\big(x_\beta | M'(\boldsymbol{\xi})\big) = \frac{\partial_{\mathbf{x}_{C'}}\Big[F(x_\beta, \mathbf{x}_{C'})\Big]}{\partial_{\mathbf{x}_{C'}}\Big[F(\mathbf{x}_{C'})\Big]}.$$

Then we can write

$$F(\mathbf{x}_A \mid M) = \frac{\mathbb{P}\Big[\{\mathbf{X}_A \leq \mathbf{x}_A\} \cap \{x_\beta < X_\beta \leq x_\beta + \varepsilon\} \mid M'\Big]}{\mathbb{P}\Big[x_\beta < X_\beta \leq x_\beta + \varepsilon \mid M'\Big]} = \frac{\frac{F(\mathbf{x}_A, x_\beta + \varepsilon | M') - F(\mathbf{x}_A, x_\beta | M')}{\varepsilon}}{\frac{F(x_\beta + \varepsilon | M') - F(x_\beta | M')}{\varepsilon}}.$$

Thus, since $\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_C)\Big] > 0$ by hypothesis, we obtain

$$\begin{aligned} F(\mathbf{x}_A | \mathbf{x}_C) &= \lim_{\varepsilon \to 0^+, \boldsymbol{\xi} \to 0^+} \frac{\frac{F(\mathbf{x}_A, x_\beta + \varepsilon | M') - F(\mathbf{x}_A, x_\beta | M')}{\varepsilon}}{\frac{F(x_\beta + \varepsilon | M') - F(x_\beta | M')}{\varepsilon}} = \frac{\lim_{\varepsilon \to 0^+} \frac{F(\mathbf{x}_A, x_\beta + \varepsilon | \mathbf{x}_{C'}) - F(\mathbf{x}_A, x_\beta | \mathbf{x}_{C'})}{\varepsilon}}{\lim_{\varepsilon \to 0^+} \frac{F(x_\beta + \varepsilon | \mathbf{x}_{C'}) - F(x_\beta | \mathbf{x}_{C'})}{\varepsilon}} \\ &= \frac{\partial_{x_\beta, \mathbf{x}_{C'}}\Big[F(\mathbf{x}_A, x_\beta, \mathbf{x}_{C'})\Big]}{\partial_{x_\beta, \mathbf{x}_{C'}}\Big[F(x_\beta, \mathbf{x}_{C'})\Big]} = \frac{\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_A, \mathbf{x}_C)\Big]}{\partial_{\mathbf{x}_C}\Big[F(\mathbf{x}_C)\Big]}. \end{aligned}$$

■

Thus a conditional CDF of the form $F(\mathbf{x}_A | \mathbf{x}_C)$ can be obtained by differentiation of the joint CDF. By Schwarz's Theorem this differentiation is invariant to the order in which variables are processed provided that the derivatives required to compute $F(\mathbf{x}_A | \mathbf{x}_C)$ exist and are continuous.

A.1 Derivation of the Derivative-sum-product Algorithm

To begin, let $\mathcal{G} = (V, S, E)$ be a tree-structured CDN and suppose we wish to compute the joint probability $P(\mathbf{x})$ and evaluate it at observation $\mathbf{x}$. We note that we can root the graph at some node α and we can write the joint CDF as

$$F(\mathbf{x}) = \prod_{s \in \mathcal{N}(\alpha)} T_s\big(\mathbf{x}_{\tau_s^\alpha}\big),$$

where $\mathbf{x}_{\tau_s^\alpha}$ denotes the vector of configurations for all variables in the subtree τ_s^α rooted at variable node α and containing function node s (Figure 19), and $T_s\big(\mathbf{x}_{\tau_s^\alpha}\big)$ corresponds to the product of all functions located in the subtree τ_s^α.

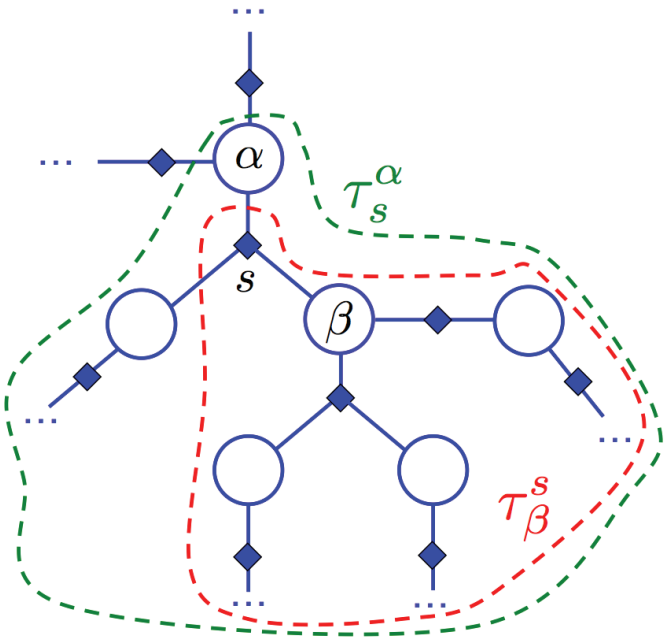


Figure 19: Example of the subtrees $\tau_s^\alpha, \tau_\beta^s$ for a tree-structured CDN given by the graph $\mathcal{G}$.

Now suppose we are interested in computing the probability

$$P(\mathbf{x}) = \partial_{\mathbf{x}}\Big[F(\mathbf{x})\Big] = \partial_{\mathbf{x}}\left[\prod_{s\in\mathcal{N}(\alpha)} T_s\left(\mathbf{x}_{\tau_s^\alpha}\right)\right].$$

Here, we take advantage of the fact that the graph has a tree structure, so that

$$\partial_{\mathbf{x}}\left[\prod_{s\in\mathcal{N}(\alpha)} T_s\left(\mathbf{x}_{\tau_s^\alpha}\right)\right] = \partial_{x_\alpha}\left[\prod_{s\in\mathcal{N}(\alpha)} \partial_{\mathbf{x}_{\tau_s^\alpha\setminus\alpha}}\Big[T_s\left(\mathbf{x}_{\tau_s^\alpha}\right)\Big]\right] = \partial_{x_\alpha}\left[\prod_{s\in\mathcal{N}(\alpha)} \mu_{s\to\alpha}\left(\mathbf{x}_{\tau_s^\alpha}\right)\right].$$

We have introduced the set of functions $\mu_{s\to\alpha}(\mathbf{x}) \equiv \mu_{s\to\alpha}\left(\mathbf{x}_{\tau_s^\alpha}\right)$ defined by

$$\mu_{s\to\alpha}(\mathbf{x}) \equiv \mu_{s\to\alpha}\left(\mathbf{x}_{\tau_s^\alpha}\right) = \partial_{\mathbf{x}_{\tau_s^\alpha\setminus\alpha}}\Big[T_s\left(\mathbf{x}_{\tau_s^\alpha}\right)\Big], \tag{3}$$

where we have assumed that each of the derivatives/finite differences have been evaluated at the desired values $\mathbf{x}_{\tau_s^\alpha\setminus\alpha}$. By its definition, $\mu_{s\to\alpha}(\mathbf{x})$ only depends on variables in the subtree τ_s^α and corresponds to the higher-order derivative of the joint CDF with respect to variables in the subtree $\tau_s^\alpha \setminus \alpha$. We can thus view the functions $\mu_{s\to\alpha}(\mathbf{x})$ as messages being passed from each function node $s \in \mathcal{N}(\alpha)$ in the CDN to a neighboring variable node α.

We can now write $T_s\left(\mathbf{x}_{\tau_s^\alpha}\right)$ as a product of functions owing to the tree structure of the graph $\mathcal{G}$, so that

$$T_s\left(\mathbf{x}_{\tau_s^\alpha}\right) = \phi_s(x_\alpha, \mathbf{x}_{\mathcal{N}(s)\setminus\alpha}) \prod_{\beta\in\mathcal{N}(s)\setminus\alpha} T_\beta\left(\mathbf{x}_{\tau_\beta^s}\right), \tag{4}$$

where $\mathbf{x}_{\tau_\beta^s}$ denotes the vector of configurations for all variables in the subtree τ_β^s which is rooted at function node s and contains node β (Figure 19), and T_β is the product of all functions in the subtree τ_β^s. Substituting Equation (4) into Equation (3), we obtain

$$\begin{aligned}\mu_{s\to\alpha}(\mathbf{x}) \equiv \mu_{s\to\alpha}\left(\mathbf{x}_{\tau_s^\alpha}\right) &= \partial_{\mathbf{x}_{\tau_s^\alpha\setminus\alpha}}\left[\phi_s(x_\alpha,\mathbf{x}_{\mathcal{N}(s)\setminus\alpha})\prod_{\beta\in\mathcal{N}(s)\setminus\alpha}T_\beta\left(\mathbf{x}_{\tau_\beta^s}\right)\right]\\ &= \partial_{\mathbf{x}_{\mathcal{N}(s)\setminus\alpha}}\left[\phi_s(x_\alpha,\mathbf{x}_{\mathcal{N}(s)\setminus\alpha})\prod_{\beta\in\mathcal{N}(s)\setminus\alpha}\partial_{\mathbf{x}_{\tau_\beta^s\setminus\beta}}\left[T_\beta\left(\mathbf{x}_{\tau_\beta^s}\right)\right]\right]\\ &= \partial_{\mathbf{x}_{\mathcal{N}(s)\setminus\alpha}}\left[\phi_s(x_\alpha,\mathbf{x}_{\mathcal{N}(s)\setminus\alpha})\prod_{\beta\in\mathcal{N}(s)\setminus\alpha}\mu_{\beta\to s}\left(\mathbf{x}_{\tau_\beta^s}\right)\right]. \qquad (5)\end{aligned}$$

Here we have defined messages $\mu_{\beta\to s}(\mathbf{x}) \equiv \mu_{\beta\to s}\left(\mathbf{x}_{\tau_\beta^s}\right)$ from variable nodes to function nodes. Similar to the definition for $\mu_{s\to\alpha}(\mathbf{x})$, the message $\mu_{\beta\to s}(\mathbf{x})$ only depends on variables in the subtree τ_β^s and corresponds to the higher-order derivative of the joint CDF with respect to variables in the subtree $\tau_\beta^s \setminus \beta$.

Finally, to compute the messages $\mu_{\beta\to s}(\mathbf{x})$ from variables to functions, we can write each of the functions $T_\beta\left(\mathbf{x}_{\tau_\beta^s}\right)$ as a product such that

$$T_\beta\left(\mathbf{x}_{\tau_\beta^s}\right) = \prod_{s'\in\mathcal{N}(\beta)\setminus s} T_{s'}\left(\mathbf{x}_{\tau_{s'}^\beta}\right),$$

where $T_{s'}$ is defined identically to T_s above but for function node s'. Substituting this into the expression for $\mu_{\beta\to s}(\mathbf{x})$ in Equation (5) yields

$$\begin{aligned}\mu_{\beta\to s}(\mathbf{x}) = \partial_{\mathbf{x}_{\tau_\beta^s\setminus\beta}}\left[T_\beta\left(\mathbf{x}_{\tau_\beta^s}\right)\right] &= \prod_{s'\in\mathcal{N}(\beta)\setminus s}\partial_{\mathbf{x}_{\tau_{s'}^\beta\setminus\beta}}\left[T_{s'}\left(\mathbf{x}_{\tau_{s'}^\beta}\right)\right]\\ &= \prod_{s'\in\mathcal{N}(\beta)\setminus s}\mu_{s'\to\beta}(\mathbf{x}).\end{aligned}$$

Thus, to compute messages from variables to functions, we simply take the product of all incoming messages except for the message coming from the destination function node. As in the sum-product algorithm, variables with only two neighboring functions simply pass messages through unchanged. We see here that the process of differentiation in a CDN can be implemented as an algorithm in which we pass messages $\mu_{\alpha\to s}$ from variables to neighboring function nodes and messages $\mu_{s\to\alpha}$ from functions to neighboring variable nodes. Messages can be computed recursively from one another as described above: we start from an arbitrary root variable node α and propagate messages up from leaf nodes to the root node. As in the sum-product algorithm, leaf variable nodes α' send the message $\mu_{\alpha'\to s}(\mathbf{x}) = 1$ while leaf function nodes $\phi_s(x_{\alpha'})$ send the message $\mu_{s\to\alpha'}(\mathbf{x}) = \phi_s(x_{\alpha'})$.

The message-passing algorithm proceeds until messages have been propagated along every edge in the network and the root variable node has received all incoming messages from the remainder of the network. Once all messages have been sent, we can obtain the probability density of the

variables in the graph from differentiating the product of incoming messages at the root node α, so that

$$P(\mathbf{x}) = \partial_{x_\alpha}\left[\prod_{s\in\mathcal{N}(\alpha)} \mu_{s\to\alpha}(\mathbf{x})\right].$$

A.2 Ordinal Regression

In many domains, one is faced with the problem of predicting multinomial variables that can each take one of a finite number of values in some discrete set $\mathcal{X} = \{r_1, \cdots, r_K\}$ for some integer K. Such multinomial variables can then be distinguished as being of the type

- *Nominal*, or *categorical*, so that the set $\mathcal{X}$ does not admit an ordering of variable values.
- *Ordinal*, so that the set $\mathcal{X}$ admits a total ordering over variable values of the type $r_1 \prec \cdots \prec r_K$.

An example of a nominal variable is gender, such as $\mathcal{X} = \{Male, Female\}$ and an example of an ordinal variable is a grading scheme $\mathcal{X} = \{A, B, C, D\}$ so that the possible variable values satisfy the total ordering $D \prec C \prec B \prec A$.

In ordinal regression, the goal is to predict a discrete variable $y \in \{r_1, \cdots, r_K\}$ given a set of features $\mathbf{x}$, where $r_1 \prec \cdots \prec r_K$ are an ordered set of labels. Unlike the general problem of multiclass classification in which variables to be predicted are nominal, output labels in the setting of ordinal regression are not permutation-invariant and so any model for the problem should account for the orderings of the output variable values.

One model for performing ordinal regression is the *cumulative model* (McCullagh, 1980), which relates an input vector $\mathbf{x}$ to an ordinal output y via a function f and a set of *cutpoints* $\theta(r_1) < \cdots < \theta(r_K)$ along the real line $\mathbb{R}$ so that $y = r_k$ if $\theta(r_{k-1}) < f(\mathbf{x}) + \varepsilon \leq \theta(r_k)$, where ε is additive noise and we define $\theta(r_0) = -\infty, \theta(r_K) = \infty$ (Figure 20). If $P(\varepsilon)$ is the probability density function from which the noise variable ε is drawn, then we can write

$$\begin{aligned}
\mathbb{P}\big[y = r_k\big] &= \mathbb{P}\big[\theta(r_{k-1}) < f(\mathbf{x}) + \varepsilon \leq \theta(r_k)\big] \\
&= \mathbb{P}\big[\{\theta(r_{k-1}) - f(\mathbf{x}) < \varepsilon\} \bigcap \{\varepsilon \leq \theta(r_k) - f(\mathbf{x})\}\big] \\
&= F_\varepsilon(\theta(r_{k-1}) - f(\mathbf{x})) - F_\varepsilon(\theta(r_k) - f(\mathbf{x})),
\end{aligned}$$

where $F_\varepsilon \equiv F(\varepsilon)$ is the corresponding cumulative distribution function for $P(\varepsilon)$. The above equation defines a likelihood function for a given observed pair $(\mathbf{x}, y)$, so that the cutpoints $\theta(r_k)$ and the regression function $f(\mathbf{x})$ can subsequently be estimated from training data by maximizing the likelihood function with respect to the cutpoints $\theta(r_k)$ and the regression function $f(\mathbf{x})$.

A.3 Derivative-sum-product Message Updates for Learning to Rank in Multiplayer Games

Here we present the DSP algorithm for updating player ranks. Messages are ensured to be properly normalized locally by computing the constant $\mathcal{Z} = \lim_{z\to\infty} \mu(z)$ for each message and multiplying the message pair μ, λ by $\mathcal{Z}^{-1}$.

- Initialize for each player score node X_k:

$$\begin{aligned}
\mu_{X_k\to g_n}(x_k) &= s_k(x_k), \\
\lambda_{X_k\to g_n}(x_k) &= \partial_{x_k}\Big[s_k(x_k)\Big].
\end{aligned}$$

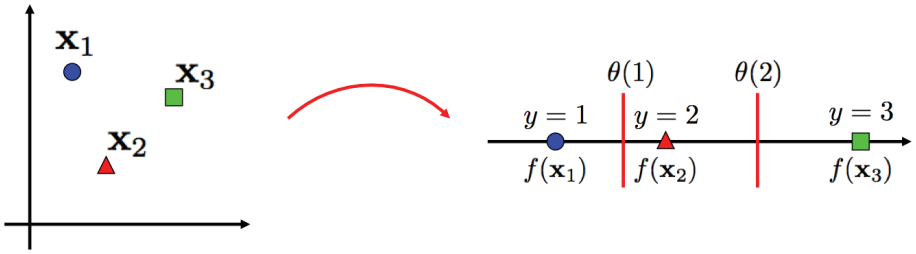


Figure 20: An illustration of the ordinal regression model. A given point has label $y = r_k$ if $\theta(r_{k-1}) < f(\mathbf{x}) + \varepsilon \leq \theta(r_k)$, where ε is a noise variable.

- Pass messages from function node g_n to team performance node R_n for neighboring player nodes $\mathbf{X}_n, n = 1, \cdots, N$:

$$\mu_{g_n \to R_n}(\mathbf{r}, \mathbf{x}) = \sum_{\substack{s,t | \mathbf{X}_s \cup \mathbf{X}_t = \mathbf{X}_n \\ \mathbf{X}_s \cap \mathbf{X}_t = \emptyset}} \partial_{\mathbf{x}_s} \left[g_n(\mathbf{x}_n, r_n) \right] \prod_{j | X_j \in \mathbf{X}_s} \mu_{X_j \to g_n}(x_j) \prod_{j | X_j \in \mathbf{X}_t} \lambda_{X_j \to g_n}(x_j),$$

$$\lambda_{g_n \to R_n}(\mathbf{r}, \mathbf{x}) = \sum_{\substack{s,t | \mathbf{X}_s \cup \mathbf{X}_t = \mathbf{X}_n \\ \mathbf{X}_s \cap \mathbf{X}_t = \emptyset}} \partial_{\mathbf{x}_s, r_n} \left[g_n(\mathbf{x}_n, r_n) \right] \prod_{j | X_j \in \mathbf{X}_s} \mu_{X_j \to g_n}(x_j) \prod_{j | X_j \in \mathbf{X}_t} \lambda_{X_j \to g_n}(x_j).$$

- Set $\mu_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) = \lambda_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) = 1$ for $n = 1$. Pass messages from team performance node R_n to neighboring team performance nodes R_{n+1} and function nodes $h_{n,n+1}$ for $n = 1, \cdots, N$:

$$\begin{aligned}
\mu_{R_n \to h_{n,n+1}}(\mathbf{r}, \mathbf{x}) &= \mu_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) \mu_{g_n \to R_n}(\mathbf{r}, \mathbf{x}), \\
\lambda_{R_n \to h_{n,n+1}}(\mathbf{r}, \mathbf{x}) &= \lambda_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) \mu_{g_n \to R_n}(\mathbf{r}, \mathbf{x}) \\
&\quad + \mu_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) \lambda_{g_n \to R_n}(\mathbf{r}, \mathbf{x}), \\
\mu_{h_{n,n+1} \to R_{n+1}}(\mathbf{r}, \mathbf{x}) &= \mu_{R_n \to h_{n,n+1}}(\mathbf{r}, \mathbf{x}) \partial_{r_n} \left[h_{n,n+1}(r_n, r_{n+1}) \right] \\
&\quad + \lambda_{R_n \to h_{n,n+1}}(\mathbf{r}, \mathbf{x}) h_{n,n+1}(r_n, r_{n+1}), \\
\lambda_{h_{n,n+1} \to R_{n+1}}(\mathbf{r}, \mathbf{x}) &= \mu_{R_n \to h_{n,n+1}}(\mathbf{r}, \mathbf{x}) \partial_{r_n, r_{n+1}} \left[h_{n,n+1}(r_n, r_{n+1}) \right] \\
&\quad + \lambda_{R_n \to h_{n,n+1}}(\mathbf{r}, \mathbf{x}) \partial_{r_{n+1}} \left[h_{n,n+1}(r_n, r_{n+1}) \right].
\end{aligned}$$

- Set $\mu_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}) = \lambda_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}) = 1$ for $n = N$. Pass messages from team performance node R_n to neighboring team performance nodes R_{n-1} and function nodes $h_{n-1,n}$ for $n =$

$1, \cdots, N$:

$$\begin{aligned}
\mu_{R_n \to h_{n-1,n}}(\mathbf{r}, \mathbf{x}) &= \mu_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}) \mu_{g_n \to R_n}(\mathbf{r}, \mathbf{x}), \\
\lambda_{R_n \to h_{n-1,n}}(\mathbf{r}, \mathbf{x}) &= \lambda_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}) \mu_{g_n \to R_n}(\mathbf{r}, \mathbf{x}) \\
&\quad + \mu_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}) \lambda_{g_n \to R_n}(\mathbf{r}, \mathbf{x}), \\
\mu_{h_{n-1,n} \to R_{n-1}}(\mathbf{r}, \mathbf{x}) &= \mu_{R_n \to h_{n-1,n}}(\mathbf{r}, \mathbf{x}) \partial_{r_n} \Big[h_{n-1,n}(r_{n-1}, r_n) \Big] \\
&\quad + \lambda_{R_n \to h_{n-1,n}}(\mathbf{r}, \mathbf{x}) h_{n-1,n}(r_{n-1}, r_n), \\
\lambda_{h_{n-1,n} \to R_{n-1}}(\mathbf{r}, \mathbf{x}) &= \mu_{R_n \to h_{n-1,n}}(\mathbf{r}, \mathbf{x}) \partial_{r_{n-1}, r_n} \Big[h_{n-1,n}(r_{n-1}, r_n) \Big] \\
&\quad + \lambda_{R_n \to h_{n-1,n}}(\mathbf{r}, \mathbf{x}) \partial_{r_{n-1}} \Big[h_{n-1,n}(r_{n-1}, r_n) \Big].
\end{aligned}$$

- Pass messages from each team performance node R_n to neighboring function nodes g_n:

$$\begin{aligned}
\mu_{R_n \to g_n}(\mathbf{r}, \mathbf{x}) &= \mu_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) \mu_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}), \\
\lambda_{R_n \to g_n}(\mathbf{r}, \mathbf{x}) &= \lambda_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) \mu_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}) \\
&\quad + \mu_{h_{n-1,n} \to R_n}(\mathbf{r}, \mathbf{x}) \lambda_{h_{n,n+1} \to R_n}(\mathbf{r}, \mathbf{x}).
\end{aligned}$$

- Pass messages from function nodes g_n to neighboring player score nodes X_k:

$$\begin{aligned}
\mu_{g_n \to X_k}(\mathbf{r}, \mathbf{x}) = &\sum_{\substack{s,t | \mathbf{X}_s \cup \mathbf{X}_t = \mathbf{X}_n \setminus X_k \\ \mathbf{X}_s \bigcap \mathbf{X}_t = \emptyset}} \prod_{j | X_j \in \mathbf{X}_s} \mu_{X_j \to g_n}(x_j) \prod_{j | X_j \in \mathbf{X}_t} \lambda_{X_j \to g_n}(x_j) \\
&\cdot \left(\partial_{\mathbf{x}_s} \Big[g_n(\mathbf{x}_n, r_n) \Big] \lambda_{R_n \to g_n}(\mathbf{r}, \mathbf{x}) + \partial_{\mathbf{x}_s, r_n} \Big[g_n(\mathbf{x}_n, r_n) \Big] \mu_{R_n \to g_n}(\mathbf{r}, \mathbf{x}) \right), \\
\lambda_{g_n \to X_k}(\mathbf{r}, \mathbf{x}) = &\sum_{\substack{s,t | \mathbf{X}_s \cup \mathbf{X}_t = \mathbf{X}_n \setminus X_k \\ \mathbf{X}_s \bigcap \mathbf{X}_t = \emptyset}} \prod_{j | X_j \in \mathbf{X}_s} \mu_{X_j \to g_n}(x_j) \prod_{j | X_j \in \mathbf{X}_t} \lambda_{X_j \to g_n}(x_j) \\
&\cdot \left(\partial_{\mathbf{x}_s, x_k} \Big[g_n(\mathbf{x}_n, r_n) \Big] \lambda_{R_n \to g_n}(\mathbf{r}, \mathbf{x}) + \partial_{\mathbf{x}_s, x_k, r_n} \Big[g_n(\mathbf{x}_n, r_n) \Big] \mu_{R_n \to g_n}(\mathbf{r}, \mathbf{x}) \right).
\end{aligned}$$

- For each player score node X_k,

$$\begin{aligned}
\mu_{X_k \to s_k}(\mathbf{r}, \mathbf{x}) &= \mu_{g_n \to X_k}(\mathbf{r}, \mathbf{x}), \\
\lambda_{X_k \to s_k}(\mathbf{r}, \mathbf{x}) &= \lambda_{g_n \to X_k}(\mathbf{r}, \mathbf{x}).
\end{aligned}$$

- Update player skill functions $s_k(x_k)$ using the multiplicative rule

$$s_k(x_k) \leftarrow s_k(x_k) \mu_{g_n \to X_k}(\mathbf{x}, \mathbf{r}).$$

References

Mathias Drton and Thomas S. Richardson. Binary models for marginal independence. *Journal of the Royal Statistical Society, Series B*, **70**:287-309, 2008.

Arpad E. Elo. *The Rating of Chess Players: Past and Present*. Arco Publishing, 1978.

Emil Julius Gumbel and C.K. Mustafi. Some analytical properties of bivariate extreme distributions. *Journal of the American Statistical Association*, **62**:569-588, 1967.

Ralf Herbrich, Thomas P. Minka and Thore Graepel. TrueSkill^{TM}: A Bayesian skill rating system. *Advances in Neural Information Processing Systems (NIPS)*, **19**:569-576, 2007.

Jim C. Huang and Brendan J. Frey. Cumulative distribution networks and the derivative-sum-product algorithm. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 290-297, Helsinki, Finland, 2008.

Jim C. Huang. *Cumulative Distribution Networks: Inference, Estimation and Applications of Graphical Models for Cumulative Distribution Functions*. PhD thesis, University of Toronto, Toronto, Ontario, Canada, 2009.

Goran Kauermann. On a dualization of graphical Gaussian models. *Scandinavian Journal of Statistics*, **23**:105-116, 1996.

Frank R. Kschischang, Brendan J. Frey and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, **47(2)**:498-519, 2001.

Steffen L. Lauritzen and David J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society Series B (Methodological)* **50(2)**:157-224,1988.

Steffen L. Lauritzen. *Graphical Models*. Oxford University Press Inc., New York, New York, 1996.

Erich L. Lehmann. Ordered families of distributions. *The Annals of Mathematical Statistics*, **26**:399-419, 1955.

Peter McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society, Series B (Methodological)*, **42(2)**:109-142, 1980.

Thomas P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 362-369, Seattle, Washington, 2001.

Radford M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, 1993.

Roger B. Nelsen *An Introduction to Copulas*. Springer, New York, New York, 1999.

Athanasios Papoulis and S. Unnikrishna Pillai. *Probability, Random Variables and Stochastic Processes*. McGraw Hill, New York, New York, 2001.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Francisco, California, 1988.

Thomas S. Richardson and Peter Spirtes. Ancestral graph Markov models. *Annals of Statistics*, **30**:962-1030, 2002.

Thomas S. Richardson. Markov properties for acyclic directed mixed graphs. *Scandinavian Journal of Statistics*, **30**:145-157, 2003.

Moshe Shaked and J. George Shanthikumar. *Stochastic Orders and Their Applications*. Academic Press, San Diego, California, 1994.

Ricardo Silva and Zoubin Ghahramani. Factorial mixture of Gaussians and the marginal independence model. In *Proceedings of the Twelfth Annual Conference on Artificial Intelligence and Statistics (AISTATS)*, *JMLR: W & CP*, **5**:520-527, 2009.

Ricardo Silva and Zoubin Ghahramani. The hidden life of latent variables: Bayesian learning with mixed graph models. *Journal of Machine Learning Research (JMLR)*, **10**:1187-1238, 2009.

Alan Stuart and Keith Ord. *Kendall's Advanced Theory of Statistics, Volume 1: Distribution Theory*. Edward Arnold, London, UK, 1994.

Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer, New York, New York, 2004.

Yair Weiss and William T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, **13**:2173-2200, 2001.

Journal of Machine Learning Research 12 (2011) 349-384 Submitted 6/10; Revised 12/10; Published 2/11

Models of Cooperative Teaching and Learning

Sandra Zilles ZILLES@CS.UREGINA.CA
Department of Computer Science
University of Regina
Regina, SK, Canada, S4S 0A2

Steffen Lange S.LANGE@FBI.H-DA.DE
Department of Computer Science
Darmstadt University of Applied Sciences
Haardtring 100, 64295 Darmstadt, Germany

Robert Holte HOLTE@CS.UALBERTA.CA
Department of Computing Science
University of Alberta
Edmonton, AB, Canada, T6G 2E8

Martin Zinkevich MAZ@YAHOO-INC.COM
Yahoo! Inc.
701 First Avenue
Sunnyvale, CA 94089, USA

Editor: Nicolò Cesa-Bianchi

Abstract

While most supervised machine learning models assume that training examples are sampled at random or adversarially, this article is concerned with models of learning from a cooperative teacher that selects "helpful" training examples. The number of training examples a learner needs for identifying a concept in a given class C of possible target concepts (sample complexity of C) is lower in models assuming such teachers, that is, "helpful" examples can speed up the learning process.

The problem of how a teacher and a learner can cooperate in order to reduce the sample complexity, yet without using "coding tricks", has been widely addressed. Nevertheless, the resulting teaching and learning protocols do not seem to make the teacher select intuitively "helpful" examples. The two models introduced in this paper are built on what we call *subset teaching sets* and *recursive teaching sets*. They extend previous models of teaching by letting both the teacher and the learner exploit *knowing* that the partner is cooperative. For this purpose, we introduce a new notion of "coding trick"/"collusion".

We show how both resulting sample complexity measures (the *subset teaching dimension* and the *recursive teaching dimension*) can be arbitrarily lower than the classic teaching dimension and known variants thereof, without using coding tricks. For instance, monomials can be taught with only two examples independent of the number of variables.

The subset teaching dimension turns out to be nonmonotonic with respect to subclasses of concept classes. We discuss why this nonmonotonicity might be inherent in many interesting cooperative teaching and learning scenarios.

Keywords: teaching dimension, learning Boolean functions, interactive learning, collusion

©2011 Sandra Zilles, Steffen Lange, Robert Holte and Martin Zinkevich.

1. Introduction

A central problem in machine learning is that learning algorithms often require large quantities of data. Data may be available only in limited quantity, putting successful deployment of standard machine learning techniques beyond reach. This problem is addressed by models of machine learning that are enhanced by interaction between a learning algorithm (learner, for short) and its environment, whose main purpose is to reduce the amount of data needed for learning. Interaction here means that at least one party actively controls which information is exchanged about the target object to be learned. Most classic machine learning models address the "average case" of data presentation to a learner (labeled examples are drawn independently at random from some distribution) or even the "worst case" (examples are drawn in an adversarial fashion). This results in the design of learners requiring more data than would be necessary under more optimistic (and often realistic) assumptions. As opposed to that, interactive learning refers to a "good case" in which representative examples are selected, whereby the number of examples needed for successful learning may shrink significantly.

Interactive machine learning is of high relevance for a variety of applications, for example, those in which a human interacts with and is observed by a learning system. A systematic and formally founded study of interactive learning is expected to result in algorithms that can reduce the cost of acquiring training data in real-world applications.

This paper focusses on particular formal models of interactive concept learning. Considering a finite instance space and a class of (thus finite) concepts over that space, it is obvious that each concept can be uniquely determined if enough examples are known. Much less obvious is how to minimize the number of examples required to identify a concept, and with this aim in mind models of *cooperative learning* and learning from *good examples* were designed and analyzed. The selection of good examples to be presented to a learner is often modeled using a teaching device (teacher) that is assumed to be benevolent by selecting examples expediting the learning process (see, for instance, Angluin and Kriķis, 1997; Jackson and Tomkins, 1992; Goldman and Mathias, 1996; Mathias, 1997).

Throughout this paper we assume that teaching/learning proceeds in a simple protocol; the teacher presents a batch of labeled examples (that is, a set of instances, each paired with a label 1 or 0, according to whether or not the instance belongs to the target concept) to the learner and the learner returns a concept it believes to be the target concept. If the learner's conjecture is correct, the exchange is considered successful. The sample size, that is, the number of examples the teacher presents to the learner, is the object of optimization; in particular we are concerned with the worst case sample size measured over all concepts in the underlying class C of all possible target concepts. Other than that, computational complexity issues are not the focus of this paper.

A typical question is *How can a teacher and a learner cooperatively minimize the worst case sample size without using coding tricks?*—a coding trick being, for example, any *a priori* agreement on encoding concepts in examples, depending on the concept class C. For instance, if teacher and learner agreed on a specific order for the concept representations and the instances and agreed to use the j^{th} instance in this ordering to teach the j^{th} concept, that would be a coding trick. In practice, the teacher and the learner might not be able to agree on such an order, for instance, if the teacher is a human who does not have the same representation of a concept as the machine has. There is so far no generally accepted definition of the term "coding trick" (sometimes also called "collusion"); the

reader is referred to Angluin and Kriķis (1997), Ott and Stephan (2002) and Goldman and Mathias (1996). It is often more convenient to define what constitutes a valid pair of teacher and learner.

The most popular teaching model is the one introduced by Goldman and Mathias (1996). Here a team of teacher and learner is considered valid if, for every concept c in the underlying class C the following properties hold.

- The teacher selects a set S of labeled examples consistent with c.
- On input of *any superset* of S of examples that are labeled consistently with c, the learner will return a hypothesis representing c.

The idea behind this definition is that the absence of examples in the sample S cannot be used for encoding knowledge about the target concept. This is completely in line with notions of inductive inference from good examples, see Freivalds et al. (1993) and Lange et al. (1998).

One way for a teacher and a learner to form a valid team under these constraints is for the teacher to select, for every concept $c \in C$, a sample S that is consistent with c but inconsistent with every other concept in C. The size of the minimum such sample is called the *teaching dimension* of c in C. The teaching dimension of the class C is the maximum teaching dimension over all concepts in C. For more information, the reader is referred to the original literature on teaching dimension and variants thereof (Shinohara and Miyano, 1991; Goldman and Kearns, 1995; Anthony et al., 1992).

The teaching dimension however does not always seem to capture the intuitive idea of cooperation in teaching and learning. Consider the following simple example. Let C_0 consist of the empty concept and all singleton concepts over a given instance space $X = \{x_1, \ldots, x_n\}$. Each singleton concept $\{x_i\}$ has a teaching dimension of 1, since the single positive example $(x_i, +)$ is sufficient for determining $\{x_i\}$. This matches our intuition that concepts in this class are easy to teach. In contrast to that, the empty concept has a teaching dimension of n—every example has to be presented. However, if the learner assumed the teacher was cooperative—and would therefore present a positive example if the target concept was non-empty—the learner could confidently conjecture the empty concept upon seeing just one negative example.

Let us extend this reasoning to a slightly more complex example, the class of all boolean functions that can be represented as a monomial over m variables ($m = 4$ in this example). Imagine yourself in the role of a learner knowing your teacher will present helpful examples. If the teacher sent you the examples

$$(0100, +), (0111, +),$$

what would be your conjecture? Presumably most people would conjecture the monomial $M \equiv \overline{v_1} \wedge v_2$, as does for instance the algorithm proposed by Valiant (1984). Note that this choice is not uniquely determined by the data: the empty (always true) monomial and the monomials $\overline{v_1}$ and v_2 are also consistent with these examples. And yet M seems the best choice, because we'd think the teacher would not have kept any bit in the two examples constant if it was not in the position of a relevant variable. In this example, the natural conjecture is the most specific concept consistent with the sample, but that does not, in general, capture the intuitive idea of cooperative learning. In particular, if, instead of the class of all monomials, the class of all complements of these concepts over the same instance space is chosen, then a cooperative teacher and learner would need only two negatively labeled example for teaching the complement of the concept associated with $\overline{v_1} \wedge v_2$, which is now the least specific concept in the class. Going further, one could swap $+$ for $-$ and vice versa only for some of the instances. In effect, only the labels in the examples chosen by the teacher

would change, but not the instances as such. The concepts guessed by the learner would then be neither the most specific nor the least specific concepts.

Could the learner's reasoning about the teacher's behavior in these examples be implemented without a coding trick? We will argue below that, for a very intuitive, yet mathematically rigorous definition of coding tricks, no coding trick is necessary to achieve exactly this behavior of teacher and learner; there are general strategies that teachers and learners can independently implement to cooperatively learn any finite concept class. When applied to the class of monomials this principle enables any monomial to be learned from just two examples, regardless of the number m of variables.

Our approach is to define a new model of cooperation in learning, based on the idea that each partner in the cooperation tries to reduce the sample size by exploiting the assumption that the other partner does so. If this idea is iteratively propagated by both partners, one can refine teaching sets iteratively ending up with a framework for highly efficient teaching and learning without any coding tricks. It is important to note that teacher and learner do not agree on any order of the concept class or any order of the instances. All they know about each others' strategies is a general assumption about how cooperation should work independent of the concept class or its representation.

We show that the resulting variant of the teaching dimension—called the *subset teaching dimension (STD)*—is not only a uniform lower bound of the teaching dimension but can be constant where the original teaching dimension is exponential, even in cases where only one iteration is needed. For example, as illustrated above, the STD of the class of monomials over $m \geq 2$ variables is 2, in contrast to its original teaching dimension of 2^m.

Some examples however will reveal a nonmonotonicity of the subset teaching dimension: some classes possess subclasses with a higher subset teaching dimension, which is at first glance not very intuitive. We will explain below why in a cooperative model such a nonmonotonicity does not have to contradict intuition; additionally we introduce a second model of cooperative teaching and learning, that results in a monotonic dimension, called the *recursive teaching dimension (RTD)*. Recursive teaching is based on the idea to let the teacher and the learner exploit a hierarchical structure that is intrinsic in the concept class. The canonical hierarchy associated with a concept class C is a nesting of C, starting with the class of all concepts in C that are easiest to teach (i.e., have the lowest teaching dimension) and then applying the nesting process recursively to the remaining set of concepts. At every stage, the recursive teaching sets for the concepts that are easiest to teach are the teaching sets for these concepts with respect to the class of remaining concepts. The recursive teaching dimension is the size of the largest recursive teaching set constructed this way.

The RTD-model is not as intuitive a model of cooperative teaching and learning as the STD-model is, but it displays a surprising set of properties. Besides its monotonicity, the RTD corresponds to teacher-learner protocols that do not violate Goldman and Mathias's definition of teaching and learning without coding tricks. Nevertheless, substantial improvements over the classical teaching dimension are obtained. A recent study furthermore shows that the recursive teaching dimension is a combinatorial parameter of importance when analyzing the complexity of learning problems from the perspective of active learning, teaching, learning from random examples, and sample compression, see Doliwa et al. (2010).

Both our teaching protocols significantly improve sample efficiency compared to previously studied variants of the teaching dimension.

This paper is a correction and extension of an earlier publication (Zilles et al., 2008). In this earlier publication, both Proposition 5(1) and the conjecture in Lemma 23 were wrong.

2. Related Work

The problem of defining what are "good" or "helpful" examples in learning has been studied in several fields of learning theory.

Various learning models, which each involve one particular type of teacher, were proposed by Goldman and Kearns (1995), Goldman and Mathias (1996), Mathias (1997), Jackson and Tomkins (1992), Shinohara and Miyano (1991), Angluin and Kriķis (1997), Angluin and Kriķis (2003), Balbach (2008) and Kobayashi and Shinohara (2009); these studies mostly focus on learning boolean functions. See also Balbach and Zeugmann (2009) for a recent survey. The teaching dimension model, independently introduced by Goldman and Kearns (1991; 1995) and by Shinohara and Miyano (1991), is concerned with the sample complexity of teaching arbitrary consistent learners. Samples that will allow any consistent learner to identify the target concept are called *teaching sets*; the maximum size of minimal teaching sets of all concepts in the underlying concept class C is called the *teaching dimension* of C. The problem of avoiding unfair "coding tricks" between teachers and learners is addressed in particular by Goldman and Mathias (1996) with the introduction of a formal notion of collusion-free learning. It is known that computing (the size of) minimal teaching sets is in general intractable, see Servedio (2001), which is one reason why the polynomial-time models introduced by Jackson and Tomkins (1992) are of interest. Jackson and Tomkins no longer require that teachers choose samples that make any consistent learner successful; they rather focus on specific teacher/learner pairs. Loosening the requirement of learners being consistent, Kobayashi and Shinohara (2009) analyze how restrictions on the number of examples given by the teacher influence the worst-case error of the hypothesis returned by a learner.

The teaching dimension was analyzed in the context of online learning, see Ben-David and Eiron (1998) and Rivest and Yin (1995), and in the model of learning from queries, for example, by Hegedűs (1995) and by Hanneke (2007), with a focus on active learning in the PAC framework. In contrast to these models, in inductive inference the learning process is not necessarily considered to be finite. Approaches to defining learning infinite concepts from good examples (Freivalds et al., 1993; Lange et al., 1998) do not focus on the size of a finite sample of good examples, but rather on characterizing the cases in which learners can identify concepts from only finitely many examples.

One of the two approaches we present in this paper is mainly based on an idea by Balbach (2008). He defined and analyzed a model in which, under the premise that the teacher uses a minimal teaching set (as defined by Goldman and Kearns, 1991, 1995) as a sample, a learner can reduce the size of a required sample by eliminating concepts which possess a teaching set smaller than the number of examples provided by the teacher so far. Iterating this idea, the size of the teaching sets might be gradually reduced significantly. Though our approach is syntactically quite similar to Balbach's, the underlying idea is a different one (we do not consider elimination by the sample size but elimination by the sample content as compared to all possible teaching sets). The resulting variant of the teaching dimension in general yields different performance results in terms of sample size than Balbach's model does.

3. The Teaching Dimension and the Balbach Teaching Dimension

Let $\mathbb{N}$ denote the set of all non-negative integers, $\emptyset$ denote the empty set, and $|M|$ denote the cardinality of a finite set M. For any $k \in \mathbb{N}$, the power set of $\{1,\ldots,k\}$ will be denoted by $2^{[k]}$.

In the models of teaching and learning to be defined below, we will always assume that the goal in an interaction between a teacher and a learner is to make the learner identify a (finite) concept over a (finite) instance space X.

Most of the recent work on teaching (cf. Balbach, 2008; Zilles et al., 2008; Balbach and Zeugmann, 2009; Kobayashi and Shinohara, 2009) defines a concept simply as a subset of X and a concept class as a set of subsets of X. In effect, this is exactly the definition we would need for introducing the teaching models we define below. However, the definition and discussion of the notion of collusion (i.e., the conceptualization of what constitutes a coding trick), see Section 4, motivates a more general definition of concepts and concept classes. This more general definition considers the instance space X as an ordered set and every concept class C as an ordered set of subsets of X.

To formalize this, let $X = \{1,\ldots,n\}$. Concepts and concept classes are defined as follows.

Definition 1 *Let $z \in \mathbb{N}$.*

A concept class of cardinality z is defined by an injective mapping $C : \{1,\ldots,z\} \to 2^{[n]}$. Every $i \in \{1,\ldots,z\}$ and thus every concept $C(i)$ is associated with a membership function on $X = \{1,\ldots,n\}$, given by $C(i)(j) = +$ if $j \in C(i)$, and $C(i)(j) = -$ if $j \notin C(i)$, where $1 \leq j \leq n$. Thus a concept class C of cardinality $z \in \mathbb{N}$ is represented as a matrix $(C(i)(j))_{1\leq i\leq z, 1\leq j\leq n}$ over $\{+,-\}$.

$\mathcal{C}_z$ denotes the collection of all concept classes of cardinality z. $\mathcal{C} = \bigcup_{z\in\mathbb{N}} \mathcal{C}_z$ denotes the collection of all concept classes (of any cardinality).

Consequently, concepts and concept classes considered below will always be finite.

Definition 2 *Let $z \in \mathbb{N}$ and $C \in \mathcal{C}_z$.*

A sample is a set $S = \{(j_1,l_1),\ldots,(j_r,l_r)\} \subseteq X \times \{+,-\}$, where every element (j,l) of S is called a (labeled) example.

Let $i \in \{1,\ldots,z\}$. $C(i)$ is consistent with S (and S is consistent with $C(i)$) if $C(i)(j_t) = l_t$ for all $t \in \{1,\ldots,r\}$. Denote

$$Cons(S,C) = \{i \in \{1,\ldots,z\} \mid C(i) \text{ is consistent with } S\}.$$

The power set of $\{1,\ldots,n\} \times \{+,-\}$, that is, the set of all samples, is denoted by $\mathcal{S}$.

3.1 Protocols for Teaching and Learning in General

In what follows, we assume that a teacher selects a sample for a given target concept and that a learner, given any sample S, always returns an index of a concept from the underlying concept class C. Formally, if $z \in \mathbb{N}$ and $(C(i)(j))_{1\leq i\leq z, 1\leq j\leq n}$ is a concept class in $\mathcal{C}_z$, a *teacher* for C is a function $\tau : \{1,\ldots,z\} \to \mathcal{S}$; a *learner* for C is a function $\lambda : \mathcal{S} \to \{1,\ldots,z\}$.

In order to constrain the definition of validity of a teacher/learner pair to a desired form of interaction in a learning process, the notion of adversaries will be useful. Adversaries will be considered third parties with the option to modify a sample generated by a teacher before this sample is given to a learner. Formally, an *adversary* is a relation $Ad \subseteq \mathcal{S}^3$. Intuitively, if $(\tau(i),C(i),S) \in Ad$ for some $i \in \{1,\ldots,z\}$ and some teacher τ for $C = (C(i)(j))_{1\leq i\leq z, 1\leq j\leq n}$, then the adversary has the option to modify $\tau(i)$ to S and the learner communicating with τ will get S rather than $\tau(i)$ as input. A special adversary is the so-called *trivial adversary* Ad^*, which satisfies $(S_1,S_2,S) \in Ad^*$ if and only if $S_1 = S$. This adversary does not modify the samples generated by the teacher at all.

All teaching and learning models introduced below will involve a very simple *protocol* between a teacher and a learner (and an adversary).

Definition 3 *Let P be a mapping that maps every concept class $C \in \mathcal{C}$ to a pair $P(C) = (\tau, \lambda)$ where τ is a teacher for C and λ is a learner for C. P is called a protocol; given $C \in \mathcal{C}$, the pair $P(C)$ is called a protocol for C.*

1. *Let $z \in \mathbb{N}$ and let $C \in \mathcal{C}_z$ be a concept class. Let Ad_C be an adversary. $P(C) = (\tau, \lambda)$ is called successful for C with respect to Ad_C if $\lambda(S) = i$ for all pairs (i, S) where $i \in \{1, \ldots, z\}$, $S \in \mathcal{S}$, and $(\tau(i), C(i), S) \in Ad_C$.*

2. *Let $\mathcal{A} = (Ad_C)_{C \in \mathcal{C}}$ be a family of adversaries. P is called successful with respect to $\mathcal{A}$ if, for all $C \in \mathcal{C}$, $P(C)$ is successful for C with respect to Ad_C.*

Protocols differ in the strategies according to which the teacher and the learner operate, that is, according to which the teacher selects a sample and according to which the learner selects a concept.

In all protocols considered below, teachers always select consistent samples for every given target concept and learners, given any sample S, always return a concept consistent with S if such a concept exists in the underlying class C. Formally, all teachers τ for a concept class $C \in \mathcal{C}_z$ will fulfill $i \in Cons(\tau(i), C)$ for all $i \in \{1, \ldots, z\}$; all learners λ for a class C will fulfill $\lambda(S) \in Cons(S, C)$ for all $S \in \mathcal{S}$ with $Cons(S, C) \neq \emptyset$. Moreover, all the adversaries Ad we present below will have the following property:

$$\text{for any three samples } S_1, S_2, S \in \mathcal{S}, \text{ if } (S_1, S_2, S) \in Ad \text{ then } S_1 \subseteq S \subseteq S_2\,.$$

However, this does not mean that we consider other forms of teachers, learners, or adversaries illegitimate. They are just beyond the scope of this paper.

The goal in sample-efficient teaching and learning is to design protocols that, for every concept class C, are successful for C while reducing the (worst-case) size of the samples the teacher presents to the learner for any target concept in C. At the same time, by introducing adversaries, one tries to avoid certain forms of collusion, an issue that we will discuss in Section 4.

3.2 Protocols Using Minimal Teaching Sets and Balbach Teaching Sets

The fundamental model of teaching we consider here is based on the notion of *minimal teaching sets*, which is due to Goldman and Kearns (1995) and Shinohara and Miyano (1991).

Let $z \in \mathbb{N}$ and let $C \in \mathcal{C}_z$ be a concept class. Let S be a sample. S is called a *teaching set* for i with respect to C if $Cons(S, C) = \{i\}$. A teaching set allows a learning algorithm to uniquely identify a concept in the concept class C. Teaching sets of minimal size are called *minimal teaching sets*. The *teaching dimension* of i in C is the size of such a minimal teaching set, that is, $TD(i, C) = \min\{|S| \mid Cons(S, C) = \{i\}\}$, the worst case of which defines the teaching dimension of C, that is, $TD(C) = \max\{TD(i, C) \mid 1 \leq i \leq z\}$. To refer to the set of all minimal teaching sets of i with respect to C, we use

$$TS(i, C) = \{S \mid Cons(S, C) = \{i\} \text{ and } |S| = TD(i, C)\}\,.$$

Minimal teaching sets induce the following protocol.

Protocol 4 *Let P be a protocol. P is called a teaching set protocol (TS-protocol for short) if the following two properties hold for every $C \in \mathcal{C}$, where $P(C) = (\tau, \lambda)$.*

1. $\tau(i) \in TS(i,C)$ *for all* $i \in \{1,\ldots,z\}$,

2. $\lambda(S) \in Cons(S,C)$ *for all* $S \in \mathcal{S}$ *with* $Cons(S,C) \neq \emptyset$.

This protocol is obviously successful with respect to the family consisting only of the trivial adversary. The teaching dimension of a concept class C is then a measure of the worst case sample size required in this protocol with respect to Ad^* when teaching/learning any concept in C.

The reason that, for every concept class $C \in \mathcal{C}_z$, the protocol $P(C)$ is successful (with respect to Ad^*) is simply that a teaching set eliminates all but one concept due to inconsistency. However, if the learner knew $TD(i,C)$ for every $i \in \{1,\ldots,z\}$ then sometimes concepts could also be eliminated by the mere number of examples presented to the learner. For instance, assume a learner knows that all but one concept $C(i)$ have a teaching set of size one and that the teacher will teach using teaching sets. After having seen 2 examples, no matter what they are, the learner could eliminate all concepts but $C(i)$. This idea, referred to as elimination by sample size, was introduced by Balbach (2008). If a teacher knew that a learner eliminates by consistency and by sample size then the teacher could consequently reduce the size of some teaching sets (e.g., here, if $TD(i,C) \geq 3$, a new "teaching set" for i could be built consisting of only 2 examples).

More than that—this idea is iterated by Balbach (2008): if the learner knew that the teacher uses such reduced "teaching sets" then the learner could adapt his assumption on the size of the samples to be expected for each concept, which could in turn result in a further reduction of the "teaching sets" by the teacher and so on. The following definition captures this idea formally.

Definition 5 (Balbach, 2008)
Let $z \in \mathbb{N}$ *and let* $C \in \mathcal{C}_z$ *be a concept class. Let* $i \in \{1,\ldots,z\}$ *and* S *a sample. Let* $BTD^0(i,C) = TD(i,C)$. *We define iterated dimensions for all* $k \in \mathbb{N}$ *as follows.*

- $Cons_{size}(S,C,k) = \{i \in Cons(S,C) \mid BTD^k(i,C) \geq |S|\}$.
- $BTD^{k+1}(i,C) = \min\{|S| \mid Cons_{size}(S,C,k) = \{i\}\}$

Let κ *be minimal such that* $BTD^{\kappa+1}(i,C) = BTD^{\kappa}(i,C)$ *for all* $i \in \{1,\ldots,z\}$. *The Balbach teaching dimension* $BTD(i,C)$ *of* i *in* C *is defined by* $BTD(i,C) = BTD^{\kappa}(i,C)$ *and the Balbach teaching dimension* $BTD(C)$ *of the class* C *is* $BTD(C) = \max\{BTD(i,C) \mid 1 \leq i \leq z\}$.[1] *For every* $i \in \{1,\ldots,z\}$ *we define*

$$BTS(i,C) = \{S \mid Cons_{size}(S,C,\kappa) = \{i\} \text{ and } |S| = BTD(i,C)\}$$

and call every set in $BTS(i,C)$ *a minimal Balbach teaching set of* i *with respect to* C.

By $Cons_{size}(S,C)$ *we denote the set* $Cons_{size}(S,C,\kappa)$.

The Balbach teaching dimension measures the sample complexity of the following protocol with respect to the trivial adversary.

Protocol 6 *Let* P *be a protocol.* P *is called a Balbach teaching set protocol (BTS-protocol for short) if the following two properties hold for every* $C \in \mathcal{C}$, *where* $P(C) = (\tau,\lambda)$.

1. $\tau(i) \in BTS(i,C)$ *for all* $i \in \{1,\ldots,z\}$,

1. Balbach (2008) denotes this by *IOTTD*, called iterated optimal teacher teaching dimension; we deviate from this notation for the sake of convenience.

2. $\lambda(S) \in \{i \mid$ *there is some* $S' \in BTS(i,C)$ *such that* $S' \subseteq S\}$ *for all* $S \in \mathcal{S}$ *that contain a set* $S' \in BTS(i,C)$ *for some* $i \in \{1,\ldots,z\}$.

Obviously, $BTD(C) \leq TD(C)$ for every concept class $C \in \mathcal{C}$. How much the sample complexity can actually be reduced by a cooperative teacher/learner pair according to this "elimination by sample size" principle, is illustrated by the concept class C_0 which consists of the empty concept and all singleton concepts over X. The teaching dimension of this class is n, whereas the *BTD* is 2.

3.3 Teaching Monomials

A standard example of a class of boolean functions studied in learning theory is the class $\mathcal{F}_m$ of monomials over a set $\{v_1,\ldots,v_m\}$ of m variables, for any $m \geq 2$.[2] Usually, this class is just defined by choosing $X = \{0,1\}^m$ as the underlying instance space. Then, for any monomial M, the corresponding concept is defined as the set of those assignments in $\{0,1\}^m$ for which M evaluates positively. Within our more general notion of concept classes, there is more than just one class of all monomials over m variables (which we will later consider as equivalent). This is due to distinguishing different possible orderings over X and over the class of monomials itself.

Definition 7 *Let* $m \in \mathbb{N}$, $m \geq 2$ *and assume* $n = 2^m$, *that is,* $X = \{1,\ldots,2^m\}$.

Let $\mathrm{bin} : \{1,\ldots,2^m\} \to \{0,1\}^m$ *be a bijection, that is, a repetition-free enumeration of all bit strings of length* m. *Let* $\mathrm{mon} : \{1,\ldots,3^m\} \to \mathcal{F}_m$ *be a bijective enumeration of all monomial functions over* m *variables* $v_1,\ldots,v_m$.

A mapping $C : \{1,\ldots,3^m\} \to 2^{[2^m]}$ *is called a concept class of all monomials over* m *variables if, for all* $i \in \{1,\ldots,3^m\}$ *and all* $j \in \{1,\ldots,2^m\}$,

$$C(i)(j) = \begin{cases} +, & \text{if } \mathrm{mon}(i) \text{ evaluates to TRUE when assigning } \mathrm{bin}(j) \text{ to } (v_1,\ldots,v_m), \\ -, & \text{if } \mathrm{mon}(i) \text{ evaluates to FALSE when assigning } \mathrm{bin}(j) \text{ to } (v_1,\ldots,v_m). \end{cases}$$

It turns out that a class of all monomials contains only one concept for which the *BTD*-iteration yields an improvement.

Theorem 8 (Balbach, 2008) *Let* $m \in \mathbb{N}$, $m \geq 2$. *Let* $C : \{1,\ldots,3^m\} \to 2^{[2^m]}$ *be a concept class of all monomials over* m *variables. Let* $i^* \in \{1,\ldots,3^m\}$ *with* $C(i^*) = \emptyset$ *be an index for the concept representing the contradictory monomial.*

1. $BTD(i^*,C) = m+2 < 2^m = TD(i^*,C)$.

2. $BTD(i,C) = TD(i,C)$ *for all* $i \in \{1,\ldots,3^m\} \setminus \{i^*\}$.

The intuitive reason for $BTD(i^*,C) = m+2$ in Theorem 8 is that samples for $C(i^*)$ of size $m+1$ or smaller are consistent also with monomials different from $C(i^*)$, namely those monomials that contain every variable exactly once (each such monomial is positive for exactly one of the 2^m instances). These other monomials hence cannot be eliminated—neither by size nor by inconsistency.

2. A monomial over $\{v_1,\ldots,v_m\}$ is a conjunction of literals over $\{v_1,\ldots,v_m\}$, also called a 1-CNF or a 1-term DNF.

4. Avoiding Coding Tricks

Intuitively, the trivial adversary of course does not prevent teacher and learner from using coding tricks. One way of defining what a coding trick is—or what a valid (collusion-free) behaviour of a teacher/learner is supposed to look like—is to require success with respect to a specific non-trivial type of adversary.

Goldman and Mathias (1996) called a pair of teacher and learner valid for a concept class $C \in \mathcal{C}_z$ if, for every concept $C(i)$ in the class C, the following properties hold.

- The teacher selects a set S of labeled examples consistent with $C(i)$.
- On input of *any superset* of S of examples that are labeled consistently with $C(i)$, the learner will return a hypothesis representing $C(i)$.

In other words, they considered a teacher-learner pair (τ,λ) a valid protocol for C if and only if it is successful with respect to *any* adversary Ad_C that fulfills $\tau(i) \subseteq S \subseteq C(i)$ for all $i \in \{1,\ldots,z\}$ and all $S \in \mathcal{S}$ with $(\tau(i),C(i),S) \in Ad_C$.

Obviously, teacher/learner pairs using minimal teaching sets according to the *TS*-protocol (Protocol 4) are valid in this sense.

Theorem 9 *Let $z \in \mathbb{N}$ and let $C \in \mathcal{C}_z$ be a concept class. Let τ be a teacher for C, λ a learner for C. If (τ,λ) is a TS-protocol for C then (τ,λ) is successful with respect to any adversary Ad_C that fulfills $\tau(i) \subseteq S \subseteq C(i)$ for all $i \in \{1,\ldots,z\}$.*

Proof. Immediate from the definitions. □

Not only the protocol based on the teaching dimension (Protocol 4), but also the protocol based on the Balbach teaching dimension (Protocol 6) yields only valid teacher/learner pairs according to this definition—a consequence of Theorem 10.

Theorem 10 *Let $z \in \mathbb{N}$ and let $C \in \mathcal{C}_z$ be a concept class. Let $i \in \{1,\ldots,z\}$, $S \in BTS(i,C)$, and $T \supseteq S$ such that $i \in Cons(T,C)$. Then there is no $i' \in Cons(T,C)$ such that $i \neq i'$ and $S' \subseteq T$ for some $S' \in BTS(i',C)$.*

Proof. Assume there is some $i' \in Cons(T,C)$ such that $i \neq i'$ and some $S' \in BTS(i',C)$ such that $S' \subseteq T$. Since both $C(i)$ and $C(i')$ are consistent with T and both S and S' are subsets of T, we have $i \in Cons(S',C)$ and $i' \in Cons(S,C)$. Now let $\kappa \geq 1$ be minimal such that $BTD^{\kappa}(i^*,C) = BTD(i^*,C)$ for all $i^* \in C$. From $i' \in Cons(S,C)$ and $S \in BTS(i,C)$ we obtain

$$|S'| = BTD^{\kappa}(i',C) \leq BTD^{\kappa-1}(i',C) < |S|.$$

Similarly, $i \in Cons(S',C)$ and $S' \in BTS(i',C)$ yields

$$|S| = BTD^{\kappa}(i,C) \leq BTD^{\kappa-1}(i,C) < |S'|.$$

This is a contradiction. □

This implies that every *BTS*-protocol is valid in the sense of the definition given by Goldman and Mathias (1996).

Corollary 11 *Let $z \in \mathbb{N}$ and let $C \in \mathcal{C}_z$ be a concept class. Let τ be a teacher for C, λ a learner for C. If (τ,λ) is a BTS-protocol for C then (τ,λ) is successful with respect to any adversary Ad_C that fulfills $\tau(i) \subseteq S \subseteq C(i)$ for all $i \in \{1,\ldots,z\}$.*

Goldman and Mathias's definition of valid teacher/learner pairs encompasses a broad set of scenarios. It accommodates all consistent learners even those that do not make any prior assumptions about the source of information (the teacher) beyond it being noise-free. However, in many application scenarios (e.g., whenever a human interacts with a computer or in robot-robot interaction) it is reasonable to assume that (almost) all the examples selected by the teacher are helpful or particularly important for the target concept in the context of the underlying concept class. Processing a sample S selected by a teacher, a learner could exploit such an assumption by excluding not only all concepts that are inconsistent with S but also all concepts for which some examples in S would not seem particularly helpful/important. This would immediately call Goldman and Mathias's definition of validity into question.

Here we propose a more relaxed definition of what a valid teacher/learner pair is (and thus, implicitly, a new definition of collusion). It is important to notice, first of all, that in parts of the existing literature, teaching sets and teaching dimension are defined via properties of *sets* rather than properties of *representations* of sets, see Balbach (2008) and Kobayashi and Shinohara (2009). Whenever this is the case, teacher/learner pairs cannot make use of the language they use for representing instances in X or concepts in C. For example, teacher and learner cannot agree on an *order* over the instance space or over the concept class in order to encode information in samples just by the rank of their members with respect to the agreed-upon orders.

We want to make this an explicit part of the definition of collusion-free teacher/learner pairs.

Intuitively, the complexity of teaching/learning concepts in a class should not depend on certain representational features, such as any order over X or over C itself. Moreover, negating the values of all concepts on a single instance should not affect the complexity of teaching and learning either. In other words, we want protocols to be "invariant" with respect to the following equivalence relation over $\mathcal{C}$.

Definition 12 *Let $z \in \mathbb{N}$. Let $C = (C(i)(j))_{1\le i\le z, 1\le j\le n}$ and $C' = (C'(i)(j))_{1\le i\le z, 1\le j\le n}$ be two concept classes in $\mathcal{C}_z$. C and C' are called equivalent if there is a bijection $f_{row} : \{1,\ldots,z\} \to \{1,\ldots,z\}$, a bijection $f_{col} : \{1,\ldots,n\} \to \{1,\ldots,n\}$, and for every $j \in \{1,\ldots,n\}$ a bijection $\ell_j : \{+,-\} \to \{+,-\}$, such that*

$$C(i)(j) = \ell_j(C'(f_{row}(i))(f_{col}(j)) \text{ for all } i \in \{1,\ldots,z\},\ j \in \{1,\ldots,n\}.$$

In this case, $(f_{row}, f_{col}, (\ell_j)_{1\le j\le n})$ is said to witness that C and C' are equivalent.

We call a protocol collusion-free if it obeys this equivalence relation in the following sense.

Definition 13 *Let P be a protocol. P is collusion-free if, for every $z \in \mathbb{N}$ and $C, C' \in \mathcal{C}_z$, where C and C' are equivalent as witnessed by $(f_{row}, f_{col}, (\ell_j)_{1\le j\le n})$, the following two properties hold for $P(C) = (\tau,\lambda)$ and $P(C') = (\tau',\lambda')$.*

1. *If $1 \le i \le z$ and $\tau(i) = \{(j_1,l_1),\ldots,(j_r,l_r)\}$, then*

$$\tau'(f_{row}(i)) = \{(f_{col}(j_1),\ell_j(l_1)),\ldots,(f_{col}(j_r),\ell_j(l_r))\}.$$

2. *If* $\{(j_1,l_1),\ldots,(j_r,l_r)\} \in \mathcal{S}$ *and* $\lambda(\{(j_1,l_1),\ldots,(j_r,l_r)\}) = i$, *then*

$$\lambda'(\{(f_{col}(j_1),\ell_j(l_1)),\ldots,(f_{col}(j_r),\ell_j(l_r))\}) = f_{row}(i)\,.$$

It is obvious that both protocols introduced above are collusion-free.

Theorem 14 *1. Every teaching set protocol is collusion-free.*

2. Every Balbach teaching set protocol is collusion-free.

Proof. Immediate from the definitions. □

The new protocols we define below are collusion-free as well. This means that all protocols studied in this article are defined independently of the order over X and C. Concept classes can hence be considered as sets of sets rather than matrices. Consequently, Definition 1 is more general than required in the rest of this paper. We therefore ease notation as follows.

$X = \{x_1,\ldots,x_n\}$ denotes the instance space. A concept c is a subset of X and a concept class C is a subset of the power set of X. We identify every concept c with its membership function given by $c(x_i) = +$ if $x_i \in c$, and $c(x_i) = -$ if $x_i \notin c$, where $1 \leq i \leq n$. Given a sample $S = \{(y_1,l_1),\ldots,(y_r,l_r)\} \subseteq X \times \{+,-\}$, we call c consistent with S if $c(y_i) = l_i$ for all $i \in \{1,\ldots,r\}$. If C is a concept class then $Cons(S,C) = \{c \in C \mid c \text{ is consistent with } S\}$. S is called a teaching set for c with respect to C if $Cons(S,C) = \{c\}$. Then $TD(c,C) = \min\{|S| \mid Cons(S,C) = \{c\}\}$, $TD(C) = \max\{TD(c,C) \mid c \in C\}$, and $TS(c,C) = \{S \mid Cons(S,C) = \{c\} \text{ and } |S| = TD(c,C)\}$. The notations concerning the Balbach teaching model are adapted by analogy.

5. The Subset Teaching Dimension

The approach studied by Balbach (2008) does not always meet the intuitive idea of teacher and learner exploiting the knowledge that either partner behaves cooperatively. Consider for instance one more time the class C_0 containing the empty concept and all singletons over $X = \{x_1,\ldots,x_n\}$. Each concept $\{x_i\}$ has the unique minimal teaching set $\{(x_i,+)\}$ in this class, whereas the empty concept only has a teaching set of size n, namely $\{(x_1,-),\ldots,(x_n,-)\}$. The idea of elimination by size allows a learner to conjecture the empty concept as soon as two examples have been provided, due to the fact that all other concepts possess a teaching set of size one. This is why the empty concept has a *BTD* equal to 2 in this example.

However, as we have argued in Section 1, it would also make sense to devise a learner in a way to conjecture the empty concept as soon as a first example for that concept is provided—knowing that the teacher would not use a negative example for any other concept in the class. In terms of teaching sets this means to reduce the teaching sets to their minimal subsets that are not contained in minimal teaching sets for other concepts in the given concept class.

In fact, a technicality in the definition of the Balbach teaching dimension (Definition 5) disallows the Balbach teaching dimension to be 1 unless the teaching dimension itself is already 1, as the following proposition states.

Proposition 15 *Let C be a concept class. If $BTD(C) = 1$ then $TD(C) = 1$.*

Proof. Let $BTD(C) = 1$. Assume $TD(C) > 1$.

Since $TD(C) > 1$, there exists a concept $\hat{c} \in C$ such that $TD(\hat{c},C) > 1$. Since $BTD(\hat{c},C) = 1$, there exists a minimal $\kappa \geq 1$ such that $BTD^{\kappa}(\hat{c},C) = BTD(\hat{c},C) = 1$. In particular, there exists a sample S such that $|S| = 1$ and

$$\{c \in Cons(S,C) \mid BTD^{\kappa-1}(c,C) \geq 1\} = \{\hat{c}\}.$$

Since $BTD^{\kappa-1}(c,C) \geq 1$ trivially holds for all $c \in C$, we obtain $Cons(S,C) = \{\hat{c}\}$. Consequently, as $|S| = 1$, it follows that $TD(\hat{c},C) = 1$. This contradicts the choice of $\hat{c}$. Thus $TD(C) = 1$. □

So, if the Balbach model improves on the worst case teaching complexity, it does so only by improving the teaching dimension to a value of at least 2.

5.1 The Model

We formalize the idea of cooperative teaching and learning using subsets of teaching sets as follows.

Definition 16 *Let C be a concept class, $c \in C$, and S a sample. Let $STD^0(c,C) = TD(c,C)$, $STS^0(c,C) = TS(c,C)$. We define iterated sets for all $k \in \mathbb{N}$ as follows.*

- $Cons_{sub}(S,C,k) = \{c \in C \mid S \subseteq S' \text{ for some } S' \in STS^k(c,C)\}$.
- $STD^{k+1}(c,C) = \min\{|S| \mid Cons_{sub}(S,C,k) = \{c\}\}$
- $STS^{k+1}(c,C) = \{S \mid Cons_{sub}(S,C,k) = \{c\}, |S| = STD^{k+1}(c,C)\}$.

Let κ be minimal such that $STS^{\kappa+1}(c,C) = STS^{\kappa}(c,C)$ for all $c \in C$.[3]

A sample S such that $Cons_{sub}(S,C,\kappa) = \{c\}$ is called a subset teaching set for c in C. The subset teaching dimension $STD(c,C)$ of c in C is defined by $STD(c,C) = STD^{\kappa}(c,C)$ and we denote by $STS(c,C) = STS^{\kappa}(c,C)$ the set of all minimal subset teaching sets for c in C. The subset teaching dimension $STD(C)$ of C is defined by $STD(C) = \max\{STD(c,C) \mid c \in C\}$.

For illustration, consider again the concept class C_0, that is, $C_0 = \{c_i \mid 0 \leq i \leq n\}$, where $c_0 = \emptyset$ and $c_i = \{x_i\}$ for all $i \in \{1,\ldots,n\}$. Obviously, for $k \geq 1$,

$$STS^k(c_i) = \{\{(x_i,+)\}\} \text{ for all } i \in \{1,\ldots,n\}$$

and

$$STS^k(c_0) = \{\{(x_i,-)\} \mid 1 \leq i \leq n\}.$$

Hence $STD(C_0) = 1$ although $TD(C_0) = n$.

Note that the example of the concept class C_0 establishes that the subset teaching dimension can be 1 even if the teaching dimension is larger, in contrast to Proposition 15.

The definition of $STS(c,C)$ induces a protocol for teaching and learning: For a target concept c, a teacher presents the examples in a subset teaching set for c to the learner. The learner will also be able to pre-compute all subset teaching sets for all concepts and determine the target concept from the sample provided by the teacher.[4]

3. Such a κ exists because $STD^0(c,C)$ is finite and can hence be reduced only finitely often.
4. Note that we focus on sample size here, but neglect efficiency issues arising from the pre-computation of all subset teaching sets.

Protocol 17 *Let P be a protocol. P is called a subset teaching set protocol (STS-protocol for short) if the following two properties hold for every $C \subseteq \mathcal{C}$, where $P(C) = (\tau, \lambda)$.*

1. *$\tau(c) \in STS(c,C)$ for all $c \in C$,*
2. *$\lambda(S) \in \{c \mid$ there is some $S' \in STS(c,C)$ such that $S' \subseteq S\}$ for all $S \in \mathcal{S}$ that contain a set $S' \in STS(c,C)$ for some $c \in C$.*

Note that Definition 16 does not presume any special order of the concept representations or of the instances, that is, teacher and learner do not have to agree on any such order to make use of the teaching and learning protocol. That means, given a special concept class C, the computation of its subset teaching sets does not involve any special coding trick depending on C—it just follows a general rule.

By definition, every subset teaching set protocol is collusion-free. However, teacher-learner pairs following a subset teaching set protocol are not necessarily valid in the sense of Goldman and Mathias's definition. This is easily seen for the concept class C_θ of all linear threshold functions over three instances x_1, x_2, x_3. This class has four concepts, namely $c_1 = \{x_1, x_2, x_3\}$, $c_2 = \{x_2, x_3\}$, $c_3 = \{x_3\}$, and $c_4 = \{\}$. It is easy to verify that $\{(x_1, -)\}$ is a subset teaching set for c_2 and is consistent with c_3. Similarly, $\{(x_3, +)\}$ is a subset teaching set for c_3 and is consistent with c_2. Hence $\{(x_1, -), (x_3, +)\}$ is consistent with both c_2 and c_3 and contains a subset teaching set for c_2 as well as a subset teaching set for c_3. Obviously, there exists a teacher-learner pair (τ, λ) satisfying the properties of an $STS-protocol$ for this class, such that $\tau(c_2) = \{(x_1, -)\}, \tau(c_3) = \{(x_3, +)\}$, and $\lambda(\{(x_1, -), (x_3, +)\}) = c_2$. However, there is no learner λ' such that (τ, λ') is a valid teacher-learner pair for C_θ. Such a learner λ' would have to hypothesize both c_2 and c_3 on input $\{(x_1, -), (x_3, +)\}$. See Table 1 for illustration of this example.

concept	x_1	x_2	x_3	STS^0	STS^1
$\{x_1, x_2, x_3\}$	$+$	$+$	$+$	$\{(x_1, +)\}$	$\{(x_1, +)\}$
$\{x_2, x_3\}$	$-$	$+$	$+$	$\{(x_1, -), (x_2, +)\}$	$\{(x_1, -)\}, \{(x_2, +)\}$
$\{x_3\}$	$-$	$-$	$+$	$\{(x_2, -), (x_3, +)\}$	$\{(x_2, -)\}, \{(x_3, +)\}$
$\{\}$	$-$	$-$	$-$	$\{(x_3, -)\}$	$\{(x_3, -)\}$

Table 1: Iterated subset teaching sets for the class C_θ.

5.2 Comparison to the Balbach Teaching Dimension

Obviously, when using the trivial adversary, Protocol 17 based on the subset teaching dimension never requires a sample larger than a teaching set; often a smaller sample is sufficient. However, compared to the Balbach teaching dimension, the subset teaching dimension is superior in some cases and inferior in others. The latter may seem unintuitive, but is possible because Balbach's teaching sets are not restricted to be subsets of the original teaching sets.

Theorem 18 *1. For each $u \in \mathbb{N}$ there is a concept class C such that $STD(C) = 1$ and $BTD(C) = u$.*

2. For each $u \geq 3$ there is a concept class C such that $BTD(C) = 3$ and $STD(C) = u$.

Proof. Assertion 1. Let $n = 2^u + u$ be the number of instances in X. Define a concept class $C = C^u_{\text{pair}}$ as follows. For every $s = (s_1, \ldots, s_u) \in \{+,-\}^u$, C contains the concepts $c_{s,0} = \{x_i \mid 1 \leq i \leq u \text{ and } s_i = +\}$ and $c_{s,1} = c_{s,0} \cup \{x_{u+1+int(s)}\}$. Here $int(s) \in \mathbb{N}$ is defined as the sum of all values 2^{u-i} for which $s_i = +$, $1 \leq i \leq u$. We claim that $STD(C) = 1$ and $BTD(C) = u$. See Table 2 for the case $u = 2$.

Let $s = (s_1, \ldots, s_u) \in \{+,-\}^u$. Then

$$\begin{aligned} TS(c_{s,0}, C) &= \{\{(x_i, s_i) \mid 1 \leq i \leq u\} \cup \{(x_{u+1+int(s)}, -)\}\} \\ \text{and } TS(c_{s,1}, C) &= \{\{(x_{u+1+int(s)}, +)\}\} \end{aligned}$$

Since for each $c \in C$ the minimal teaching set for c with respect to C contains an example that does not occur in the minimal teaching set for any other concept $c' \in C$, one obtains $STD(C) = 1$ in just one iteration.

In contrast to that, we obtain

$$\begin{aligned} BTD^0(c_{s,0}, C) &= u + 1\,, \\ BTD^1(c_{s,0}, C) &= u\,, \\ \text{and } BTD^0(c_{s,1}, C) &= 1 \text{ for all } s \in \{+,-\}^u\,. \end{aligned}$$

Consider any $s \in \{+,-\}^u$ and any sample $S \subseteq \{(x, c_{s,0}(x)) \mid x \in X\}$ with $|S| = u - 1$. Clearly there is some $s' \in \{+,-\}^u$ with $s' \neq s$ such that $c_{s',0} \in Cons(S, C)$. So $|Cons(S, C, +)| > 1$ and in particular $Cons(S, C, +) \neq \{c_{s,0}\}$. Hence $BTD^2(c_{s,0}, C) = BTD^1(c_{s,0}, C)$, which finally implies $BTD(C) = u$.

Assertion 2. Let $n = u + 1$ be the number of instances in X. Define a concept class $C = C^u_{1/2}$ as follows. For every $i, j \in \{1, \ldots, u+1\}$, C contains the concept $\{x_i\}$ and the concept $\{x_i, x_j\}$. See Table 3 for the case $u = 4$.

Then the only minimal teaching set for a singleton $\{x_i\}$ is the sample $S^i = \{(x, -) \mid x \neq x_i\}$ with $|S^i| = u$. The only minimal teaching set for a concept $\{x_i, x_j\}$ with $i \neq j$ is the sample $S^{i,j} = \{(x_i, +), (x_j, +)\}$.

On the one hand, every subset of every minimal teaching set for a concept $c \in C$ is contained in some minimal teaching set for some concept $c' \in C$ with $c \neq c'$. Thus $STS^k(c, C) = TS(c, C)$ for all $c \in C$ and all $k \in \mathbb{N}$. Hence $STD(C) = TD(C) = u$.

On the other hand, any sample S containing $(x_i, +)$ and two negative examples $(x_\alpha, -)$ and $(x_\beta, -)$ (where i, α, and β are pairwise distinct) is in $BTS(\{x_i\}, C)$. This holds because every other concept in C that is consistent with this sample is a concept containing two instances and thus has a teaching set of size smaller than 3 ($= |S|$). Thus $BTD(C) = 3$. □

5.3 Teaching Monomials

This section provides an analysis of the *STD* for a more natural example, the monomials, showing that the very intuitive example given in the introduction is indeed what a cooperative teacher and learner in an *STS*-protocol would do. The main result is that the *STD* of the class of all monomials is 2, independent on the number m of variables, whereas its teaching dimension is exponential in m and its *BTD* is linear in m, see Balbach (2008).

Theorem 19 *Let $m \in \mathbb{N}$, $m \geq 2$ and C the class of all boolean functions over m variables that can be represented by a monomial. Then $STD(C) = 2$.*

concept	x_1	x_2	x_3	x_4	x_5	x_6	STS^0	STS^1
$\emptyset$	[−]	[−]	[−]	−	−	−	$\{(x_1,-),(x_2,-),(x_3,-)\}$	$\{(x_3,-)\}$
$\{x_3\}$	−	−	[+]	−	−	−	$\{(x_3,+)\}$	$\{(x_3,+)\}$
$\{x_2\}$	[−]	[+]	−	[−]	−	−	$\{(x_1,-),(x_2,+),(x_4,-)\}$	$\{(x_4,-)\}$
$\{x_2,x_4\}$	−	+	−	[+]	−	−	$\{(x_4,+)\}$	$\{(x_4,+)\}$
$\{x_1\}$	[+]	[−]	−	−	[−]	−	$\{(x_1,+),(x_2,-),(x_5,-)\}$	$\{(x_5,-)\}$
$\{x_1,x_5\}$	+	−	−	−	[+]	−	$\{(x_5,+)\}$	$\{(x_5,+)\}$
$\{x_1,x_2\}$	[+]	[+]	−	−	−	[−]	$\{(x_1,+),(x_2,+),(x_6,-)\}$	$\{(x_6,-)\}$
$\{x_1,x_2,x_6\}$	+	+	−	−	−	[+]	$\{(x_6,+)\}$	$\{(x_6,+)\}$

Table 2: Iterated subset teaching sets for the class C^u_{pair} with $u = 2$, where $C^u_{\text{pair}} = \{c_{--,0}, c_{--,1} \ldots, c_{++,0}, c_{++,1}\}$ with $c_{--,0} = \emptyset$, $c_{--,1} = \{x_3\}$, $c_{-+,0} = \{x_2\}$, $c_{-+,1} = \{x_2,x_4\}$, $c_{+-,0} = \{x_1\}$, $c_{+-,1} = \{x_1,x_5\}$, $c_{++,0} = \{x_1,x_2\}$, $c_{++,1} = \{x_1,x_2,x_6\}$. All labels contributing to minimal teaching sets are highlighted by square brackets.

concept	x_1	x_2	x_3	x_4	x_5	*TS* (equal to *STS*)
$\{x_1\}$	**+**	−	−	−	−	$\{(x_2,-),(x_3,-),(x_4,-),(x_5,-)\}$
$\{x_2\}$	−	**+**	−	−	−	$\{(x_1,-),(x_3,-),(x_4,-),(x_5,-)\}$
$\{x_3\}$	−	−	**+**	−	−	$\{(x_1,-),(x_2,-),(x_4,-),(x_5,-)\}$
$\{x_4\}$	−	−	−	**+**	−	$\{(x_1,-),(x_2,-),(x_3,-),(x_5,-)\}$
$\{x_5\}$	−	−	−	−	**+**	$\{(x_1,-),(x_2,-),(x_3,-),(x_4,-)\}$
$\{x_1,x_2\}$	**+**	**+**	−	−	−	$\{(x_1,+),(x_2,+)\}$
$\{x_1,x_3\}$	**+**	−	**+**	−	−	$\{(x_1,+),(x_3,+)\}$
$\{x_1,x_4\}$	**+**	−	−	**+**	−	$\{(x_1,+),(x_4,+)\}$
$\{x_1,x_5\}$	**+**	−	−	−	**+**	$\{(x_1,+),(x_5,+)\}$
$\{x_2,x_3\}$	−	**+**	**+**	−	−	$\{(x_2,+),(x_3,+)\}$
$\{x_2,x_4\}$	−	**+**	−	**+**	−	$\{(x_2,+),(x_4,+)\}$
$\{x_2,x_5\}$	−	**+**	−	−	**+**	$\{(x_2,+),(x_5,+)\}$
$\{x_3,x_4\}$	−	−	**+**	**+**	−	$\{(x_3,+),(x_4,+)\}$
$\{x_3,x_5\}$	−	−	**+**	−	**+**	$\{(x_3,+),(x_5,+)\}$
$\{x_4,x_5\}$	−	−	−	**+**	**+**	$\{(x_4,+),(x_5,+)\}$

Table 3: Iterated subset teaching sets for the class $C^u_{1/2}$ with $u = 4$.

Proof. Let $m \in \mathbb{N}$, $m \geq 2$ and $s = (s_1, \ldots, s_m)$, $s' = (s'_1, \ldots, s'_m)$ elements in $\{0,1\}^m$. Let $\Delta(s,s')$ denote the Hamming distance of s and s', that is, $\Delta(s,s') = \sum_{1 \leq i \leq m} |s(i) - s'(i)|$.

We distinguish the following types of monomials M over m variables.

Type 1: M is the empty monomial (i.e., the always true concept).

Type 2: M involves m variables, $M \not\equiv v_1 \wedge \overline{v_1}$.[5]

Type 3: M involves k variables, $1 \leq k < m$, $M \not\equiv v_1 \wedge \overline{v_1}$.

Type 4: M is contradictory, that is, $M \equiv v_1 \wedge \overline{v_1}$.

5. The symbols $\equiv$ and $\not\equiv$ denote functional equivalence and semantic non-equivalence of boolean formulae, respectively.

The following facts summarize some rather obvious properties of the corresponding minimal teaching sets for monomials (cf., for example Balbach, 2008, for more details).

Fact 1: Let M be of Type 1 and let $s,s' \in \{0,1\}^m$ such that $\Delta(s,s') = m$. Then $S = \{(s,+),(s',+)\}$ forms a minimal teaching set for M, that is, $S \in STS^0(M,C)$.

Fact 2: Let M be of Type 2 and let $s \in \{0,1\}^m$ be the unique assignment for which M evaluates positively. Moreover, let $s_1,\ldots,s_m \in \{0,1\}^m$ be the m unique assignments with $\Delta(s,s_1) = \cdots = \Delta(s,s_m) = 1$. Then $S = \{(s,+),(s_1,-),\ldots,(s_m,-)\}$ forms the one and only minimal teaching set for M, that is, $S \in STS^0(M,C)$. (Note that any two negative examples in S have Hamming distance 2.)

Fact 3: Let M be of Type 3 and let $s \in \{0,1\}^m$ be one assignment for which M evaluates positively. Moreover, let $s' \in \{0,1\}^m$ be the unique assignment with $\Delta(s,s') = m-k$ for which M evaluates positively and let $s_1,\ldots,s_k \in \{0,1\}^m$ be the k unique assignments with $\Delta(s,s_1) = \cdots = \Delta(s,s_k) = 1$ for which M evaluates negatively. Then $S = \{(s,+),(s',+),(s_1,-),\ldots,(s_k,-)\}$ forms a minimal teaching set for M, that is, $S \in STS^0(M,C)$. (Note that any two negative examples in S have Hamming distance 2.)

Fact 4: Let M be of Type 4 and let $S = \{(s,-) \mid s \in \{0,1\}^m\}$. Then S forms the one and only minimal teaching set for M, that is, $S \in STS^0(M,C)$.

After the first iteration the following facts can be observed.

Fact 1(a): Let M be of Type 1 and let $S \in STS^0(M,C)$. Then $S \in STS^1(M,C)$.

This is due to the observation that any singleton subset $S' \subseteq S$ is a subset of a teaching set in $STS^0(M',C)$ for some M' of Type 2.

Fact 2(a): Let M be of Type 2 and let $S \in STS^0(M,C)$. Then $S \in STS^1(M,C)$.

This is due to the observation that any proper subset $S' \subset S$ is a subset of a teaching set in $STS^0(M',C)$ for some M' of Type 3, if S' contains one positive example, or for some M' of Type 4, otherwise.

Fact 3(a): Let M be of Type 3 and let $s \in \{0,1\}^m$ be one assignment for which M evaluates positively. Moreover, let $s' \in \{0,1\}^m$ be the unique assignment with $\Delta(s,s') = m-k$ for which M evaluates positively and let $S = \{(s,+),(s',+)\}$. Then $S \in STS^1(M,C)$.

This is due to the following observations: (i) S is not a subset of any teaching set S' in $STS^0(M',C)$ for some M' of Type 1, since the two positive examples in S' have Hamming distance m. (ii) S is obviously not a subset of any teaching set S' in $STS^0(M',C)$ for some $M' \not\equiv M$ of Type 3. (iii) Any sufficiently small "different" subset S' of some teaching set in $STS^0(M,C)$—that is, S' contains at most two examples, but not two positive examples—is a subset of any teaching set in $STS^0(M',C)$ for some M' of Type 2, if S' contains one positive example, or for some M' of Type 4, otherwise.

Fact 4(a): Let M be of Type 4 and let $s \in \{0,1\}^m$ be any assignment. Moreover, let $s' \in \{0,1\}^m$ be any assignment with $\Delta(s,s') \neq 2$ and let $S = \{(s,-),(s',-)\}$. Then $S \in STS^1(M,C)$.

This is due to the following observations: (i) S is not a subset of any teaching set S' in $STS^0(M',C)$ for some M' of Type 2 or of Type 3, since any two negative examples in S' have Hamming distance 2. (ii) Any sufficiently small "different" subset S' of the unique teaching set in $STS^0(M,C)$—that is, S' contains at most two negative examples, but two having Hamming distance 2—is a subset of a teaching set in $STS^0(M',C)$ for some M' of Type 2.

After the second iteration the following facts can be observed.

Fact 1(b): Let M be of Type 1 and let $S \in STS^1(M,C)$. Then $S \in STS^2(M,C)$.

This is due to the observation that any singleton subset $S' \subseteq S$ is a subset of a teaching set in $STS^1(M',C)$ for some M' of Type 2.

Fact 2(b): Let M be of Type 2 and let $s \in \{0,1\}^m$ be the unique assignment for which M evaluates positively. Moreover, let $s' \in \{0,1\}^m$ be any assignments with $\Delta(s,s') = 1$ and let $S = \{(s,+),(s',-)\}$. Then $S \in STS^2(M,C)$.

This is due to the following observations: (i) S is not a subset of any teaching set S' in $STS^1(M',C)$ for some M' of Type 1, of Type 3 or of Type 4, since none of these teaching sets contains one positive and one negative example. (ii) S is obviously not a subset of any teaching set S' in $STS^1(M',C)$ for some $M' \not\equiv M$ of Type 2. (iii) Any sufficiently small "different" subset S' of a teaching set in $STS^1(M,C)$—that is, S' contains at most two examples, but not a positive and a negative example—is a subset of a teaching set in $STS^1(M',C)$ for some M' of Type 3, if S' contains one positive example, or for some $M' \not\equiv M$ of Type 2, otherwise.

Fact 3(b): Let M be of Type 3 and let $S \in STS^1(M,C)$. Then $S \in STS^2(M,C)$.

This is due to the observation that any singleton subset $S' \subseteq S$ is a subset of a teaching set in $STS^1(M',C)$ for some M' of Type 2.

Fact 4(b): Let M be of Type 4 and let $S \in STS^1(M,C)$. Then $S \in STS^2(M,C)$.

This is due to the observation that any singleton subset $S' \subseteq S$ is a subset of a teaching set in $STS^1(M',C)$ for some monomial M' of Type 2.

Note at this point that, for any monomial M of any type, we have $STD^2(M,C) = 2$.

Finally, it is easily seen that $STD^3(M,C) = STD^2(M,C) = 2$ for all $M \in C$. □

For illustration of this proof in case $m = 2$ see Table 4.

A further simple example showing that the *STD* can be constant as compared to an exponential teaching dimension, this time with an *STD* of 1, is the following.

Let $C^m_{\vee DNF}$ contain all boolean functions over $m \geq 2$ variables that can be represented by a 2-term DNF of the form $v_1 \vee M$, where M is a monomial that contains, for each i with $2 \leq i \leq m$, either the literal v_i or the literal $\overline{v_i}$. Moreover, $C^m_{\vee DNF}$ contains the boolean function that can be represented by the monomial $M' \equiv v_1$.[6]

Theorem 20 *Let $m \in \mathbb{N}$, $m \geq 2$.*

1. $TD(C^m_{\vee DNF}) = 2^{m-1}$.

2. $STD(C^m_{\vee DNF}) = 1$.

Proof. Assertion 1. Let S be a sample that is consistent with M'. Assume that for some $s \in \{0,1\}^m$, the sample S does not contain the negative example $(s,-)$. Obviously, there is a 2-term DNF $D \equiv v_1 \vee M$ such that D is consistent with $S \cup \{(s,+)\}$ and $D \not\equiv M'$. Hence S is not a teaching set for M'. Since there are exactly 2^{m-1} 2-term DNFs that represent pairwise distinct functions in C, a teaching set for M' must contain at least 2^{m-1} examples.

Assertion 2. The proof is straightforward: Obviously, $TD(D,C) = 1$ for all $D \in C$ with $D \not\equiv M'$. In particular, $STD(D,C) = 1$ for all $D \in C$ with $D \not\equiv M'$. It remains to show that $STD(M',C) = 1$. For this it suffices to see that a minimal teaching set for M' in C must contain negative examples, while no minimal teaching set for any $D \in C$ with $D \not\equiv M'$ contains any negative examples. Hence $STD^2(M',C) = 1$ and thus $STD(M',C) = 1$. □

6. Here and in the proof of Theorem 20, as in the proof of Theorem 19, the symbol $\equiv$ denotes functional equivalence of boolean formulae.

monomial	00	01	10	11	STS^0	STS^1
v_1	−	−	+	+	{(10,+),(11,+),(00,-)} {(10,+),(11,+),(01,-)}	{(10,+),(11,+)}
$\overline{v_1}$	+	+	−	−	{(00,+),(01,+),(10,-)} {(00,+),(01,+),(11,-)}	{(00,+),(01,+)}
v_2	−	+	−	+	{(01,+),(11,+),(00,-)} {(01,+),(11,+),(10,-)}	{(01,+),(11,+)}
$\overline{v_2}$	+	−	+	−	{(00,+),(10,+),(01,-)} {(00,+),(10,+),(11,-)}	{(00,+),(10,+)}
$v_1 \wedge v_2$	−	−	−	+	{(11,+),(01,-),(10,-)}	{(11,+),(01,-),(10,-)}
$v_1 \wedge \overline{v_2}$	−	−	+	−	{(10,+),(00,-),(11,-)}	{(10,+),(00,-),(11,-)}
$\overline{v_1} \wedge v_2$	−	+	−	−	{(01,+),(00,-),(11,-)}	{(01,+),(00,-),(11,-)}
$\overline{v_1} \wedge \overline{v_2}$	+	−	−	−	{(00,+),(01,-),(10,-)}	{(00,+),(01,-),(10,-)}
$v_1 \wedge \overline{v_1}$	−	−	−	−	{(00,-),(01,-),(10,-),(11,-)}	{(00,-),(01,-)} {(00,-),(10,-)} {(01,-),(11,-)} {(10,-),(11,-)}
$\mathtt{T}$	+	+	+	+	{(00,+),(11,+)} {(01,+),(10,+)}	{(00,+),(11,+)} {(01,+),(10,+)}

monomial	00	01	10	11	STS^2	STS^3
v_1	−	−	+	+	{(10,+),(11,+)}	{(10,+),(11,+)}
$\overline{v_1}$	+	+	−	−	{(00,+),(01,+)}	{(00,+),(01,+)}
v_2	−	+	−	+	{(01,+),(11,+)}	{(01,+),(11,+)}
$\overline{v_2}$	+	−	+	−	{(00,+),(10,+)}	{(00,+),(10,+)}
$v_1 \wedge v_2$	−	−	−	+	{(11,+),(01,-)} {(11,+),(10,-)}	{(11,+),(01,-)} {(11,+),(10,-)}
$v_1 \wedge \overline{v_2}$	−	−	+	−	{(10,+),(00,-)} {(10,+),(11,-)}	{(10,+),(00,-)} {(10,+),(11,-)}
$\overline{v_1} \wedge v_2$	−	+	−	−	{(01,+),(00,-)} {(01,+),(11,-)}	{(01,+),(00,-)} {(01,+),(11,-)}
$\overline{v_1} \wedge \overline{v_2}$	+	−	−	−	{(00,+),(01,-)} {(00,+),(10,-)}	{(00,+),(01,-)} {(00,+),(10,-)}
$v_1 \wedge \overline{v_1}$	−	−	−	−	{(00,-),(01,-)} {(00,-),(10,-)} {(01,-),(11,-)} {(10,-),(11,-)}	{(00,-),(01,-)} {(00,-),(10,-)} {(01,-),(11,-)} {(10,-),(11,-)}
$\mathtt{T}$	+	+	+	+	{(00,+),(11,+)} {(01,+),(10,+)}	{(00,+),(11,+)} {(01,+),(10,+)}

Table 4: Iterated subset teaching sets for the class of all monomials over $m = 2$ variables. Here $\mathtt{T}$ denotes the empty monomial. For better readability, the instances (denoting the second through fifth columns) are written in the form of bit strings representing truth assignments to the two variables.

6. Why Smaller Classes can be Harder to Teach

Interpreting the subset teaching dimension as a measure of complexity of a concept class in terms of cooperative teaching and learning, we observe a fact that is worth discussing, namely the non-monotonicity of this complexity notion, as stated by the following theorem.

Theorem 21 *There is a concept class C such that $STD(C') > STD(C)$ for some subclass $C' \subset C$.*

Proof. This is witnessed by the concept classes $C = C^u_{1/2} \cup \{\emptyset\}$ and its subclass $C' = C^u_{1/2}$ used in the proof of Theorem 18.2, for any $u > 2$ (see Table 3 and Table 5 for $u = 4$). $STD(C^u_{1/2} \cup \{\emptyset\}) = 2$ while $STD(C^u_{1/2}) = u$. □

In contrast to that, it is not hard to show that *BTD* in fact is monotonic, see Theorem 22.

Theorem 22 *If C is a concept class and $C' \subseteq C$ a subclass of C, then $BTD(C') \leq BTD(C)$.*

Proof. Fix C and $C' \subseteq C$. We will prove by induction on k that

$$BTD^k(c,C') \leq BTD^k(c,C) \text{ for all } c \in C' \tag{1}$$

for all $k \in \mathbb{N}$.

$k = 0$: Property (1) holds because of $BTD^0(c,C') = TD(c,C') \leq TD(c,C) = BTD^0(c,C)$ for all $c \in C'$.

Induction hypothesis: assume (1) holds for a fixed k.

$k \rightsquigarrow k+1$: First, observe that

$$\begin{aligned} Cons_{size}(S,C',k) &= \{c \in Cons(S,C') \mid BTD^k(c,C') \geq |S|\} \\ &\subseteq \{c \in Cons(S,C') \mid BTD^k(c,C) \geq |S|\} \text{ (ind. hyp.)} \\ &\subseteq \{c \in Cons(S,C) \mid BTD^k(c,C) \geq |S|\} \\ &= Cons_{size}(S,C,k) \end{aligned}$$

Second, for all $c \in C'$ we obtain

$$\begin{aligned} BTD^{k+1}(c,C') &= \min\{|S| \mid Cons_{size}(S,C',k) = \{c\}\} \\ &\leq \min\{|S| \mid Cons_{size}(S,C,k) = \{c\}\} \\ &\leq BTD^{k+1}(c,C) \end{aligned}$$

This completes the proof. □

6.1 Nonmonotonicity After Elimination of Redundant Instances

Note that the nonmonotonicity of the subset teaching dimension holds with a fixed number of instances n. In fact, if n was not considered fixed then every concept class C' would have a superset C (via addition of instances) of lower subset teaching dimension. However, the same even holds for the teaching dimension itself which we yet consider monotonic since it is monotonic given fixed n. So whenever we speak of monotonicity we assume a fixed instance space X.

Of course such an instance space X might contain *redundant* instances the removal of which would not affect the subset teaching dimension and would retain a non-redundant subset of the

concept	x_1	x_2	x_3	x_4	x_5	STS^0
$\emptyset$	−	−	−	−	−	$\{(x_1,-),(x_2,-),(x_3,-),(x_4,-),(x_5,-)\}$
$\{x_1\}$	+	−	−	−	−	$\{(x_1,+),(x_2,-),(x_3,-),(x_4,-),(x_5,-)\}$
$\{x_2\}$	−	+	−	−	−	$\{(x_1,-),(x_2,+),(x_3,-),(x_4,-),(x_5,-)\}$
$\{x_3\}$	−	−	+	−	−	$\{(x_1,-),(x_2,-),(x_3,+),(x_4,-),(x_5,-)\}$
$\{x_4\}$	−	−	−	+	−	$\{(x_1,-),(x_2,-),(x_3,-),(x_4,+),(x_5,-)\}$
$\{x_5\}$	−	−	−	−	+	$\{(x_1,-),(x_2,-),(x_3,-),(x_4,-),(x_5,+)\}$
$\{x_1,x_2\}$	+	+	−	−	−	$\{(x_1,+),(x_2,+)\}$
$\{x_1,x_3\}$	+	−	+	−	−	$\{(x_1,+),(x_3,+)\}$
$\{x_1,x_4\}$	+	−	−	+	−	$\{(x_1,+),(x_4,+)\}$
$\{x_1,x_5\}$	+	−	−	−	+	$\{(x_1,+),(x_5,+)\}$
$\{x_2,x_3\}$	−	+	+	−	−	$\{(x_2,+),(x_3,+)\}$
$\{x_2,x_4\}$	−	+	−	+	−	$\{(x_2,+),(x_4,+)\}$
$\{x_2,x_5\}$	−	+	−	−	+	$\{(x_2,+),(x_5,+)\}$
$\{x_3,x_4\}$	−	−	+	+	−	$\{(x_3,+),(x_4,+)\}$
$\{x_3,x_5\}$	−	−	+	−	+	$\{(x_3,+),(x_5,+)\}$
$\{x_4,x_5\}$	−	−	−	+	+	$\{(x_4,+),(x_5,+)\}$
concept	x_1	x_2	x_3	x_4	x_5	STS^1
$\emptyset$	−	−	−	−	−	$\{(x_1,-),(x_2,-),(x_3,-),(x_4,-),(x_5,-)\}$
$\{x_1\}$	+	−	−	−	−	$\{(x_1,+),(x_2,-)\},\ldots,\{(x_1,+),(x_5,-)\}$
$\{x_2\}$	−	+	−	−	−	$\{(x_1,-),(x_2,+)\},\ldots,\{(x_2,+),(x_5,-)\}$
$\{x_3\}$	−	−	+	−	−	$\{(x_1,-),(x_3,+)\},\ldots,\{(x_3,+),(x_5,-)\}$
$\{x_4\}$	−	−	−	+	−	$\{(x_1,-),(x_4,+)\},\ldots,\{(x_4,+),(x_5,-)\}$
$\{x_5\}$	−	−	−	−	+	$\{(x_1,-),(x_5,+)\},\ldots,\{(x_4,-),(x_5,+)\}$
$\{x_1,x_2\}$	+	+	−	−	−	$\{(x_1,+),(x_2,+)\}$
$\{x_1,x_3\}$	+	−	+	−	−	$\{(x_1,+),(x_3,+)\}$
$\{x_1,x_4\}$	+	−	−	+	−	$\{(x_1,+),(x_4,+)\}$
$\{x_1,x_5\}$	+	−	−	−	+	$\{(x_1,+),(x_5,+)\}$
$\{x_2,x_3\}$	−	+	+	−	−	$\{(x_2,+),(x_3,+)\}$
$\{x_2,x_4\}$	−	+	−	+	−	$\{(x_2,+),(x_4,+)\}$
$\{x_2,x_5\}$	−	+	−	−	+	$\{(x_2,+),(x_5,+)\}$
$\{x_3,x_4\}$	−	−	+	+	−	$\{(x_3,+),(x_4,+)\}$
$\{x_3,x_5\}$	−	−	+	−	+	$\{(x_3,+),(x_5,+)\}$
$\{x_4,x_5\}$	−	−	−	+	+	$\{(x_4,+),(x_5,+)\}$

Table 5: Iterated subset teaching sets for the class $C^u_{1/2} \cup \{\emptyset\}$ with $u = 4$; two iterations. In the third iteration, the sample for the empty concept (first row) will be reduced to all its subsets of size two, thus witnessing an *STD* of 2.

set of all subset teaching sets. In the following subsection, where we discuss a possible intuition behind the nonmonotonicity of the *STD*, redundancy conditions on instances will actually play an important role and show the usefulness of the following technical discussion. However, it is not straightforward to impose a suitable redundancy condition characterizing when an instance can be removed.

We derive such a condition starting with a redundancy condition for the original variant of teaching sets. For that purpose we introduce the notion C^{-x} for the concept class resulting from C after removing the instance x from the instance space X. Here C is any concept class over X and

$x \in X$ is any instance. For example, if $X = \{x_1, x_2, x_3\}$ and $C = \{\{x_1\}, \{x_1, x_2\}, \{x_2, x_3\}\}$ then

$$C^{-x_3} = \{\{x_1\}, \{x_1, x_2\}, \{x_2\}\}$$

considered over the instance space $\{x_1, x_2\}$.

To ease notation, we use a single name c for both a concept $c \in C$ and its corresponding concept in the class C^{-x} for any $x \in X$. It will always be clear from the context which concept is referred to.

Lemma 23 *Let C be a concept class over X and $x \in X$. Suppose for all $c \in C$ and for all $S \in TS(c, C)$*

$$(x, c(x)) \in S \Rightarrow \exists y \neq x \, [(S \setminus \{(x, c(x))\}) \cup \{(y, c(y))\} \in TS(c, C)] .$$

Then the following two assertions are true.

1. $|C^{-x}| = |C|$.
2. *For all $c \in C$ and for all samples S*

$$S \in TS(c, C^{-x}) \iff [S \in TS(c, C) \wedge (x, c(x)) \notin S] .$$

Proof. Assertion 1. Assume $|C^{-x}| < |C|$.

Then there must be two distinct concepts $c, c' \in C$ such that c and c' disagree only in x, that is, $c(y) = c'(y)$ for all $y \in X \setminus \{x\}$ and $c(x) \neq c'(x)$. Consequently, $(x, c(x))$ must be contained in some $S \in TS(c, C)$. By the premise of the lemma, this implies that there is some $y \in X \setminus \{x\}$ such that $(S \setminus \{(x, c(x))\}) \cup \{(y, c(y))\} \in TS(c, C)$. Hence $(S \setminus \{(x, c(x))\}) \cup \{(y, c(y))\}$ is a teaching set for c in C that does not contain $(x, c(x))$. However, $(S \setminus \{(x, c(x))\}) \cup \{(y, c(y))\}$ is consistent with c', which is a contradiction. Therefore $|C^{-x}| = |C|$.

Assertion 2. Let $c \in C$ be an arbitrary concept and let S be any sample over X.

First assume $S \in TS(c, C)$ and $(x, c(x)) \notin S$. By Assertion 1, $|C^{-x}| = |C|$ and therefore $TD(c, C^{-x}) \geq TD(c, C)$. Thus we immediately obtain $S \in TS(c, C^{-x})$.

Second assume $S \in TS(c, C^{-x})$. By definition, we have $(x, c(x)) \notin S$. Hence it remains to prove that $S \in TS(c, C)$. If $S \notin TS(c, C)$ then there exists some $T \in TS(c, C)$ such that $|T| < |S|$, because otherwise $|C^{-x}|$ would be smaller than $|C|$. We distinguish two cases.

Case 1. $(x, c(x)) \notin T$.

Then $T \in TS(c, C^{-x})$ in contradiction to the facts $S \in TS(c, C^{-x})$ and $|S| \neq |T|$.

Case 2. $(x, c(x)) \in T$.

Then by the premise of the lemma there exists a $y \neq x$ such that

$$A \stackrel{\text{def}}{=} (S \setminus \{(x, c(x))\}) \cup \{(y, c(y))\} \in TS(c, C) .$$

Since $(x, c(x)) \notin A$ we have $A \in TS(c, C^{-x})$ and $|A| = |T| \neq |S|$. This again contradicts $S \in TS(c, C^{-x})$.

Since both cases reveal a contradiction, we obtain $S \in TS(c, C)$. □

For illustration see Table 6. In this example the instances x_4 and x_5 meet the redundancy condition. After eliminating x_5, the instance x_4 still meets the condition and can be removed as well. The new representation of the concept class then involves only the instances x_1, x_2, x_3.

Lemma 23 provides a condition on an instance x. If that instance is eliminated from the instance space then the resulting concept class C^{-x} not only has the same teaching dimension as C but, even more, for each of its concepts c the teaching sets are exactly those that are teaching sets for c in C

concept in C	x_1	x_2	x_3	x_4	x_5	TS
$\emptyset$	$-$	$-$	$-$	$-$	$-$	$\{(x_1,-),(x_3,-)\},\{(x_1,-),(x_4,-)\},\{(x_1,-),(x_5,-)\}$
$\{x_1\}$	$+$	$-$	$-$	$-$	$-$	$\{(x_1,+),(x_2,-)\},\{(x_1,+),(x_5,-)\}$
$\{x_3,x_4,x_5\}$	$-$	$-$	$+$	$+$	$+$	$\{(x_2,-),(x_3,+)\},\{(x_2,-),(x_4,+)\},\{(x_2,-),(x_5,+)\}$
$\{x_2,x_3,x_4,x_5\}$	$-$	$+$	$+$	$+$	$+$	$\{(x_1,-),(x_2,+)\},\{(x_2,+),(x_4,+)\}$
$\{x_1,x_2,x_5\}$	$+$	$+$	$-$	$-$	$+$	$\{(x_2,+),(x_3,-)\},\{(x_3,-),(x_5,+)\}$
$\{x_1,x_2,x_3,x_5\}$	$+$	$+$	$+$	$-$	$+$	$\{(x_1,+),(x_3,+)\},\{(x_3,+),(x_4,-)\}$

concept in $(C^{-x_5})^{-x_4}$	x_1	x_2	x_3	TS
$\emptyset$	$-$	$-$	$-$	$\{(x_1,-),(x_3,-)\}$
$\{x_1\}$	$+$	$-$	$-$	$\{(x_1,+),(x_2,-)\}$
$\{x_3\}$	$-$	$-$	$+$	$\{(x_2,-),(x_3,+)\}$
$\{x_2,x_3\}$	$-$	$+$	$+$	$\{(x_1,-),(x_2,+)\}$
$\{x_1,x_2\}$	$+$	$+$	$-$	$\{(x_2,+),(x_3,-)\}$
$\{x_1,x_2,x_3\}$	$+$	$+$	$+$	$\{(x_1,+),(x_3,+)\}$

Table 6: Teaching sets for a class C before and after elimination of two redundant instances.

and do not contain an example involving the eliminated instance x. Note that even though several instances might meet that condition at the same time, only one at a time may be removed. For the remaining instances it has to be checked whether the condition still holds after elimination of the first redundant instance.

In the example in Table 6, x_4 and x_5 are exactly those instances that could be eliminated without reducing the size of the concept class, that is,

$$|C| = |C^{-x_4}| = |C^{-x_5}| = |(C^{-x_4})^{-x_5}| = |(C^{-x_5})^{-x_4}|.$$

However, if we were to simply eliminate all instances x as long as $|C| = |C^{-x}|$, then the consequence of Lemma 23 would not necessarily be fulfilled any longer. For example, consider the concept class C in Table 7. Here $|C| = |C^{-x_1}|$, but removing x_1 from the instance space would increase the teaching dimension of c_1, namely $TD(c_1,C) = 1 < 2 = TD(c_1,C^{-x_1})$.

So one legitimate redundancy condition for instances—considering the preservation of teaching sets—is the one given in the premise of Lemma 23. This condition can be extended to a redundancy condition with respect to subset teaching sets.

Theorem 24 *Let C be a concept class over X and $x \in X$. Suppose for all $k \in \mathbb{N}$, for all $c \in C$, and for all $S \in STS^k(c,C)$*

$$(x,c(x)) \in S \Rightarrow \exists y \neq x\, [(S \setminus \{(x,c(x))\}) \cup \{(y,c(y))\} \in STS^k(c,C)],$$

Then the following two assertions are true.

1. $|C^{-x}| = |C|$.
2. *For all $k \in \mathbb{N}$, for all $c \in C$, and for all samples S*

$$S \in STS^k(c,C^{-x}) \iff [S \in STS^k(c,C) \wedge (x,c(x)) \notin S].$$

concept in C	x_1	x_2	x_3	TS
$c_1 = \{x_1, x_2, x_3\}$	$+$	$+$	$+$	$\{(x_1,+)\}$
$c_2 = \{x_2\}$	$-$	$+$	$-$	$\{(x_2,+),(x_3,-)\}, \{(x_2,+),(x_1,-)\}$
$c_3 = \{x_3\}$	$-$	$-$	$+$	$\{(x_3,+),(x_2,-)\}, \{(x_3,+),(x_1,-)\}$
$c_4 = \emptyset$	$-$	$-$	$-$	$\{(x_2,-),(x_3,-)\}$

concept in C^{-x_1}	x_2	x_3	TS
$c_1 = \{x_2, x_3\}$	$+$	$+$	$\{(x_2,+),(x_3,+)\}$
$c_2 = \{x_2\}$	$+$	$-$	$\{(x_2,+),(x_3,-)\}$
$c_3 = \{x_3\}$	$-$	$+$	$\{(x_3,+),(x_2,-)\}$
$c_4 = \emptyset$	$-$	$-$	$\{(x_2,-),(x_3,-)\}$

Table 7: Teaching sets for a class C before and after elimination of the instance x_1 not satisfying the premises of Lemma 23, despite fulfilling the property $|C| = |C^{-x_1}|$.

Proof. Assertion 1. This follows immediately by applying Lemma 23.1 for $k = 0$.

Assertion 2. We prove the second assertion by induction on k.

For $k = 0$ the assertion follows immediately from Lemma 23.2. So assume that the assertion is proven for some k (induction hypothesis). It remains to show that it then also holds for $k+1$.

For that purpose note that

$$\forall c \in C\ \forall A \in STS^k(c,C)\ \exists B \in STS^k(c,C^{-x})\ [|A| = |B| \ \wedge\ A \setminus \{(x,c(x))\} \subseteq B] \quad (*)$$

by combination of the induction hypothesis with the premise of the theorem.

Choose an arbitrary $c \in C$.

First assume $S \in STS^{k+1}(c,C)$ and $(x,c(x)) \notin S$. By the definition of subset teaching sets, there is an $S' \in STS^k(c,C)$ such that

$$S \subseteq S'. \tag{2}$$

Using $(*)$ we can assume without loss of generality that

$$S' \in STS^k(c,C^{-x}). \tag{3}$$

Moreover, again by the definition of subset teaching sets, one obtains $S \not\subseteq S''$ for every $S'' \in STS^k(c',C)$ with $c' \neq c$. The induction hypothesis then implies

$$S \not\subseteq S'' \text{ for every } S'' \in STS^k(c',C^{-x}) \text{ with } c' \neq c. \tag{4}$$

Due to (2), (3), (4) we get either $S \in STS^{k+1}(c,C^{-x})$ or $|S| > STD^{k+1}(c,C^{-x})$. In the latter case there would be a set $T \in STS^{k+1}(c,C^{-x})$ such that $|T| < |S|$. T is a subset of some set in $STS^k(c,C^{-x})$ and thus also of some set in $STS^k(c,C)$ by induction hypothesis. If T was contained in some $T' \in STS^k(c',C)$ for some $c' \neq c$ then we could again assume without loss of generality, using $(*)$ and $(x,c(x)) \notin T$, that T is contained in some set in $STS^k(c',C^{-x})$—in contradiction to $T \in STS^{k+1}(c,C^{-x})$. Therefore $T \in STS^{k+1}(c,C)$ and so $|T| = |S|$—a contradiction. This implies $S \in STS^{k+1}(c,C^{-x})$.

Second assume that $S \in STS^{k+1}(c,C^{-x})$. Obviously, $(x,c(x)) \notin S$, so that it remains to show $S \in STS^{k+1}(c,C)$.

Because of $S \in STS^{k+1}(c,C^{-x})$ there exists some set $S' \in STS^{k}(c,C^{-x})$ such that

$$S \subseteq S' . \tag{5}$$

The induction hypothesis implies

$$S' \in STS^{k}(c,C) . \tag{6}$$

Further, by the definition of subset teaching sets, one obtains $S \not\subseteq S''$ for every $S'' \in STS^{k}(c',C^{-x})$ with $c' \neq c$. If there was a set $S'' \in STS^{k}(c',C)$ such that $c' \neq c$ and $S \subseteq S''$ then $(*)$ would imply that without loss of generality $S'' \in STS^{k}(c',C^{-x})$. So we have

$$S \not\subseteq S'' \text{ for every } S'' \in STS^{k}(c',C) \text{ with } c' \neq c . \tag{7}$$

Combining (5), (6), (7) we get either $S \in STS^{k+1}(c,C)$ or $|S| > STD^{k+1}(c,C)$. In the latter case there would be a set $T \in STS^{k+1}(c,C)$ such that $|T| < |S|$. T is a subset of some set $T' \in STS^{k}(c,C)$. We can assume without loss of generality, using $(*)$, that $T' \in STS^{k}(c,C^{-x})$. If T was contained in some set in $STS^{k}(c',C^{-x})$ for some $c' \neq c$ then by induction hypothesis T would be contained in some set in $STS^{k}(c',C)$ for some $c' \neq c$. This is a contradiction to $T \in STS^{k+1}(c,C)$. So $T \in STS^{k+1}(c,C^{-x})$ and hence $|T| = |S|$—a contradiction. Thus $S \in STS^{k+1}(c,C)$. □

The example in Table 7 illustrates that eliminating instances x satisfying $|C^{-x}| = |C|$, without any additional constraints, can actually change the subset teaching dimension of a class. In the given example, the subset teaching dimension of C is 1, while the subset teaching dimension of C^{-x_1} is 2. The stronger condition on the instance x in the premise of Theorem 24 guarantees that eliminating x does not change the subset teaching dimension.

6.2 Nonmonotonicity and the Role of Nearest Neighbours

From a general point of view, it is not obvious how to explain why a teaching dimension resulting from a cooperative model should be nonmonotonic.

First of all, this is a counter-intuitive observation when considering *STD* as a notion of complexity—intuitively any subclass of C should be at most as complex for teaching and learning as C.

However, there is in fact an intuitive explanation for the nonmonotonicity of the complexity in cooperative teaching and learning: when teaching $c \in C$, instead of providing examples that eliminate all concepts in $C \setminus \{c\}$ (as is the idea underlying minimal teaching sets) cooperative teachers would rather pick only those examples that distinguish c from its "most similar" concepts in C. Similarity here is measured by the number of instances on which two concepts agree (i.e., dissimilarity is given by the Hamming distance between the concepts, where a concept c is represented as a bit vector $(c(x_1),\ldots,c(x_n))$). This is reflected in the subset teaching sets in all illustrative examples considered above.

Considering a class $C = C^{u}_{\text{pair}}$ (see the proof of Theorem 18.1), one observes that a subset teaching set for a concept $c_{s,0}$ contains only the negative example $(x_{u+1+int(s)},-)$ distinguishing it from $c_{s,1}$ (its nearest neighbor in terms of Hamming distance). A learner will recognize this example as the one that separates only that one pair $(c_{s,0},c_{s,1})$ of nearest neighbors. In contrast to that, if we consider only the subclass $C' = \{c_{s,0} \mid s \in \{0,1\}^u\}$, the nearest neighbors of each $c_{s,0}$ are different

ones, and every single example separating one nearest neighbor pair also separates other nearest neighbor pairs. Thus no single example can be recognized by the learner as a separating example for one unique pair of concepts.

This intuitive idea of subset teaching sets being used for distinguishing a concept from its nearest neighbors has to be treated with care though. The reason is that the concept class may contain "redundant" instances, that is, instances that could be removed from the instance space according to Theorem 24.

Such redundant instances might on the other hand affect Hamming distances and nearest neighbor relations. Only after their elimination does the notion of nearest neighbors in terms of Hamming distance become useful. Consider for instance Table 6. In the concept class C over 5 instances the only nearest neighbor of $\emptyset$ is $\{x_1\}$ and an example distinguishing $\emptyset$ from $\{x_1\}$ would be $(x_1,-)$. Moreover, no other concept is distinguished from its nearest neighbors by the instance x_1. According to the intuition explained here, this would suggest $\{(x_1,-)\}$ being a subset teaching set for $\emptyset$ although the subset teaching sets here equal the teaching sets and are all of cardinality 2.

After instance elimination of x_4,x_5 there is only one subset teaching set for $\emptyset$, namely $\{(x_1,-),(x_3,-)\}$. This is still of cardinality 2 but note that now $\emptyset$ has two nearest neighbors, namely $\{x_1\}$ and $\{x_3\}$. The two examples in the subset teaching set are those that distinguish $\emptyset$ from its nearest neighbors. Note that either one of these two examples is not unique as an example used for distinguishing a concept from its nearest neighbors: $(x_1,-)$ would be used by $\{x_2,x_3\}$ for distinguishing itself from its nearest neighbor $\{x_1,x_2,x_3\}$, and $(x_3,-)$ would be used by $\{x_1,x_2\}$ for distinguishing itself from its nearest neighbor $\{x_1,x_2,x_3\}$. So the subset teaching set for $\emptyset$ has to contain both examples.

This illustrates why a subclass of a class C can have a higher complexity than C if crucial nearest neighbors of some concepts are missing in the subclass.

To summarize,

- nonmonotonicity has an intuitive reason and is not an indication of an ill-defined version of the teaching dimension,
- nonmonotonicity would in fact be a consequence of implementing the idea that the existence of specific concepts (e.g., nearest neighbours) associated with a target concept is beneficial for teaching and learning.

So, the STD captures certain intuitions about teaching and learning that monotonic dimensions *cannot* capture; at the same time monotonicity might in other respects itself be an intuitive property of teaching and learning which then the STD cannot capture.

In particular there are two underlying intuitive properties that seem to not be satisfiable by a single variant of the teaching dimension.

7. The Recursive Teaching Dimension

On the one hand, we have the teaching framework based on the subset teaching dimension which results in a nonmonotonic dimension, and on the other hand we have a monotonic dimension in the *BTD* framework, which unfortunately does not always meet our idea of a cooperative teaching and learning protocol. That raises the question whether nonmonotonicity is necessary to achieve certain positive results. In fact, the nonmonotonicity concerning the class C^u_{pair} is not counter-intuitive, but

would a dimension that is monotonic also result in a worse sample complexity than the *STD* in general, such as, for example, for the monomials?

In other words, is there a teaching/learning framework

- resulting in a monotonic variant of a teaching dimension and
- achieving low teaching complexity results similar to the subset teaching dimension?

At this point of course it is difficult to define what "similar to the subset teaching dimension" means. However, we would like to have a constant dimension for the class of all monomials, as well as, for example, a teaching set of size 1 for the empty concept in our often used concept class C_0.

We will now via several steps introduce a monotonic variant of the teaching dimension and show that for most of the examples studied above, it is as low as the subset teaching dimension. General comparisons will be made in Section 8, in particular in order to show that this new framework is uniformly at least as efficient as the *BTD* framework, while sometimes being less efficient than the *STD* framework. This reflects to a certain extent that monotonicity constraints might affect sample efficiency.

7.1 The Model

We will first define our new variant of teaching dimension and show its monotonicity.

The nonmonotonicity of *STD* is caused by considering every STS^k-set for every concept when computing an STS^{k+1}-set for a single concept. Hence the idea in the following approach is to impose a canonical order on the concept class, in terms of the "teaching complexity" of the concepts. This is what the teaching dimension does as well, but our design principle is a recursive one. After selecting a set of concepts each of which is "easy to teach" because of possessing a small minimal teaching set, we eliminate these concepts from our concept class and consider only the remaining concepts. Again we determine those with the lowest teaching dimension, now however measured with respect to the class of remaining concepts, and so on. The resulting notion of dimension is therefore called the *recursive teaching dimension*.

Definition 25 *Let C be a concept class. The teaching hierarchy for C is the sequence $H = ((C_1,d_1), \ldots,(C_h,d_h))$ that fulfills, for all $j \in \{1,\ldots,h\}$,*

$$C_j = \{c \in \overline{C}_j \mid d_j = TD(c,\overline{C}_j) \leq TD(c',\overline{C}_j) \text{ for all } c' \in \overline{C}_j\},$$

where $\overline{C}_1 = C$ and $\overline{C}_{i+1} = C \setminus (C_1 \cup \ldots \cup C_i)$ for all $i \in \{1,\ldots,h-1\}$.

For any $j \in \{1,\ldots,h\}$ and any $c \in C_j$, a sample $S \in TS(c,\overline{C}_j)$ is called a recursive teaching set for c in C. The recursive teaching dimension $RTD(c,C)$ of c in C is then defined as $RTD(c,C) = d_j$ and we denote by $RTS(c,C) = TS(c,\overline{C}_j)$ the set of all recursive teaching sets for c in C.

The recursive teaching dimension $RTD(C)$ of C is defined by

$$RTD(C) = \max\{d_j \mid 1 \leq j \leq h\}.$$

The desired monotonicity property, see Proposition 26, follows immediately from the definition.

Proposition 26 *If C is a concept class and $C' \subseteq C$ is a subclass of C, then $RTD(C') \leq RTD(C)$.*

The definition of teaching hierarchy induces a protocol for teaching and learning: for a target concept c, a teacher uses the teaching hierarchy $H = ((C_1,d_1),\ldots,(C_h,d_h))$ for C to determine the unique index j with $c \in C_j$. The teacher then presents the examples in a teaching set from $TS(c,\overline{C}_j)$, that is, a recursive teaching set for c in C, to the learner. The learner will use the teaching hierarchy to determine the target concept from the sample provided by the teacher.

Protocol 27 *Let P be a protocol. P is called a recursive teaching set protocol (RTS-protocol for short) if the following two properties hold for every $C \subseteq \mathcal{C}$, where $P(C) = (\tau,\lambda)$.*

1. *$\tau(c) \in RTS(c,C)$ for all $c \in C$,*
2. *$\lambda(S) \in \{c \mid$ there is some $S' \in RTS(c,C)$ such that $S' \subseteq S\}$ for all $S \in \mathcal{S}$ that contain a set $S' \in RTS(c,C)$ for some $c \in C$.*

Note again that Definition 25 does not presume any special order of the concept representations or of the instances, that is, teacher and learner do not have to agree on any such order to make use of the teaching and learning protocol. The partial order resulting from the teaching hierarchy is still well-defined.

The following definition of canonical teaching plans yields an alternative definition of the recursive teaching dimension.

Definition 28 *Let C be a concept class, $|C| = z$. A teaching plan for C is a sequence $p = ((c_1,S_1),\ldots,(c_z,S_z)) \in (C \times 2^{X\times\{0,1\}})^z$ such that*

1. *$C = \{c_1,\ldots,c_z\}$.*
2. *$S_j \in TS(c_j,\{c_j,\ldots,c_z\})$ for $1 \le j \le z$.*

The order of p is given by $ord(p) = \max\{|S_j| \mid 1 \le j \le z\}$.

p is called a canonical teaching plan for C, if for any $i,j \in \{1,\ldots,z\}$:

$$i < j \Rightarrow TD(c_i,\{c_i,\ldots,c_z\}) \le TD(c_j,\{c_i,\ldots,c_z\}).$$

Note that every concept class has a canonical teaching plan. It turns out that a canonical teaching plan has the lowest possible order over all teaching plans; this order coincides with the recursive teaching dimension, see Theorem 29.

Theorem 29 *Let C be a concept class and p^* a canonical teaching plan for C. Then $ord(p^*) = \min\{ord(p) \mid p$ is a teaching plan for $C\} = RTD(C)$.*

Proof. Let C and p^* as in the theorem be given, $p^* = ((c_1,S_1),\ldots,(c_z,S_z))$. $ord(p^*) = RTD(C)$ follows by definition. It needs to be shown that

$$ord(p^*) = \min\{ord(p) \mid p \text{ is a teaching plan for } C\}.$$

Let $p' = ((c'_1,S'_1),\ldots,(c'_z,S'_z))$ be any teaching plan for C. It remains to prove that $ord(p^*) \le ord(p')$.

For that purpose choose the minimal $j \in \{1,\ldots,z\}$ such that $|S_j| = ord(p^*)$. By definition of a teaching plan, $TD(c_j,\{c_j,\ldots,c_z\}) = ord(p^*)$. Let $i \in \{1,\ldots,z\}$ be minimal such that $c'_i \in$

$\{c_j,\ldots,c_z\}$. Let $k \in \{1,\ldots,z\}$ fulfill $c_k = c'_i$. By definition of a canonical teaching plan, $TD(c_k,\{c_j,\ldots,c_z\}) \geq TD(c_j,\{c_j,\ldots,c_z\}) = ord(p^*)$. This obviously yields $ord(p') \geq TD(c'_i,\{c'_i,\ldots,c'_z\}) \geq TD(c_k,\{c_j,\ldots,c_z\}) \geq ord(p^*)$. □

To summarize briefly, the recursive teaching dimension is a monotonic complexity notion which in fact has got some of the properties we desired; for example, it is easily verified that $RTD(C_0) = 1$ (by any teaching plan in which the empty concept occurs last) and that the *RTD* of the class of all monomials equals 2 (see below). Thus the *RTD* overcomes some of the weaknesses of *BTD*, while at the same time preserving monotonicity.

Interestingly, unlike for subset teaching set protocols, the teacher-learner pairs based on recursive teaching set protocols are valid in the sense of Goldman and Mathias's definition (Goldman and Mathias, 1996). This is an immediate consequence of the following theorem.

Theorem 30 *Let C be any concept class and $c \in C$. Let S be any sample. If S is consistent with c and there is some $T \in RTS(c,C)$ such that $T \subseteq S$ then there is no concept $c' \in Cons(S,C)$ with $c' \neq c$ and $T' \subseteq S$ for some $T' \in RTS(c',C)$.*

Proof. Let C, c, S, and T as in the theorem be given. Let $H = ((C_1,d_1),\ldots,(C_h,d_h))$ be the teaching hierarchy for C and let $i \in \{1,\ldots,h\}$ be such that $c \in C_i$.

Assume there was a concept $c' \in Cons(S,C)$ with $c' \neq c$ and $T' \subseteq S$ for some $T' \in RTS(c',C)$. Let $j \in \{1,\ldots,h\}$ be such that $c' \in C_j$.

Since c is consistent with S and S contains the recursive teaching set T' for c', c is also consistent with T'. As $c \in C_i$ is consistent with a recursive teaching set for $c' \in C_j$, we obtain $j > i$.

Similarly, since c' is consistent with S and S contains the recursive teaching set T for c, we obtain $i > j$.

This is clearly a contradiction. Hence there is no concept $c' \in Cons(S,C)$ with $c' \neq c$ and $T' \subseteq S$ for some $T' \in RTS(c',C)$. □

7.2 Comparison to the Balbach Teaching Dimension

Unlike the subset teaching dimension, the recursive teaching dimension lower-bounds the Balbach dimension. To prove this, we first observe that the smallest teaching dimension of all concepts in a given concept class is a lower bound on the Balbach dimension. This is stated formally in the following lemma.

Lemma 31 *Let C be a concept class. Then $BTD(C) \geq \min\{TD(c,C) \mid c \in C\}$.*

Proof. Let $u = \min\{TD(c,C) \mid c \in C\}$. To show that $BTD(C) \geq u$, we will prove by induction on k that $u \leq BTD^k(c,C)$ for all $k \in \mathbb{N}$ for all $c \in C$.

$k = 0$: $BTD^0(c,C) = TD(c,C) \geq u$ for all $c \in C$.

Induction hypothesis: assume $u \leq BTD^k(c,C)$ for all $c \in C$ holds for a fixed k.

$k \rightsquigarrow k+1$: Suppose by way of contradiction that there is a concept $c^* \in C$ such that $u > BTD^{k+1}(c^*,C)$. In particular, there exists a sample S^* such that $|S^*| < u$ and $Cons_{size}(S^*,C,k) = \{c^*\}$.

By induction hypothesis, the set $Cons_{size}(S^*,C,k)$ defined by $\{c \in Cons(S^*,C) \mid BTD^k(c,C) \geq |S^*|\}$ is equal to $Cons(S^*,C)$. Note that $TD(c,C) \geq u$ for all $c \in C$ implies either $|Cons(S^*,C)| \geq 2$ or $Cons(S^*,C) = \emptyset$. We obtain a contradiction to $Cons_{size}(S^*,C,k) = \{c^*\}$.

This completes the proof. □

This lemma helps to prove that the recursive teaching dimension cannot exceed the Balbach dimension.

Theorem 32 *1. If C is a concept class then $RTD(C) \leq BTD(C)$.*

2. There is a concept class C such that $RTD(C) < BTD(C)$.

Proof. Assertion 1. To prove this assertion, let C be a concept class such that $RTD(C) = u$. By Theorem 29 there is a canonical teaching plan $p = ((c_1, S_1), \ldots, (c_z, S_z))$ for C such that $ord(p) = u$. Fix $j \leq \mathbb{N}$ minimal such that $|S_j| = u$ and define $C' = \{c_j \ldots, c_z\}$. Obviously, $RTD(C') = u$. Moreover, using Theorem 22, $BTD(C') \leq BTD(C)$. Thus it suffices to prove $u \leq BTD(C')$. This follows from Lemma 31, since $u = \min\{TD(c, C') \mid c \in C'\}$.

This completes the proof of Assertion 1.

Assertion 2. The second assertion is witnessed by the concept class C_0 containing the empty concept and all singletons. Obviously, $RTD(C_0) = 1$ and $BTD(C_0) = 2$. □

7.3 Teaching Monomials

In this subsection, we pick up the two examples from Section 5.3 again, this time in order to determine the recursive teaching dimension of the corresponding classes of concepts represented by boolean functions. As in the case of the subset teaching dimension, see Theorem 19, we obtain that the recursive teaching dimension of the class of all monomials over m ($m \geq 2$) variables is 2, independent of m.

Theorem 33 *Let $m \in \mathbb{N}$, $m \geq 2$, and C the class of all boolean functions over m variables that can be represented by a monomial. Then $RTD(C) = 2$.*

Proof. Fix m and C. For all $i \in \{0, \ldots, m\}$ let C^i be the subclass of all $c \in C$ that can be represented by a non-contradictory monomial M that involves i variables. There is exactly one concept in C not belonging to any subclass C^i of C, namely the concept c^* representable by a contradictory monomial.

The proof is based on the following observation.

Observation. For any $i \in \{0, \ldots, m\}$ and any $c \in C^i$: $TD(c, C' \cup \{c^*\}) \leq 2$, where $C' = \bigcup_{i \leq j \leq m} C^j$.

Now it is easily seen that $ord(p) \leq 2$ for every teaching plan $p = ((c_1, S_1), \ldots, (c_z, S_z))$ for C that meets the following requirements:

(a) $c_1 \in C^0$ and $c_z = c^*$.

(b) For any $k, k' \in \{1, \ldots, z-1\}$: If $k < k'$, then $c_k \in C^i$ and $c_{k'} \in C^j$ for some $i, j \in \{0, \ldots, m\}$ with $i \leq j$.

Therefore $RTD(C) \leq 2$.

Since obviously $TD(c, C) \geq 2$ for all $c \in C$, we obtain $RTD(C) = 2$.

(For illustration of the case $m = 2$ see Table 8.) □

For the sake of completeness, note that $RTD(C^m_{\vee DNF}) = 1$ where $C^m_{\vee DNF}$ is the class of boolean functions over m variables as defined in Section 5.3.

Theorem 34 *$RTD(C^m_{\vee DNF}) = 1$ for all $m \in \mathbb{N}$, $m \geq 2$.*

monomial	subclass	00	01	10	11	*RTS*
$\mathtt{T}$	C^0	+	+	+	+	{(00,+),(11,+)}
v_1	C^1	−	−	+	+	{(10,+),(11,+)}
$\overline{v_1}$	C^1	+	+	−	−	{(00,+),(01,+)}
v_2	C^1	−	+	−	+	{(01,+),(11,+)}
$\overline{v_2}$	C^1	+	−	+	−	{(00,+),(10,+)}
$v_1 \wedge v_2$	C^2	−	−	−	+	{(11,+)}
$v_1 \wedge \overline{v_2}$	C^2	−	−	+	−	{(10,+)}
$\overline{v_1} \wedge v_2$	C^2	−	+	−	−	{(01,+)}
$\overline{v_1} \wedge \overline{v_2}$	C^2	+	−	−	−	{(00,+)}
$v_1 \wedge \overline{v_1}$		−	−	−	−	{}

Table 8: Recursive teaching sets in the teaching hierarchy (corresponding to teaching plans of order 2) for the class of all monomials over $m = 2$ variables. $\mathtt{T}$ denotes the empty monomial. For better readability, the instances (denoting the third through sixth columns) are written in the form of bit strings representing truth assignments to the two variables.

Proof. This follows straightforwardly from the fact that $TD(c, C^m_{\vee DNF}) = 1$ for every concept c corresponding to a 2-term DNF of form $v_1 \vee M$.

(For illustration see Table 4.) □

8. Subset Teaching Dimension Versus Recursive Teaching Dimension

Comparing the *STD* to the *RTD* turns out to be a bit more complex. We can show that the recursive teaching dimension can be arbitrarily larger than the subset teaching dimension; it can even be larger than the maximal *STD* computed over all subsets of the concept class.

Theorem 35 *1. For each $u \in \mathbb{N}$ there is a concept class C such that $STD(C) = 1$ and $RTD(C) = u$.*

2. There is a concept class C such that $\max\{STD(C') \mid C' \subseteq C\} < RTD(C)$.

Proof. Assertion 1. This is witnessed by the classes C^u_{pair} defined in the proof of Theorem 18.1.

Assertion 2. To verify Assertion 2, consider the concept class $C = \{c_1, \ldots, c_6\}$ given by $c_1 = \emptyset$, $c_2 = \{x_1\}$, $c_3 = \{x_1, x_2\}$, $c_4 = \{x_2, x_3\}$, $c_5 = \{x_2, x_4\}$, $c_6 = \{x_2, x_3, x_4\}$. It is not hard to verify that $TD(c, C) = 2$ for all $c \in C$ and thus $ord(p) = 2$ for every teaching plan p for C. Therefore $RTD(C) = 2$. Moreover $STD(C') = 1$ for all $C' \subseteq C$ (the computation of $STD(C)$ is shown in Table 9; further details are omitted). □

Similarly, we can prove that the subset teaching dimension can be arbitrarily larger than the recursive teaching dimension.

Theorem 36 *For each $u \geq 2$ there is a concept class C such that $RTD(C) = 2$ and $STD(C) = u$.*

Proof. This is witnessed by the class $C = C^u_{1/2}$ used in the proof of Theorem 18.2. □

concept	x_1	x_2	x_3	x_4	STS^0	STS^1	STS^2
$\emptyset$	$-$	$-$	$-$	$-$	$\{(x_1,-),(x_2,-)\}$	$\{(x_1,-)\}$	$\{(x_1,-)\}$
$\{x_1\}$	$+$	$-$	$-$	$-$	$\{(x_1,+),(x_2,-)\}$	$\{(x_1,+),(x_2,-)\}$	$\{(x_1,+)\}$ $\{(x_2,-)\}$
$\{x_1,x_2\}$	$+$	$+$	$-$	$-$	$\{(x_1,+),(x_2,+)\}$	$\{(x_2,+)\}$	$\{(x_2,+)\}$
$\{x_2,x_3\}$	$-$	$+$	$+$	$-$	$\{(x_3,+),(x_4,-)\}$	$\{(x_4,-)\}$	$\{(x_4,-)\}$
$\{x_2,x_4\}$	$-$	$+$	$-$	$+$	$\{(x_3,-),(x_4,+)\}$	$\{(x_3,-)\}$	$\{(x_3,-)\}$
$\{x_2,x_3,x_4\}$	$-$	$+$	$+$	$+$	$\{(x_3,+),(x_4,+)\}$	$\{(x_3,+),(x_4,+)\}$	$\{(x_3,+)\}$ $\{(x_4,+)\}$

Table 9: Iterated subset teaching sets for the class $C = \{c_1,\ldots,c_6\}$ given by $c_1 = \emptyset$, $c_2 = \{x_1\}$, $c_3 = \{x_1,x_2\}$, $c_4 = \{x_2,x_3\}$, $c_5 = \{x_2,x_4\}$, $c_6 = \{x_2,x_3,x_4\}$.

Due to the incomparability of *STD* and *RTD* it seems worth analyzing conditions under which they become comparable. To this end, we define a property that is sufficient for a concept class to have a recursive teaching dimension equal to its subset teaching dimension.

Definition 37 (Subset Teaching Property) *Let C be a concept class. C fulfills the Subset Teaching Property if for every teaching plan $p = ((c_1,S_1),\ldots,(c_z,S_z))$ for C with $ord(p) = RTD(C)$ and every j with $ord(p) = |S_j|$ and $STD(c_j,C) \geq ord(p)$ there exists a teaching plan*

$$p' = ((c_1,S'_1),\ldots,(c_z,S'_z))$$

for C and a sample $S \in STS(c_j,C)$ such that $S'_j \subseteq S$ and $|S'_i| = |S_i|$ for all i.

Theorem 38 *Let C be a concept class with the Subset Teaching Property. Then $STD(C) = RTD(C)$.*

Proof. $STD(C) \geq RTD(C)$ follows trivially: if $p = ((c_1,S_1),\ldots,(c_z,S_z))$ is a teaching plan for C with $ord(p) = RTD(C)$ and if j fulfils $RTD(C) = ord(p) = |S_j|$, then there is some S'_j with $|S'_j| = |S_j|$ and some $S \in STS(c_j,C)$ such that $S'_j \subseteq S$; hence $STD(C) \geq |S| \geq |S'_j| = |S_j| = ord(p)$.

In order to show that $STD(C) \leq RTD(C)$, we prove property (P_j) for all $j \in \{1,\ldots,|C|\}$. The proof is done by induction on j.

> (P_j):
> If $p = ((c_1,S_1),\ldots,(c_z,S_z))$ is a teaching plan for C with $ord(p) = RTD(C)$ then $STD(c_j,C) \leq ord(p)$.

For $j = 1$ this is obvious, because

$$STD(c_1,C) \leq TD(c_1,C) \leq ord(p)$$

for any teaching plan $p = ((c_1,S_1),\ldots,(c_z,S_z))$ for C.

The induction hypothesis is that (P_i) holds for all $i \leq j$, j fixed.

To prove (P_{j+1}), let $p = ((c_1,S_1),\ldots,(c_z,S_z))$ be any teaching plan for C with $ord(p) = RTD(C)$. Consider the $(j+1)^{st}$ concept c_{j+1} in p.

Case 1. $|S_{j+1}| < ord(p)$.

In this case we swap c_j and c_{j+1} and get a new teaching plan

$$\begin{aligned} p' \quad = \quad & ((c_1,S_1),\ldots,(c_{j-1},S_{j-1}), \\ & (c_{j+1},T),(c_j,T'),\ldots,(c_z,S_z)) \end{aligned}$$

for C.

Note that $|T'| \leq |S_j|$. Moreover, $|T| \leq |S_{j+1}|+1 \leq ord(p) = RTD(C)$. Hence $ord(p') = RTD(C)$.

Now c_{j+1} is in j^{th} position in the teaching plan p' whose order is equal to $RTD(C)$. By induction hypothesis we get $STD(c_{j+1},C) \leq ord(p') = ord(p)$.

Case 2. $|S_{j+1}| = ord(p)$.

In this case we use the Subset Teaching Property. Assume that $STD(c_{j+1},C) > ord(p)$. By the Subset Teaching Property, there is a teaching plan

$$p' = ((c_1,S'_1),\ldots,(c_z,S'_z))$$

for C and a sample $S \in STS(c_{j+1},C)$ such that $S'_{j+1} \subseteq S$ and $|S'_i| = |S_i|$ for all i.

First, note that S'_{j+1} is not contained in any subset teaching set for any $c \in C \setminus \{c_{j+1}\}$: The concepts $c_{j+2},\ldots,c_z$ are not consistent with the sample S'_{j+1}, because $S'_{j+1} \in TS(c_{j+1},\{c_{j+1},\ldots,c_z\})$. The concepts $c_1,\ldots,c_j$ have, by induction hypothesis, a subset teaching dimension upper-bounded by $ord(p) = |S_{j+1}| = |S'_{j+1}|$. If S'_{j+1} was contained in a subset teaching set for some concept among $c_1,\ldots,c_j$, this would imply that S'_{j+1} equaled some subset teaching set for some concept among $c_1,\ldots,c_j$, and thus S'_{j+1} could not be contained in the subset teaching set S for c_{j+1}.

Second, since S'_{j+1} is contained in the subset teaching set S for c_{j+1} and not contained in any subset teaching set for any $c \in C \setminus \{c_{j+1}\}$, S'_{j+1} equals S and is itself a subset teaching set for c_{j+1}. Consequently,

$$|S'_{j+1}| = STD(c_{j+1},C) > ord(p) = ord(p') \geq |S'_{j+1}|\,.$$

This is obviously a contradiction in itself.

Hence $STD(c_{j+1},C) \leq ord(p)$.

This concludes the induction step. □

For example, the class of all linear threshold functions, cf. C_θ in Table 1, has the subset teaching property. Every teaching plan $p = ((c_1,S_1),\ldots,(c_z,S_z))$ for C_θ with $ord(p) = RTD(C_\theta) = 1$ starts either with the concept $c_1 = X$ or with the concept $c_1 = \emptyset$. In either case, S_1 actually is a subset teaching set for c_1. A recursive argument for the subsequent concepts in the teaching plan shows that C_θ has the subset teaching property.

A similar argument proves that the class of all monomials over m instances, for any $m \geq 2$, has the subset teaching property.

9. Conclusions

We introduced two new models of teaching and learning of finite concept classes, based on the idea that learners can learn from a smaller number of labeled examples if they assume that the teacher chooses helpful examples. These models contrast with the classic teaching dimension model in which no more assumptions on the learner are made than it being consistent with the information presented. As a consequence, the information-theoretic complexity resulting from our new models is in general much lower than the teaching dimension.

Such results have to be interpreted with care since one constraint in modeling teaching is that coding tricks have to be avoided. However, one of our two models, the one based on *recursive teaching sets*, complies with Goldman and Mathias's original definition of valid teaching without coding tricks, see Goldman and Mathias (1996). The model based on *subset teaching sets* does not comply with the same definition of valid teaching. As we argued though, Goldman and Mathias's definition may be too restrictive when modeling cooperation in teaching and learning. Intuitively, their definition requires a learner to hypothesize a concept c as soon as any teaching set for c is contained in the given sample. This artificially precludes the possibility of a model in which a learner assumes that all the examples selected by the teacher are representative. Hence we introduced a less restrictive definition of coding trick. Each of the protocols presented in this paper can be regarded as meta-algorithms generating teacher/learner pairs that do not involve coding tricks.

The subset teaching protocol questions not only classic definitions of coding trick but also the intuitive idea that information-theoretic complexity measures should be monotonic with respect to the inclusion of concept classes. We discussed why non-monotonicity in this context may be a natural phenomenon in cooperative teaching and learning.

For many "natural" concept classes, the subset teaching dimension and the recursive teaching dimension turn out to be equal, but in general the two measures are not comparable. This immediately implies that neither one of the two corresponding teaching models is optimal among protocols that yield collusion-free teacher-learner pairs. One could easily design a protocol that, for every concept class C, would follow the subset teaching protocol if $STD(C) \leq RTD(C)$ and would follow the recursive teaching protocol if $RTD(C) < STD(C)$. Such a protocol would comply with our definition of collusion-freeness and would strictly dominate both the subset teaching protocol and the recursive teaching protocol. In this paper, we did not address the question of optimality of teaching protocols; we focused on intuitiveness of the protocols and the resulting teaching sets.

Acknowledgments

Many thanks are due to David Kirkpatrick, Shai Ben-David, Will Evans, and Wael Emara for pointing out flaws in an earlier version of this paper and for inspiring and helpful discussions and comments.

We furthermore gratefully acknowledge the support of Laura Zilles and Michael Geilke who developed and provided a software tool for computing subset teaching sets and recursive teaching sets.

This work was partly funded by the Alberta Ingenuity Fund (AIF) and by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

D. Angluin and M. Kriķis. Teachers, learners and black boxes. In *Proc. of the 10th Annual Conference on Computational Learning Theory (COLT'97)*, pages 285–297. ACM, New York, 1997.

D. Angluin and M. Kriķis. Learning from different teachers. *Mach. Learn.*, 51:137–163, 2003.

M. Anthony, G. Brightwell, D.A. Cohen, and J. Shawe-Taylor. On exact specification by examples. In *Proc. of 5th Annual Workshop on Computational Learning Theory (COLT'92)*, pages 311–318.

ACM, New York, 1992.

F. Balbach. Measuring teachability using variants of the teaching dimension. *Theoret. Comput. Sci.*, 397(1-3):94–113, 2008.

F. Balbach and T. Zeugmann. Recent developments in algorithmic teaching. In *Proc. 3rd Intl. Conf. on Language and Automata Theory and Applications*, pages 1–18, 2009.

S. Ben-David and N. Eiron. Self-directed learning and its relation to the VC-dimension and to teacher-directed learning. *Machine Learning*, 33(1):87–104, 1998.

T. Doliwa, H.U. Simon, and S. Zilles. Recursive teaching dimension, learning complexity, and maximum classes. In *Proc. of the 21st International Conference on Algorithmic Learning Theory (ALT'10)*, volume 6331 of *Lecture Notes in Artificial Intelligence*, pages 209–223. Springer, 2010.

R. Freivalds, E.B. Kinber, and R. Wiehagen. On the power of inductive inference from good examples. *Theoret. Comput. Sci.*, 110(1):131–144, 1993.

S. Goldman and M. Kearns. On the complexity of teaching. In *Proc. of the 4th Annual Workshop on Computational Learning Theory, COLT'91*, pages 303–314. Morgan Kaufmann, 1991.

S.A. Goldman and M.J. Kearns. On the complexity of teaching. *J. Comput. Syst. Sci.*, 50(1):20–31, 1995.

S.A. Goldman and H.D. Mathias. Teaching a smarter learner. *J. Comput. Syst. Sci.*, 52(2):255–267, 1996.

S. Hanneke. Teaching dimension and the complexity of active learning. In *Proc. of the 20th Annual Conference on Learning Theory (COLT 2007)*, pages 66–81. LNCS 4539, Springer, Berlin, 2007.

T. Hegedűs. Generalized teaching dimensions and the query complexity of learning. In *Proc. of the 8th Annual Conference on Computational Learning Theory (COLT'95)*, pages 108–117. ACM, New York, 1995.

J. Jackson and A. Tomkins. A computational model of teaching. In *Proc. of 5th Annual Workshop on Computational Learning Theory (COLT'92)*, pages 319–326. ACM, New York, 1992.

H. Kobayashi and A. Shinohara. Complexity of teaching by a restricted number of examples. In *Proc. of the 22nd Annual Conference on Learning Theory, COLT'09*, pages 293–302, 2009.

S. Lange, J. Nessel, and R. Wiehagen. Learning recursive languages from good examples. *Ann. Math. Artif. Intell.*, 23(1-2):27–52, 1998.

H.D. Mathias. A model of interactive teaching. *J. Comput. Syst. Sci.*, 54(3):487–501, 1997.

M. Ott and F. Stephan. Avoiding coding tricks by hyperrobust learning. *Theoret. Comput. Sci.*, 284 (1):161–180, 2002.

R.L. Rivest and Y.L. Yin. Being taught can be faster than asking questions. In *Proc. of the 8th Annual Conference on Computational Learning Theory (COLT'95)*, pages 144–151. ACM, New York, 1995.

R.A. Servedio. On the limits of efficient teachability. *Inf. Process. Lett.*, 79(6):267–272, 2001.

A. Shinohara and S. Miyano. Teachability in computational learning. *New Generation Comput.*, 8 (4):337–348, 1991.

L.G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

S. Zilles, S. Lange, R. Holte, and M. Zinkevich. Teaching dimensions based on cooperative learning. In *Proc. of the 21st Annual Conference on Learning Theory (COLT'08)*, pages 135–146. Omnipress, 2008.

Journal of Machine Learning Research 12 (2011) 385-416 Submitted 2/10; Revised 4/10; Published 2/11

Operator Norm Convergence of Spectral Clustering on Level Sets

Bruno Pelletier BRUNO.PELLETIER@UNIV-RENNES2.FR
Department of Mathematics
IRMAR – UMR CNRS 6625
Université Rennes II
Place du Recteur Henri Le Moal, CS 24307
35043 Rennes Cedex, France

Pierre Pudlo PIERRE.PUDLO@UNIV-MONTP2.FR
I3M: Institut de Mathématiques et de Modélisation de Montpellier – UMR CNRS 5149
Université Montpellier II, CC 051
Place Eugène Bataillon
34095 Montpellier Cedex 5, France

Editor: Ulrike von Luxburg

Abstract

Following Hartigan (1975), a cluster is defined as a connected component of the t-level set of the underlying density, that is, the set of points for which the density is greater than t. A clustering algorithm which combines a density estimate with spectral clustering techniques is proposed. Our algorithm is composed of two steps. First, a nonparametric density estimate is used to extract the data points for which the estimated density takes a value greater than t. Next, the extracted points are clustered based on the eigenvectors of a graph Laplacian matrix. Under mild assumptions, we prove the almost sure convergence in operator norm of the empirical graph Laplacian operator associated with the algorithm. Furthermore, we give the typical behavior of the representation of the data set into the feature space, which establishes the strong consistency of our proposed algorithm.

Keywords: spectral clustering, graph, unsupervised classification, level sets, connected components

1. Introduction

The aim of data clustering, or unsupervised classification, is to partition a data set into several homogeneous groups relatively separated one from each other with respect to a certain distance or notion of similarity. There exists an extensive literature on clustering methods, and we refer the reader to Anderberg (1973), Hartigan (1975) and McLachlan and Peel (2000), Chapter 10 in Duda et al. (2000), and Chapter 14 in Hastie et al. (2001) for general materials on the subject. In particular, popular clustering algorithms, such as Gaussian mixture models or k-means, have proved useful in a number of applications, yet they suffer from some internal and computational limitations. Indeed, the parametric assumption at the core of mixture models may be too stringent, while the standard k-means algorithm fails at identifying complex shaped, possibly non-convex, clusters.

The class of *spectral clustering* algorithms is presently emerging as a promising alternative, showing improved performance over classical clustering algorithms on several benchmark problems

©2011 Bruno Pelletier and Pierre Pudlo.

and applications (see, e.g., Ng et al., 2002; von Luxburg, 2007). An overview of spectral clustering algorithms may be found in von Luxburg (2007), and connections with kernel methods are exposed in Fillipone et al. (2008). The spectral clustering algorithm amounts at embedding the data into a feature space by using the eigenvectors of the similarity matrix in such a way that the clusters may be separated using simple rules, for example, a separation by hyperplanes. The core component of the spectral clustering algorithm is therefore the similarity matrix, or certain normalizations of it, generally called graph Laplacian matrices; see Chung (1997). Graph Laplacian matrices may be viewed as discrete versions of bounded operators between functional spaces. The study of these operators has started out recently with the works by Belkin et al. (2004); Belkin and Niyogi (2005), Coifman and Lafon (2006), Nadler et al. (2006), Koltchinskii (1998), Giné and Koltchinskii (2006), Hein et al. (2007) and Rosasco et al. (2010), among others.

In the context of spectral clustering, the convergence of the empirical graph Laplacian operators has been established in von Luxburg et al. (2008). Their results imply the existence of an asymptotic partition of the support of the underlying distribution of the data as the number of samples goes to infinity. However this theoretical partition results from a partition in a feature space, that is, it is the pre-image of a partition of the feature space by the embedding mapping. Therefore interpreting the asymptotic partition with respect to the underlying distribution of the data remains largely an open and challenging question. Similar interpretability questions also arise in the related context of kernel methods where the data is embedded in a feature space. For instance, while it is well-known that the popular k-means clustering algorithm leads to an optimal quantizer of the underlying distribution (MacQueen, 1967; Pollard, 1981; Linder, 2002), "kernelized" versions of the k-means algorithm allow to separate groups using nonlinear decision rules but are more difficult to interpret.

The rich variety of clustering algorithms raises the question of the definition of a cluster, and as pointed out in von Luxburg and Ben-David (2005) and in García-Escudero et al. (2008), there exists many such definitions. Among these, perhaps the most intuitive and precise definition of a cluster is the one introduced by Hartigan (1975). Suppose that the data is drawn from a probability density f on $\mathbb{R}^d$ and let t be a positive number in the range of f. Then a cluster in the sense of Hartigan (1975) is a connected component of the upper t-level set

$$\mathcal{L}(t) = \left\{x \in \mathbb{R}^d : f(x) \geq t\right\}.$$

This definition has several advantages. First, it is geometrically simple. Second, it offers the possibility of filtering out possibly meaningless clusters by keeping only the observations falling in a region of high density. This proves useful, for instance, in the situation where the data exhibits a cluster structure but is contaminated by a uniform background noise.

In this context, the level t should be considered as a resolution level for the data analysis. For instance, when the threshold t is taken equal to 0, the groups in the sense of Hartigan (1975) are the connected components of the support of the underlying distribution, while as t increases, the clusters concentrate in a neighborhood of the principal modes of the density f. Several clustering algorithms deriving from Hartigan's definition have been introduced building. In Cuevas et al. (2000, 2001), and in the related work by Azzalini and Torelli (2007), clustering is performed by estimating the connected components of $\mathcal{L}(t)$. Hartigan's definition is also used in Biau et al. (2007) to define an estimate of the number of clusters based on an approximation of the level set by a neighborhood graph.

In the present paper, we adopt the definition of a cluster of Hartigan (1975), and we propose and study a spectral clustering algorithm on estimated level sets. The algorithm is composed of

two operations. Using the sample $X_1, \ldots, X_n$ of vectors of $\mathbb{R}^d$, we first construct a nonparametric density estimate $\hat{f}_n$ of the unknown density f. Next, given a positive number t, this estimate is used to extract those observations for which the estimated density exceeds the fixed threshold, that is, the observations for which $\hat{f}_n(X_i) \geq t$. In the second step of the algorithm, we perform a spectral clustering of the extracted points. The remaining data points are then left unlabeled.

Our proposal is to study the asymptotic properties of this algorithm. In the whole study, the density estimate $\hat{f}_n$ is arbitrary but supposed consistent, and the threshold t is fixed in advance. For the spectral clustering part of the algorithm, we consider the setting where the kernel function, or similarity function, between any two pairs of observations is non negative and with a compact support of diameter $2h$, for some fixed positive real number h. Our contribution contain two sets of results.

In the first set of results, we establish the almost-sure convergence in operator norm of the empirical graph Laplacian on the estimated level set. In von Luxburg et al. (2008), the authors prove the collectively compact convergence of the empirical operator, acting on the Banach space of continuous functions on some compact set. Finite sample bounds in Hilbert-Schmidt norms on Sobolev spaces are obtained in the paper by Rosasco et al. (2010). In our result, the empirical operator is acting on a Banach subspace of the Holder space $C^{0,1}$ of Lipschitz functions, which we equip with a Sobolev norm. This operator norm convergence is more amenable than the slightly weaker notion of convergence established in von Luxburg et al. (2008), and holds for any value of the scale parameter h, but the functional space that we consider is smaller. As in the related works referenced above, the operator norm convergence is derived using results from the theory of empirical processes to prove that certain classes of functions satisfy a uniform law of large numbers. We also rely on geometrical auxiliary results to obtain the convergence of the preprocessing step of the algorithm. Under mild regularity assumptions, we use the fact that the topology of the level set $\mathcal{L}(t)$ changes only when the threshold t passes a critical value of f. This allow us to define random graph Laplacian operators acting on a fixed space of functions, with large probability.

In the second set of results, we study the convergence of the spectrum of the empirical operator, as a corollary of the operator norm convergence. Depending on the values of the scale parameter h, we characterize the properties of the asymptotic partition induced by the clustering algorithm. First, we assume that h is lower than the minimal distance between any two connected components of the t-level set. Under this condition, we prove that the embedded data points concentrate on several isolated points, each of whose corresponds to a connected component of the level set, that is, observations belonging to the same connected component of the level set are mapped onto the same point in the feature space. As a consequence, in the asymptotic regime, any reasonable clustering algorithm applied on the transformed data partitions the observations according to the connected components of the level set. In this sense, recalling Hartigan's (1975) definition of a cluster, these results imply that the proposed algorithm is strongly consistent and that, asymptotically, observations of $\mathcal{L}(t)$ are assigned to the same cluster if and only if they fall in the same connected component of $\mathcal{L}(t)$. These properties follow from the ones of the continuous (i.e., population version) operator, which we establish by using arguments related to a Markov chain on a general state space. The underlying fact is that the normalized empirical graph Laplacian defines a random walk on the extracted observations, which converges to a random walk on $\mathcal{L}(t)$. Then, asymptotically, when the scale parameter is lower than the minimal distance between the connected components of $\mathcal{L}(t)$, this random walk cannot jump from one connected component to one another. Next, by exploiting the continuity of the operators in the scale parameter h, we obtain similar consistency results when h

is slightly greater than the minimal distance between two connected components of $\mathcal{L}(t)$. In this case, the embedded data points concentrates in several non-overlapping cubes, each of whose corresponds to a connected component of $\mathcal{L}(t)$. This result holds whenever h is smaller than a certain critical value h_{max}, which depends only on the underlying density f.

Finally, let us note that our consistency results hold for any value of the threshold t different from a critical value of the density f, which assume to be twice continuously differentiable. Under the stronger assumption that f is p times continuously differentiable, with $p \geq d$, Sard's lemma imply that the set of critical values of f has Lebesgue measure 0, so that the consistency would hold for almost all t. The special limit case $t = 0$ corresponds to performing a clustering on all the observations, and our results imply the convergence of the clustering to the partition of the support of the density into its connected components, for a suitable choice of the scale parameter. The proofs could be simplified in this setting, though, since no pre-processing step would be needed. Let us mention that this asymptotic partition could also be derived from the results in von Luxburg et al. (2008). At last, we obtain consistency in the sense of Hartigan's definition when the correct number of clusters is requested, which corresponds to the number of connected components of $\mathcal{L}(t)$, and when the similarity function has a compact support . Hence several questions remain largely open which are discussed further in the paper.

The paper is organized as follows. In Section 2, we start by introducing the necessary notations and assumptions. Then we define the spectral clustering algorithm on estimated level sets, and we follow by introducing the functional operators associated with the algorithm. In Section 3, we study the almost-sure convergence in operator norm of the random operators, starting with the unnormalized empirical graph Laplacian operator. The main convergence result of the normalized operator is stated in Theorem 4. Section 4 contains the second set of results on the consistency of the clustering algorithm. We start by studying the properties of the limit operator in the case where the scale parameter h is lower than the minimal distance between two connected components of $\mathcal{L}(t)$. The convergence of the spectrum, and the consistency of the algorithm, is then stated in Theorem 7. This result is extended in Theorem 10 to allow for larger values of h. We conclude this section with a discussion on possible extensions and open problems. The proofs of these theorems rely on several auxiliary technical lemmas which are collected in Sections 5. Finally, to make the paper self contained, materials and some facts from the geometry of level sets, functional analysis, and Markov chains are exposed in Appendices A, B, and C, respectively, at the end of the paper.

2. Spectral Clustering Algorithm

In this section we give a description of the spectral clustering algorithm on level sets that is suitable for our theoretical analysis.

2.1 Mathematical Setting and Assumptions

Let $\{X_i\}_{i\geq 1}$ be a sequence of i.i.d. random vectors in $\mathbb{R}^d$, with common probability measure μ. Suppose that μ admits a density f with respect to the Lebesgue measure on $\mathbb{R}^d$. The *t-level set* of f is denoted by $\mathcal{L}(t)$, that is,

$$\mathcal{L}(t) := \{x \in \mathbb{R}^d : f(x) \geq t\},$$

for all positive level t, and given $a \leq b$, $\mathcal{L}_a^b$ denotes the set $\{x \in \mathbb{R}^d : a \leq f(x) \leq b\}$. The differentiation operator with respect to x is denoted by D. We assume that f satisfies the following conditions.

Assumption 1. *(i)* f is of class $\mathcal{C}^2$ on $\mathbb{R}^d$; *(ii)* $\|D_x f\| > 0$ on the set $\{x \in \mathbb{R}^d : f(x) = t\}$; *(iii)* f, Df, and $D^2 f$ are uniformly bounded on $\mathbb{R}^d$.

Note that under Assumption 1, $\mathcal{L}(t)$ is compact whenever t belongs to the interior of the range of f. Moreover, $\mathcal{L}(t)$ has a finite number ℓ of connected components $\mathcal{C}_j$, $j = 1, \ldots, \ell$. For ease of notation, the dependence of $\mathcal{C}_j$ on t is omitted. The minimal distance between the connected components of $\mathcal{L}(t)$ is denoted by d_{min}, that is,

$$d_{min} := \inf_{i \neq j} \operatorname{dist}\big(\mathcal{C}_i, \mathcal{C}_j\big). \tag{1}$$

Let $\widehat{f}_n$ be a consistent density estimate of f based on the random sample $X_1, \ldots, X_n$. The t-level set of $\widehat{f}_n$ is denoted by $\mathcal{L}_n(t)$, that is,

$$\mathcal{L}_n(t) := \big\{x \in \mathbb{R}^d : \widehat{f}_n(x) \geq t\big\}.$$

Let $J(n)$ be the set of integers defined by

$$J(n) := \big\{j \in \{1, \ldots, n\} : \widehat{f}_n(X_j) \geq t\big\}.$$

The cardinality of $J(n)$ is denoted by $j(n)$.

Let $k : \mathbb{R}^d \to \mathbb{R}_+$ be a fixed function. The unit ball of $\mathbb{R}^d$ centered at the origin is denoted by B, and the ball centered at $x \in \mathbb{R}^d$ and of radius r is denoted by $x + rB$. We assume throughout that the function k satisfies the following set of conditions.

Assumption 2. *(i)* k is of class $\mathcal{C}^2$ on $\mathbb{R}^d$; *(ii)* the support of k is B; *(iii)* k is uniformly bounded from below on $B/2$ by some positive number; and *(iv)* $k(-x) = k(x)$ for all $x \in \mathbb{R}^d$.

Let h be a positive number. We denote by $k_h : \mathbb{R}^d \to \mathbb{R}_+$ the map defined by $k_h(u) := k(u/h)$.

2.2 Algorithm

The first ingredient of our algorithm is the *similarity matrix* $\mathbf{K}_{n,h}$ whose elements are given by

$$\mathbf{K}_{n,h}(i, j) := k_h(X_j - X_i),$$

and where the integers i and j range over the random set $J(n)$. Hence $\mathbf{K}_{n,h}$ is a random matrix indexed by $J(n) \times J(n)$, whose values depend on the function k_h, and on the observations X_j lying in the estimated level set $\mathcal{L}_n(t)$. Next, we introduce the diagonal *normalization matrix* $\mathbf{D}_{n,h}$ whose diagonal entries are given by

$$\mathbf{D}_{n,h}(i, i) := \sum_{j \in J(n)} \mathbf{K}_{n,h}(i, j), \quad i \in J(n).$$

Note that the diagonal elements of $\mathbf{D}_{n,h}$ are positive since $\mathbf{K}_{n,h}(i, i) > 0$.

The spectral clustering algorithm is based on the matrix $\mathbf{Q}_{n,h}$ defined by

$$\mathbf{Q}_{n,h} := \mathbf{D}_{n,h}^{-1}\mathbf{K}_{n,h}.$$

Observe that $\mathbf{Q}_{n,h}$ is a random Markovian transition matrix. Note also that the (random) eigenvalues of $\mathbf{Q}_{n,h}$ are real numbers and that $\mathbf{Q}_{n,h}$ is diagonalizable. Indeed the matrix $\mathbf{Q}_{n,h}$ is conjugate to the symmetric matrix $\mathbf{S}_{n,h} := \mathbf{D}_{n,h}^{-1/2}\mathbf{K}_{n,h}\mathbf{D}_{n,h}^{-1/2}$ since we may write

$$\mathbf{Q}_{n,h} = \mathbf{D}_{n,h}^{-1/2}\mathbf{S}_{n,h}\mathbf{D}_{n,h}^{1/2}.$$

Moreover, the inequality $\|\mathbf{Q}_{n,h}\|_\infty \leq 1$ implies that the spectrum $\sigma(\mathbf{Q}_{n,h})$ is a subset of $[-1;+1]$. Let $1 = \lambda_{n,1} \geq \lambda_{n,2} \geq \ldots \geq \lambda_{n,j(n)} \geq -1$ be the eigenvalues of $\mathbf{Q}_{n,h}$, where in this enumeration, an eigenvalue is repeated as many times as its multiplicity.

To implement the spectral clustering algorithm, the data points of the partitioning problem are first embedded into $\mathbb{R}^\ell$ by using the eigenvectors of $\mathbf{Q}_{n,h}$ associated with the ℓ largest eigenvalues, namely $\lambda_{n,1}, \lambda_{n,2}, \ldots \lambda_{n,\ell}$. More precisely, fix a collection $V_{n,1}, V_{n,2}, \ldots, V_{n,\ell}$ of such eigenvectors with components respectively given by $V_{n,k} = \{V_{n,k,j}\}_{j\in J(n)}$, for $k = 1,\ldots,\ell$. Then the j^{th} data point, for j in $J(n)$, is represented by the vector $\rho_n(X_j)$ of the feature space $\mathbb{R}^\ell$ defined by $\rho_n(X_j) := \{V_{n,k,j}\}_{1\leq k\leq \ell}$. At last, the embedded points are partitioned using a classical clustering method, such as the k-means algorithm for instance.

2.3 Functional Operators Associated With the Matrices of the Algorithm

As exposed in the Introduction, some functional operators are associated with the matrices acting on $\mathbb{C}^{J(n)}$ defined in the previous paragraph. The link between matrices and functional operators is provided by the evaluation map defined in (3) below. As a consequence, asymptotic results on the clustering algorithm may be derived by studying first the limit behavior of these operators.

To this aim, let us first introduce some additional notation. For $\mathcal{D}$ a subset of $\mathbb{R}^d$, let $W(\mathcal{D})$ be the Banach space of complex-valued, bounded, and continuously differentiable functions with bounded gradient, endowed with the norm

$$\|g\|_W := \|g\|_\infty + \|Dg\|_\infty.$$

Consider the non-oriented graph whose vertices are the X_j's for j ranging in $J(n)$. The similarity matrix $\mathbf{K}_{n,h}$ gives random weights to the edges of the graph and the random transition matrix $\mathbf{Q}_{n,h}$ defines a random walk on the vertices of a random graph. Associated with this random walk is the transition operator $Q_{n,h} : W\left(\mathcal{L}_n(t)\right) \to W\left(\mathcal{L}_n(t)\right)$ defined for any function g by

$$Q_{n,h}g(x) := \int_{\mathcal{L}_n(t)} q_{n,h}(x,y)g(y)\mathbb{P}_n^t(dy).$$

In this equation, $\mathbb{P}_n^t$ is the discrete random probability measure given by

$$\mathbb{P}_n^t := \frac{1}{j(n)}\sum_{j\in J(n)} \delta_{X_j},$$

and

$$q_{n,h}(x,y) := \frac{k_h(y-x)}{K_{n,h}(x)}, \quad \text{where } K_{n,h}(x) := \int_{\mathcal{L}_n(t)} k_h(y-x)\mathbb{P}_n^t(dy). \tag{2}$$

In the definition of $q_{n,h}$, we use the convention that $0/0 = 0$, but this situation does not occur in the proofs of our results.

Given the *evaluation map* $\pi_n : W\big(\mathcal{L}_n(t)\big) \to \mathbb{C}^{j(n)}$ defined by

$$\pi_n(g) := \big\{g(X_j)\big\}_{j\in J(n)}, \tag{3}$$

the matrix $\mathbf{Q}_{n,h}$ and the operator $Q_{n,h}$ are related by $\mathbf{Q}_{n,h} \circ \pi_n = \pi_n \circ Q_{n,h}$. Using this relation, asymptotic properties of the spectral clustering algorithm may be deduced from the limit behavior of the sequence of operators $\{Q_{n,h}\}_n$. The difficulty, though, is that $Q_{n,h}$ acts on $W\big(\mathcal{L}_n(t)\big)$ and $\mathcal{L}_n(t)$ is a random set which varies with the sample. For this reason, we introduce a sequence of operators $\widehat{Q}_{n,h}$ acting on $W\big(\mathcal{L}(t)\big)$ and constructed from $Q_{n,h}$ as follows.

First of all, recall that under Assumption 1, the gradient of f does not vanish on the set $\{x \in \mathbb{R}^d : f(x) = t\}$. Since f is of class $\mathcal{C}^2$, a continuity argument implies that there exists $\varepsilon_0 > 0$ such that $\mathcal{L}_{t-\varepsilon_0}^{t+\varepsilon_0}$ contains no critical points of f. Under this condition, Lemma 17 states that $\mathcal{L}(t+\varepsilon)$ is diffeomorphic to $\mathcal{L}(t)$ for every ε such that $|\varepsilon| \leq \varepsilon_0$. In all of the following, it is assumed that ε_0 is small enough so that

$$\varepsilon_0/\alpha(\varepsilon_0) < h/2, \quad \text{where } \alpha(\varepsilon_0) := \inf\big\{\|Df(x)\|; x \in \mathcal{L}_{t-\varepsilon_0}^{t}\big\}. \tag{4}$$

Let $\{\varepsilon_n\}_n$ be a sequence of positive numbers such that $\varepsilon_n \leq \varepsilon_0$ for each n, and $\varepsilon_n \to 0$ as $n \to \infty$. In Lemma 17 an explicit diffeomorphism φ_n carrying $\mathcal{L}(t)$ to $\mathcal{L}(t-\varepsilon_n)$ is constructed, that is,

$$\varphi_n : \mathcal{L}(t) \xrightarrow{\cong} \mathcal{L}(t-\varepsilon_n).$$

The diffeomorphism φ_n induces the linear operator $\Phi_n : W\big(\mathcal{L}(t)\big) \to W\big(\mathcal{L}(t-\varepsilon_n)\big)$ defined by $\Phi_n g = g \circ \varphi_n^{-1}$.

Second, let Ω_n be the probability event defined by

$$\Omega_n = \Big[\|\widehat{f}_n - f\|_\infty \leq \varepsilon_n\Big] \cap \left[\inf\Big\{\|D\widehat{f}_n(x)\|, x \in \mathcal{L}_{t-\varepsilon_0}^{t+\varepsilon_0}\Big\} \geq \frac{1}{2}\|Df\|_\infty\right].$$

Note that on the event Ω_n, the following inclusions hold:

$$\mathcal{L}(t+\varepsilon_n) \subset \mathcal{L}_n(t) \subset \mathcal{L}(t-\varepsilon_n).$$

We assume that the indicator function $\mathbf{1}_{\Omega_n}$ tends to 1 almost surely as $n \to \infty$, which is satisfied by common density estimates $\widehat{f}_n$ under mild assumptions. For instance, consider a kernel density estimate with a Gaussian kernel. It is a classical exercise to prove that $\|\hat{f}_n - \mathbb{E}\hat{f}_n\|_\infty$ converges to 0 almost surely as n goes to infinity (see, e.g., Example 38 in Pollard, 1984, p. 35, or Chapter 3 in Prakasa Rao, 1983) under appropriate conditions on the bandwidth sequence. Moreover, under the conditions on f in Assumption 1, the norm of the gradient of f is uniformly bounded on $\mathbb{R}^d$, so by using a Taylor expansion, it is easy to prove that the bias term $\|\mathbb{E}\hat{f}_n - f\|_\infty \to 0$ as well. Hence $\|\hat{f}_n - f\|_\infty \to 0$ almost surely. Furthermore, under Assumption 1, $\|D^2 f\|$ is uniformly bounded on $\mathbb{R}^d$ so the same reasoning leads to the almost sure convergence to 0 of $\|D\hat{f}_n - Df\|_\infty$. Together, these facts imply that $\mathbf{1}_{\Omega_n} \to 1$ almost surely as $n \to \infty$.

We are now in a position to define the operator $\widehat{Q}_{n,h} : W\big(\mathcal{L}(t)\big) \to W\big(\mathcal{L}(t)\big)$. On the event Ω_n, for all function g in $W\big(\mathcal{L}(t)\big)$, we define $\widehat{Q}_{n,h}g$ by the relation

$$\widehat{Q}_{n,h}g(x) = \frac{1}{j(n)} \sum_{j\in J(n)} q_{n,h}(\varphi_n(x), X_j) g\big(\varphi_n^{-1}(X_j)\big), \quad \text{for all } x \in \mathcal{L}(t), \tag{5}$$

and we extend the definition of $\widehat{Q}_{n,h}$ to the whole probability space by setting it to the null operator on the complement Ω_n^c of Ω_n, that is, on Ω_n^c, the function $\widehat{Q}_{n,h}g$ is identically zero for each $g \in W\big(\mathcal{L}(t)\big)$. With a slight abuse of notation, we may note that $\widehat{Q}_{n,h} = \Phi_n^{-1} Q_{n,h} \Phi_n$, so that essentially, the operators $\widehat{Q}_{n,h}$ and $Q_{n,h}$ are conjugate and have equal spectra, which are in turn related to the spectrum of the matrix $\mathbf{Q}_{n,h}$. This is made precise in Proposition 1 below.

Proposition 1 *On the event Ω_n, the spectrum of the functional operator is $\widehat{Q}_{n,h}$ is $\sigma(\widehat{Q}_{n,h}) = \{0\} \cup \sigma(\mathbf{Q}_{n,h})$. Moreover, if if $\lambda \neq 0$, the eigenspaces are isomorphic, that is,*

$$\pi_n \Phi_n : N(\widehat{Q}_{n,h} - \lambda) \xrightarrow{\cong} N(\mathbf{Q}_{n,h} - \lambda),$$

where $\pi_n \Phi_n$ acts on $W\big(\mathcal{L}(t)\big)$ as $\phi_n \Phi_n g(x) = g\big(\varphi_n^{-1}(x)\big)$.

Proof From Equation (5), the range $R(\widehat{Q}_{n,h})$ of $\widehat{Q}_{n,h}$ is spanned by the finite collection of functions

$$f_j : \left| \begin{array}{lcl} \mathcal{L}(t) & \to & \mathbb{C} \\ x & \mapsto & q_{n,h}(\varphi_n(x), X_j), \end{array} \right.$$

for all $j \in J(n)$. Moreover, these functions form a basis of $R(\widehat{Q}_{n,h})$. To show this, let V be a vector in $\mathbb{C}^{J(n)}$ such that

$$\sum_{j \in J(n)} V_j f_j(x) = 0 \quad \text{for all } x \in \mathcal{L}(t).$$

By definition of $q_{n,h}$, setting $y = \varphi_n(x)$, we have

$$\sum_{j \in J(n)} V_j \frac{k_h(y - X_j)}{K_{n,h}(y)} = 0 \quad \text{for all } y \in \mathcal{L}(t - \varepsilon_n).$$

Since the support of k_h is hB, the support of the function $K_{n,h}$ is equal to $\bigcup_{j \in J(n)} (X_j + hB)$, and since k_h is positive, it follows that $V_j = 0$ for all j in $J(n)$. Hence $\{f_j : j \in J(n)\}$ is a basis of $R(\widehat{Q}_{n,h})$.

Now let g be an eigenfunction of $\widehat{Q}_{n,h}$ associated with an eigenvalue $\lambda \neq 0$. Then for all x in $\mathcal{L}(t)$

$$\frac{1}{j(n)} \sum_{j \in J(n)} q_{n,h}\big(\varphi_n(x), X_j\big) g\big(\varphi_n^{-1}(X_j)\big) = \lambda g(x). \tag{6}$$

Since we consider a non-zero eigenvalue, g is in the range of $\widehat{Q}_{n,h}$, and since the functions $\{f_j : j \in J(n)\}$ form a basis of $R(\widehat{Q}_{n,h})$, there exists a unique vector $V = \{V_j\}_{j \in J(n)} \in \mathbb{C}^{j(n)}$ such that

$$g(x) = \frac{1}{\lambda j(n)} \sum_{j \in J(n)} V_j q_{n,h}(\varphi_n(x), X_j), \quad x \in \mathcal{L}(t).$$

Therefore $V_j = g\big(\varphi_n^{-1}(X_j)\big)$ for all j in $J(n)$. Moreover, by evaluating (6) at any $x = \varphi_n^{-1}(X_i)$ with $i \in J(n)$,

$$\sum_{j \in J(n)} q_{n,h}\big(X_i, X_j\big) g\big(\varphi_n^{-1}(X_j)\big) = \lambda g\big(\varphi_n^{-1}(X_i)\big),$$

which implies that $\mathbf{Q}_{n,h} V = \lambda V$. Consequently, V is an eigenvector of $\mathbf{Q}_{n,h}$ associated with the eigenvalue λ. Hence

$$\sigma(\widehat{Q}_{n,h}) \subset \sigma(\mathbf{Q}_{n,h}) \cup \{0\}, \tag{7}$$

and by unicity of V, it follows that the map $\pi_n\Phi_n : N(\widehat{Q}_{n,h} - \lambda) \longrightarrow N(\mathbf{Q}_{n,h} - \lambda)$ is injective.

Conversely, let V be an eigenvector of the matrix $\mathbf{Q}_{n,h}$ associated with a non-zero eigenvalue λ. Consider the function g of $W\big(\mathcal{L}(t)\big)$ defined by

$$g(x) = \frac{1}{\lambda j(n)} \sum_{j \in J(n)} V_j q_{n,h}(\varphi_n(x), X_j), \quad \text{for all } x \in \mathcal{L}(t).$$

Observe that for all j in $J(n)$,

$$\begin{aligned} g\big(\varphi_n^{-1}(X_j)\big) &= \frac{1}{\lambda j(n)} \sum_{j' \in J(n)} q_{n,h}(X_j, X_{j'}) V_{j'} && \text{by definition of } g, \\ &= \frac{1}{\lambda j(n)} \sum_{j' \in J(n)} \frac{j(n)}{\mathbf{K}_{n,h}(j)} k_h(X_{j'} - X_j) V_{j'} && \text{by definition of } \mathbf{K}_{n,h} \text{ and } q_{n,h}, \\ &= \frac{1}{\lambda} \big(\mathbf{Q}_{n,h} V\big)_j = V_j && \text{since } V \text{ is an eigenvector.} \end{aligned}$$

Hence it follows that for all $x \in \mathcal{L}(t)$,

$$\begin{aligned} \widehat{Q}_{n,h} g(x) &= \frac{1}{j(n)} \sum_{j \in J(n)} q_{n,h}(\varphi_n(x), X_j) g\big(\varphi_n^{-1}(X_j)\big) && \text{using (5)} \\ &= \frac{1}{j(n)} \sum_{j \in J(n)} q_{n,h}(\varphi_n(x), X_j) V_j && \text{since } g\big(\varphi_n^{-1}(X_j)\big) = V_j. \\ &= \lambda g(x). \end{aligned}$$

Consequently,

$$\sigma(\mathbf{Q}_{n,h}) \subset \sigma(\widehat{Q}_{n,h}), \tag{8}$$

and the map $\pi_n\Phi_n : N(\widehat{Q}_{n,h} - \lambda) \longrightarrow N(\mathbf{Q}_{n,h} - \lambda)$ is surjective. Combining (7) and (8), and since 0 belongs to $\sigma(\mathbf{Q}_{n,h})$, we obtain the equality

$$\sigma(\widehat{Q}_{n,h}) = \{0\} \cup \sigma(\mathbf{Q}_{n,h}).$$

At last, since $\pi_n\Phi_n$ is both injective and surjective, the subspaces $N(\widehat{Q}_{n,h} - \lambda)$ and $N(\mathbf{Q}_{n,h} - \lambda)$ are isomorphic for any $\lambda \neq 0$. ■

Remark 2 *Albeit the relevant part of $\widehat{Q}_{n,h}$ is defined on Ω_n for technical reasons, this does not bring any difficulty as long as one is concerned with almost sure convergence. To see this, let $(\Omega, \mathcal{A}, P)$ be the probability space on which the X_i's are defined. Denote by Ω_∞ the event on which $\mathbf{1}_{\Omega_n}$ tends to 1, and recall that $P(\Omega_\infty) = 1$ by assumption. Thus, for every $\omega \in \Omega$, there exists a random integer $n_0(\omega)$ such that, for each $n \geq n_0(\omega)$, ω lies in Ω_n. Besides $n_0(\omega)$ is finite on Ω_∞. Hence in particular, if $\{Z_n\}$ is a sequence of random variables such that $Z_n \mathbf{1}_{\Omega_n}$ converges almost surely to some random variable Z_∞, then $Z_n \to Z_\infty$ almost surely.*

3. Operator Norm Convergence

In this section, we start by establishing the uniform convergence of an unnormalized empirical functional operator. The main operator norm convergence result (Theorem 4) is stated in Section 3.2. The proofs of these theorems rely on several auxiliary lemmas which are stated and proved in Section 5.

3.1 Unnormalized Operators

Let $r : \mathcal{L}(t-\varepsilon_0) \times \mathbb{R}^d \to \mathbb{R}$ be a given function. Define the linear operators R_n and R on $W\big(\mathcal{L}(t)\big)$ respectively by

$$R_n g(x) := \int_{\mathcal{L}_n(t)} r\big(\varphi_n(x), y\big) g\big(\varphi_n^{-1}(y)\big) \mathbb{P}_n^t(dy), \qquad \text{and} \qquad Rg(x) := \int_{\mathcal{L}(t)} r(x,y) g(y) \mu^t(dy).$$

Proposition 3 *Assume the following conditions on the function r:*
(i) r is continuously differentiable with compact support ;
(ii) r is uniformly bounded on $\mathcal{L}(t-\varepsilon_0) \times \mathbb{R}^d$, that is, $\|r\|_\infty < \infty$;
(iii) the differential $D_x r$ of the function r with respect to x is uniformly bounded on $\mathcal{L}(t-\varepsilon_0) \times \mathbb{R}^d$, that is, $\|D_x r\|_\infty := \sup\big\{\|D_x r(x,y)\| : (x,y) \in \mathcal{L}(t-\varepsilon_0) \times \mathbb{R}^d\big\} < \infty$.
Then, as $n \to \infty$,

$$\sup\Big\{\big\|R_n g - Rg\big\|_\infty : \|g\|_W \le 1\Big\} \to 0 \quad \textit{almost surely}.$$

The key argument for proving Proposition 3 is that the collection of functions

$$\Big\{y \mapsto r(x,y) g(y) \mathbf{1}_{\mathcal{L}(t)}(y) : x \in \mathcal{L}(t), \|g\|_{W(\mathcal{L}(t))} \le 1\Big\}$$

is Glivenko-Cantelli, which is proved in Lemma 13. Let us recall that a collection $\mathcal{F}$ of functions is said to be Glivenko-Cantelli, or to satisfy a uniform law of large number, if

$$\sup_{g \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^n g(X_i) - \mathbb{E}[X] \right| \to 0 \quad \text{almost surely,}$$

where $X, X_1, X_2, \ldots$ are i.i.d. random variables.

Proof In all this proof, we shall use the following convention: given a function g defined only on some subset $\mathcal{D}$ of $\mathbb{R}^d$, for any subset $\mathcal{A} \subset \mathcal{D}$, and any $x \in \mathbb{R}^d$, the notation $g(x)\mathbf{1}_{\mathcal{A}}(x)$ stands for $g(x)$ is $x \in \mathcal{A}$ and for 0 otherwise. Set

$$\begin{aligned}
S_n g(x) &:= \frac{1}{\mu(\mathcal{L}(t))} \frac{1}{n} \sum_{i=1}^n r\big(\varphi_n(x), X_i\big)\, g\big(\varphi_n^{-1}(X_i)\big) \mathbf{1}_{\mathcal{L}_n(t)}(X_i), \\
T_n g(x) &:= \frac{1}{\mu\big(\mathcal{L}(t)\big)} \frac{1}{n} \sum_{i=1}^n r\big(\varphi_n(x), X_i\big) g(X_i) \mathbf{1}_{\mathcal{L}(t)}(X_i), \\
U_n g(x) &:= \frac{1}{\mu\big(\mathcal{L}(t)\big)} \frac{1}{n} \sum_{i=1}^n r\big(x, X_i\big) g(X_i) \mathbf{1}_{\mathcal{L}(t)}(X_i).
\end{aligned}$$

and consider the inequality

$$\begin{aligned}
\big|R_n g(x) - Rg(x)\big| \le \big|R_n g(x) - S_n g(x)\big| + \big|S_n g(x) - T_n g(x)\big| \\
+ \big|T_n g(x) - U_n g(x)\big| + \big|U_n g(x) - Rg(x)\big|,
\end{aligned} \tag{9}$$

for all $x \in \mathcal{L}(t)$ and all $g \in W\big(\mathcal{L}(t)\big)$.

The first term in (9) is bounded uniformly by

$$\big|R_n g(x) - S_n g(x)\big| \le \left| \frac{n}{j(n)} - \frac{1}{\mu\big(\mathcal{L}(t)\big)} \right| \|r\|_\infty \|g\|_\infty$$

and since $j(n)/n$ tends to $\mu(\mathcal{L}(t))$ almost surely as $n \to \infty$, we conclude that

$$\sup\Big\{ \big\|R_n g - S_n g\big\|_\infty : \|g\|_W \le 1 \Big\} \to 0 \quad \text{a.s. as } n \to \infty. \tag{10}$$

For the second term in (9), we have

$$\begin{aligned} |S_n g(x) - T_n g(x)| &\le \frac{\|r\|_\infty}{\mu\big(\mathcal{L}(t)\big)} \frac{1}{n} \sum_{i=1}^n \big| g\big(\varphi_n^{-1}(X_i)\big) \mathbf{1}_{\mathcal{L}_n(t)}(X_i) - g(X_i) \mathbf{1}_{\mathcal{L}(t)}(X_i) \big| \\ &= \frac{\|r\|_\infty}{\mu\big(\mathcal{L}(t)\big)} \frac{1}{n} \sum_{i=1}^n g_n(X_i), \end{aligned} \tag{11}$$

where g_n is the function defined on the whole space $\mathbb{R}^d$ by

$$g_n(x) = \Big| g\big(\varphi_n^{-1}(x)\big) \mathbf{1}_{\mathcal{L}_n(t)}(x) - g(x) \mathbf{1}_{\mathcal{L}(t)}(x) \Big|.$$

Consider the partition of $\mathbb{R}^d$ given by $\mathbb{R}^d = B_{1,n} \cup B_{2,n} \cup B_{3,n} \cup B_{4,n}$, where

$$\begin{aligned} B_{1,n} &:= \mathcal{L}_n(t) \cap \mathcal{L}(t), \quad & B_{2,n} &:= \mathcal{L}_n(t) \cap \mathcal{L}(t)^c, \\ B_{3,n} &:= \mathcal{L}_n(t)^c \cap \mathcal{L}(t), \quad & B_{4,n} &:= \mathcal{L}_n(t)^c \cap \mathcal{L}(t)^c. \end{aligned}$$

The sum over i in (11) may be split into four parts as

$$\frac{1}{n} \sum_{i=1}^n g_n(X_i) = I_1(x,g) + I_2(x,g) + I_3(x,g) + I_4(x,g) \tag{12}$$

where

$$I_k(x,g) := \frac{1}{n} \sum_{i=1}^n g_n(X_i) \mathbf{1}\{X_i \in B_{k,n}\}.$$

First, $I_{4,n}(x,g) = 0$ since g_n is identically 0 on $B_{4,n}$. Second,

$$I_2(x,g) + I_3(x,g) \le \|g\|_\infty \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{\mathcal{L}(t) \Delta \mathcal{L}_n(t)}(X_i) \tag{13}$$

Applying Lemma 11 together with the almost sure convergence of $\mathbf{1}_{\Omega_n}$ to 1, we obtain that

$$\frac{1}{n} \sum_{j=1}^n \mathbf{1}_{\mathcal{L}(t) \Delta \mathcal{L}_n(t)}(X_j) \to 0 \quad \text{almost surely.} \tag{14}$$

Third,

$$\begin{aligned} I_1(x,g) &\le \sup_{x \in \mathcal{L}(t)} \Big| g\big(\varphi_n^{-1}(x)\big) - g(x) \Big| \le \|D_x g\|_\infty \sup_{x \in \mathcal{L}(t)} \|\varphi_n^{-1}(x) - x\| \\ &\le \|D_x g\|_\infty \sup_{x \in \mathcal{L}(t)} \|x - \varphi_n(x)\| \to 0 \end{aligned} \tag{15}$$

as $n \to \infty$ by Lemma 17. Thus, combining (11), (12), (13), (14) and (15) leads to

$$\sup\left\{\left\|S_n g - T_n g\right\|_\infty : \|g\|_W \leq 1\right\} \to 0 \quad \text{a.s. as } n \to \infty. \tag{16}$$

For the third term in (9), using the inequality

$$\left|r\big(\varphi_n(x), X_i\big) - r\big(x, X_i\big)\right| \leq \|D_x r\|_\infty \sup_{x \in \mathcal{L}(t)} \|\varphi_n(x) - x\|$$

we deduce that

$$\left|T_n g(x) - U_n g(x)\right| \leq \frac{1}{\mu\big(\mathcal{L}(t)\big)} \|g\|_\infty \|D_x r\|_\infty \sup_{x \in \mathcal{L}(t)} \|\varphi_n(x) - x\|.$$

and so

$$\sup\left\{\left\|T_n g - U_n g\right\|_\infty : \|g\|_W \leq 1\right\} \to 0 \quad \text{a.s. as } n \to \infty, \tag{17}$$

by Lemma 17.

At last, for the fourth term in (9), we conclude by Lemma 13 that

$$\sup\left\{\left\|U_n g - Rg\right\|_\infty : \|g\|_W \leq 1\right\} \to 0 \quad \text{a.s. as } n \to \infty.$$

Finally, reporting (10), (16) and (17) in (9) yields the desired result. ■

3.2 Normalized Operators

Theorem 4 states that $\widehat{Q}_{n,h}$ converges in operator norm to the limit operator $Q_h : W\big(\mathcal{L}(t)\big) \to W\big(\mathcal{L}(t)\big)$ defined by

$$Q_h g(x) = \int_{\mathcal{L}(t)} q_h(x,y) g(y) \mu^t(dy), \tag{18}$$

where μ^t denotes the conditional distribution of X given the event $\big[X \in \mathcal{L}(t)\big]$, and where

$$q_h(x,y) = \frac{k_h(y-x)}{K_h(x)}, \quad \text{with } K_h(x) = \int_{\mathcal{L}(t)} k_h(y-x) \mu^t(dy). \tag{19}$$

Theorem 4 (Operator Norm Convergence) *Suppose that Assumptions 1 and 2 hold. We have*

$$\left\|\widehat{Q}_{n,h} - Q_h\right\|_W \to 0 \quad \textit{almost surely as } n \to \infty.$$

Proof We will prove that, as $n \to \infty$, almost surely,

$$\sup\left\{\left\|\widehat{Q}_{n,h} g - Q_h g\right\|_\infty : \|g\|_W \leq 1\right\} \to 0 \tag{20}$$

and

$$\sup\left\{\left\|D_x\big[\widehat{Q}_{n,h} g\big] - D_x\big[Q_h g\big]\right\|_\infty : \|g\|_W \leq 1\right\} \to 0 \tag{21}$$

To this aim, we introduce the operator $\widetilde{Q}_{n,h}$ acting on $W(\mathcal{L}(t))$ as

$$\widetilde{Q}_{n,h} g(x) = \int_{\mathcal{L}_n(t)} q_h(\varphi_n(x), y) g\big(\varphi_n^{-1}(y)\big) \mathbb{P}_n^t(dy).$$

Proof of (20) For all $g \in W\big(\mathcal{L}(t)\big)$, we have

$$\big\|\widehat{Q}_{n,h}g - Q_h g\big\|_\infty \le \big\|\widehat{Q}_{n,h}g - \widetilde{Q}_{n,h}g\big\|_\infty + \big\|\widetilde{Q}_{n,h}g - Q_h g\big\|_\infty. \tag{22}$$

First, by Lemma 14, the function $r = q_h$ satisfies the condition in Proposition 3, so that

$$\sup\Big\{\|\widetilde{Q}_{n,h}g - Q_h g\|_\infty : \|g\|_W \le 1\Big\} \to 0 \tag{23}$$

with probability one as $n \to \infty$.

Next, since $\|q_h\|_\infty < \infty$ by Lemma 14, there exists a finite constant C_h such that,

$$\|\widetilde{Q}_{n,h}g\|_\infty \le C_h \quad \text{for all } n \text{ and all } g \text{ with } \|g\|_W \le 1. \tag{24}$$

By definition of $q_{n,h}$, for all x, y in the level set $\mathcal{L}(t)$, we have

$$q_{n,h}(x,y) = \frac{K_h(x)}{K_{n,h}(x)} q_h(x,y).$$

So

$$\begin{aligned}\Big|\widehat{Q}_{n,h}g(x) - \widetilde{Q}_{n,h}g(x)\Big| &= \left|\frac{K_n\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} - 1\right| \Big|\widetilde{Q}_{n,h}g(x)\Big| \\ &\le C_h \sup_{x\in\mathcal{L}(t)} \left|\frac{K_n\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} - 1\right|,\end{aligned}$$

where C_h is as in (24). Applying Lemma 16 yields

$$\sup\Big\{\|\widehat{Q}_{n,h}g - \widetilde{Q}_{n,h}g\|_\infty : \|g\|_W \le 1\Big\} \to 0 \tag{25}$$

with probability one as $n \to \infty$. Reporting (23) and (25) in (22) proves (20).

Proof of (21) We have

$$\Big\|D_x\Big[\widehat{Q}_{n,h}g\Big] - D_x\Big[Q_h g\Big]\Big\|_\infty \le \Big\|D_x\Big[\widehat{Q}_{n,h}g\Big] - D_x\Big[\widetilde{Q}_h g\Big]\Big\|_\infty + \Big\|D_x\Big[\widetilde{Q}_{n,h}g\Big] - D_x\Big[Q_h g\Big]\Big\|_\infty. \tag{26}$$

The second term in right han side of (26) is bounded by

$$\Big\|D_x\Big[\widetilde{Q}_{n,h}g\Big] - D_x\Big[Q_h g\Big]\Big\|_\infty \le \big\|D_x\varphi_n\big\|_\infty \big\|R_n g - Rg\big\|_\infty,$$

where

$$\begin{aligned}R_n g(x) &:= \int_{\mathcal{L}_n(t)} (D_x q_h)(\varphi_n(x), y) g\big(\varphi_n^{-1}(y)\big) \mathbb{P}_n^t(dy) \quad \text{and} \\ Rg(x) &:= \int_{\mathcal{L}(t)} (D_x q_h)(\varphi_n(x), y) g\big(\varphi_n^{-1}(y)\big) \mu^t(dy).\end{aligned}$$

By lemma 17, $x \mapsto D_x\varphi_n(x)$ converges to the identity matrix I_d of $\mathbb{R}^d$, uniformly in x over $\mathcal{L}(t)$. So $\|D_x\varphi_n(x)\|$ is bounded by some finite constant C_φ uniformly over n and $x \in \mathcal{L}(t)$ and

$$\left\| D_x\left[\widetilde{Q}_{n,h}g\right] - D_x\left[Q_h g\right]\right\|_\infty \leq C_\varphi \left\|R_n g - Rg\right\|_\infty.$$

By Lemma 14, the map $r : (x,y) \mapsto D_x q_h(x,y)$ satisfies the conditions in Proposition 3. Thus, $\|R_n g - Rg\|_\infty$ converges to 0 almost surely, uniformly over g in the unit ball of $W(\mathcal{L}(t))$, and we deduce that

$$\sup\left\{ \left\| D_x\left[\widetilde{Q}_{n,h}g\right] - D_x\left[Q_h g\right]\right\|_\infty : \|g\|_W \leq 1 \right\} \to 0 \quad \text{a.s. as } n \to \infty. \tag{27}$$

For the first term in right hand side of (26), observe first that there exists a constant C'_h such that, for all n and all g in the unit ball of $W\big(\mathcal{L}(t)\big)$,

$$\|R_{n,h}g\|_\infty \leq C'_h, \quad \text{for all } n \text{ and all } g \text{ with } \|g\|_W \leq 1, \tag{28}$$

by Lemma 14.

On the one hand, we have

$$D_x\big[q_{n,h}(\varphi_n(x),y)\big] = \frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} D_x\varphi_n(x)(D_x q_h)\big(\varphi_n(x),y\big) + D_x\left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)}\right] q_h\big(\varphi_n(x),y\big).$$

Hence,

$$D_x\left[\widehat{Q}_{n,h}g(x)\right] = \frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} D_x\varphi_n(x) R_n g(x) + D_x\left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)}\right] \widetilde{Q}_{n,h}g(x).$$

On the other hand, since $D_x\big[q_h\big(\varphi_n(x),y\big)\big] = D_x\varphi_n(x)(D_x q_h)\big(\varphi_n(x),y\big)$,

$$D_x\left[\widetilde{Q}_{n,h}g(x)\right] = D_x\varphi_n(x) R_n g(x).$$

Thus,

$$D_x\left[\widehat{Q}_{n,h}g(x)\right] - D_x\left[\widetilde{Q}_h g(x)\right] = D_x\left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)}\right]\widetilde{Q}_{n,h}g(x) + \left(\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} - 1\right) D_x\varphi_n(x) R_n g(x).$$

Using the Inequalities (24) and (28), we obtain

$$\left\| D_x\left[\widehat{Q}_{n,h}g\right] - D_x\left[\widetilde{Q}_h g\right]\right\|_\infty \leq C_h \sup_{x\in\mathcal{L}(t)} \left| D_x\left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)}\right]\right| + C'_h C_\varphi \sup_{x\in\mathcal{L}(t)} \left| \frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} - 1\right|.$$

and by applying Lemma 16, we deduce that

$$\sup\left\{ \left\| D_x\left[\widehat{Q}_{n,h}g\right] - D_x\left[\widetilde{Q}_h g\right]\right\|_\infty : \|g\|_W \leq 1 \right\} \to 0 \quad \text{a.s. as } n \to \infty. \tag{29}$$

Reporting (27) and (29) in (26) proves (21). ■

4. Consistency of the Algorithm

The consistency of the algorithm relies on the operator norm convergence of $\widehat{Q}_{n,h}$ to the limit operator Q_h (Theorem 4), on the spectral properties of Q_h stated below in Section 4.1, and on the results collected in Appendix B on the perturbation theory of linear operators, The starting point is the fact that, provided that $h < d_{\min}$, the connected components of the level set $\mathcal{L}(t)$ are the recurrent classes of the Markov chain whose transitions are defined by Q_h. Indeed, this process cannot jump from one component to another component. Hence Q_h defines the desired clustering via its eigenspace corresponding to the eigenvalue 1, since this latter is spanned by the characteristic functions of the connected components of $\mathcal{L}(t)$, as stated in Proposition 6 below.

In Section 4.2, the consistency of the clustering is obtained in Theorem 7 in the case where the scale parameter h is lower than d_{min} defined in (1), which is the minimum distance between any two connected components of $\mathcal{L}(t)$. Using the continuity of Q_h in h, we then obtain the main consistency in Theorem 10 of Section 4.3, where h is allowed to be larger than d_{min}, up to a value depending only on the underlying density f.

4.1 Properties of the Limit Operator Q_h When $h < d_{min}$

The transition kernel $q_h(x,dy) := q_h(x,y)\mu^t(dy)$ associated with the operator Q_h defines a Markov chain with state space $\mathcal{L}(t)$, which is not countable. Familiar notions such as irreducibility, aperiodicity, and positive recurrence, which are valid for a Markov chain on a countable state space, may be extended to the non-countable case. The relevant definitions and materials on Markov chains on a general state space are summarized in Appendix C. The properties of the Markov chain with transition kernel $q_h(x,dy)$ are stated in Proposition 5 below.

Recall that $\mathcal{L}(t)$ has ℓ connected components $\mathcal{C}_1,\ldots,\mathcal{C}_\ell$ and that d_{min}, defined in (1), is the minimal distance between the connected components of $\mathcal{L}(t)$.

Proposition 5 *Consider the Markov chain with state space $\mathcal{L}(t)$ and transition kernel $q_h(x,dy)$, and assume that $h < d_{min}$.*
1. The chain is Feller and topologically aperiodic.
2. When started at a point x in some connected component of the state space, the chain evolves within this connected component only.
3. When the state space is reduced to some connected component of $\mathcal{L}(t)$, the chain is open set irreducible and positive Harris recurrent.
4. When the state space is reduced to some connected component $\mathcal{C}_k$ of $\mathcal{L}(t)$, the Markov chain has a unique invariant distribution $\nu_k(dy)$ and, for all $x \in \mathcal{C}_k$, the sequence of distributions $\left\{q_h^n(x,dy)\right\}_{n\in\mathbb{N}}$ over $\mathcal{C}_k$ converges in total variation to $\nu_k(dy)$.

Proof Denote by $\{\xi_n\}$ the Markov chain with transition kernel $q_h(x,dy)$. For all $x \in \mathcal{L}(t)$, the distribution $q_h(x,dy) = q_h(x,y)\mu^t(dy)$ is absolutely continuous with respect to the Lebesgue measure, with density $y \mapsto f_h(x,y)$ defined by

$$f_h(x,y) = q_h(x,y)\frac{f(y)}{\int_{y'\in\mathcal{L}(t)} f(y')dy'}\mathbf{1}_{\mathcal{L}(t)}(y).$$

Since the similarity function k_h and the density f are both continuous, the map $(x,y) \mapsto f_h(x,y)$ is continuous.

Now, by induction on n, the distribution of ξ_n conditioned by $\xi_0 = x$, which is $q_h^{n+1}(x,dy)$ is also absolutely continuous with respect to the Lebesgue measure and its density $y \mapsto f_h^n(x,y)$ satisfies

$$f_h^n(x,y) = \int_{z\in\mathcal{L}(t)} f_h^{n-1}(x,z) f_h(z,y) dz = \int_{z\in\mathcal{L}(t)} f_h(x,z) f_h^{n-1}(z,y) dz, \tag{30}$$

where the last equality follows from the Markov property. Moreover, one easily sees by induction that the map $(x,y) \mapsto f_h^n(x,y)$ is continuous.

1. Since the similarity function k_h is continuous, with compact support hB, the map

$$x \mapsto Q_h g(x) = \int_{\mathcal{L}(t)} q_h(x,dy) g(y)$$

is continuous for every bounded, measurable function g. Hence, the chain is Feller.

Now we have to prove that the chain is topologically aperiodic, that is, that $q_h^n(x, x+\eta B) > 0$ for each $x \in \mathcal{L}(t)$, for all $n \geq 1$ and $\eta > 0$, where $q_h^n(x,\cdot)$ is the distribution of ξ_n conditioned on $\xi_0 = x$. Since the distribution $q_h^n(x,\cdot)$ admits a continuous density $f_h^n(x,\cdot)$, it is enough to prove that $f_h^n(x,x) > 0$. Since k_h is bounded from below on $(h/2)B$ by Assumption 2, the density $f_h(x,y)$ is strictly positive for all $y \in x + hB/2$. By induction over n, using (30), $f_h^n(x,x) > 0$ and the chain is topologically aperiodic.

2. Without loss of generality, since the numbering of the connected components is arbitrary, assume that $x \in \mathcal{C}_1$. Let y be a point of $\mathcal{L}(t)$ which does not belong to $\mathcal{C}_1$. Then $\|y-x\| \geq d_{min} > h$ so that $q_h(x,y) = 0$. Whence,

$$P_x(\xi_1 \in \mathcal{C}_1) = q_h(x,\mathcal{C}_1) = \int_{\mathcal{C}_1} q_h(x,y)\mu^t(dy) = \int_{\mathcal{L}(t)} q_h(x,y)\mu^t(dy) = 1.$$

3. Assume that the state space is reduced to $\mathcal{C}_1$. In order to prove that the chain is open set irreducible, it is enough to prove that, for each $x, y \in \mathcal{C}_1$ and $\eta > 0$, there exists some integer N such that $P_x(\xi_N \in y + \eta B) = q_h^N(x, y+\eta B)$ is positive. Observe that $q_h^n(x,dy)$ is the distribution with density

$$q_h^n(x,y) = \int_{x_1,\ldots,x_{n-1}\in\mathcal{C}_1} q_h(x,x_1) q_h(x_1,x_2) \ldots q_h(x_{n-1},y) dx_1 dx_2 dx_{n-1}$$

and $(x_1,\ldots,x_{n-1}) \mapsto q_h(x,x_1)q_h(x_1,x_2)\ldots q_h(x_{n-1},y)$ is continuous. Hence, it is enough to prove that there exists some integer N and a finite sequence $x_1, \ldots x_N$ such that

$$q_h(x,x_1)q_h(x_1,x_2)\ldots q_h(x_{N-1},y) > 0.$$

Since $\mathcal{C}_1$ is connected, there exists a finite sequence $x_0, x_1, \ldots x_N$ of points in $\mathcal{C}_1$ such that $x_0 = x$, $x_N = y$, and $\|x_i - x_{i+1}\| \leq h/2$ for each i. Therefore

$$q_h(x,x_1)q_h(x_1,x_2)\ldots q_h(x_{N-1},y) > 0$$

which proves that the chain is open set irreducible.

Since $\mathcal{C}_1$ is compact, the chain is non-evanescent, and so it is Harris recurrent. Recall that $k(x) = k(-x)$ from Assumption 2. Therefore $k_h(y-x) = k_h(x-y)$ which yields

$$K_h(x) q_h(x,dy)\mu^t(dx) = K_h(y) q_h(y,dx)\mu^t(dy).$$

By integrating the previous relation with respect to x over $\mathcal{C}_1$, one may verify that $K_h(x)\mu^t(dx)$ is an invariant measure. At last $\int_{\mathcal{C}_1} K_h(x)\mu^t(dx) < \infty$, which proves that the chain is positive.

4. This ergodic property is a direct application of the last part of Appendix C. ■

Proposition 6 *Assume that $h < d_{min}$. If g is continuous and $Q_h g = g$, then g is constant on the connected components of $\mathcal{L}(t)$.*

Proof The numbering of the connected components is arbitrary. Hence it is enough to prove that g is constant over $\mathcal{C}_1$. For this, fix x in $\mathcal{C}_1$ and note that $g = Q_h g$ implies $g = Q_h^n g$ for any $n \geq 1$. By Proposition 5, the chain is open set irreducible, topologically aperiodic, and positive Harris recurrent on $\mathcal{C}_1$. Moreover, $q_h^n(x,dy)$ converges in total variation norm to $\nu_1(dy)$, where ν_1 is the unique invariant distribution when state space is reduced to $\mathcal{C}_1$. Specifically,

$$Q_h^n g(x) \longrightarrow \int_{\mathcal{C}_1} g(y)\nu_1(dy) \quad \text{as } n \to \infty.$$

Hence, for every x in $\mathcal{C}_1$,

$$g(x) = \int_{\mathcal{C}_1} g(y)\nu_1(dy),$$

and since the last integral does not depend on x, it follows that g is a constant function on $\mathcal{C}_1$. ■

4.2 Spectral Convergence

Theorem 7 states that the representation of the extracted part of the data set into the feature space $\mathbb{R}^\ell$ (see the end of Section 2.2) tends to concentrate around ℓ different centroids. Moreover, each of these centroids corresponds to a cluster, that is, to a connected component of $\mathcal{L}(t)$. As a trivial consequence, any partitioning algorithm (e.g., k-means) applied in the feature space will asymptotically yield the desired clustering. In other words, the clustering algorithm is consistent.

More precisely, using the convergence in operator norm of $\widehat{Q}_{n,h}$ towards Q_h, together with the results of functional analysis given in Appendix B, we obtain the following Theorem which describes the asymptotic behavior of the algorithm. Let us denote by $J(\infty)$ the set of integers j such that X_j is in the level set $\mathcal{L}(t)$. For all $j \in J(\infty)$, define $k(j)$ as the integer such that $X_j \in \mathcal{C}_{k(j)}$.

Theorem 7 *Suppose that Assumptions 1 and 2 hold, and that h is in $(0; d_{min})$.*
1. The first ℓ eigenvalues $\lambda_{n,1}, \lambda_{n,2}, \ldots, \lambda_{n,\ell}$ of $\mathbf{Q}_{n,h}$ converge to 1 almost surely as $n \to \infty$, and there exists $\eta_0 > 0$ such that for all $j > \ell$, $\lambda_{n,j}$ belongs to $\{z : |z-1| \geq \eta_0\}$ for n large enough, with probability one.
2. There exists a sequence $\{\xi_n\}_n$ of invertible linear transformations of $\mathbb{R}^\ell$ such that, for all $j \in J(\infty)$, $\xi_n \rho_n(X_j)$ converges almost surely to $e_{k(j)}$, where $e_{k(j)}$ is the vector of $\mathbb{R}^\ell$ whose components are all 0 except the $k(j)^{\text{th}}$ component equal to 1.

Proof 1. The spectrum of Q_h may be decomposed as $\sigma(Q_h) = \sigma_1(Q_h) \cup \sigma_2(Q_h)$, where $\sigma_1(Q_h) = \{1\}$ and where $\sigma_2(Q_h) = \sigma(Q_h) \setminus \{1\}$. Since 1 is an isolated eigenvalue, there exists η_0 in the open interval $(0;1)$ such that $\sigma(Q_h) \cap \{z \in \mathbb{C} : |z-1| \leq \eta_0\}$ is reduced to the singleton $\{1\}$. Moreover,

1 is an eigenvalue of Q_h of multiplicity ℓ, by Proposition 6. Hence by Theorem 18, $W\big(\mathcal{L}(t)\big)$ decomposes into $W\big(\mathcal{L}(t)\big) = M_1 \oplus M_2$ where $M_1 = N(Q_h - 1)$ and M_2 is mapped into itself by Q_h.

Split the spectrum of $\widehat{Q}_{n,h}$ as $\sigma\big(\widehat{Q}_{n,h}\big) = \sigma_1\big(\widehat{Q}_{n,h}\big) \cup \sigma_2\big(\widehat{Q}_{n,h}\big)$, where

$$\sigma_1\big(\widehat{Q}_{n,h}\big) = \sigma\big(\widehat{Q}_{n,h}\big) \cap \big\{z \in \mathbb{C} : |z-1| < \eta_0\big\}.$$

By Theorem 18, this decomposition of the spectrum of $\widehat{Q}_{n,h}$ yields a decomposition of $W\big(\mathcal{L}(t)\big)$ as $W\big(\mathcal{L}(t)\big) = M_{n,1} \oplus M_{n,2}$, where $M_{n,1}$ and $M_{n,2}$ are stable subspaces under $\widehat{Q}_{n,h}$ and

$$M_{n,1} := \bigoplus_{\lambda \in \sigma_1(\widehat{Q}_{n,h})} N(\widehat{Q}_{n,h} - \lambda).$$

By Proposition 1, $\sigma(\widehat{Q}_{n,h}) = \sigma(\mathbf{Q}_{n,h}) \cup \{0\}$. Statement 6 of Theorem 19 implies that, for all n large enough, the total multiplicity of the eigenvalues in $\sigma_1(\widehat{Q}_{n,h})$ is $\dim(M_1) = \dim(N(Q_h - 1)) = \ell$. Hence, for all $j > \ell$, $\lambda_{n,j}$ belongs to $\{z : |z-1| \geq \eta_0\}$. Moreover, statement 4 of Theorem 19 proves that the first ℓ eigenvalues converges to 1.

2. In addition to the convergence of the eigenvalues of $\mathbf{Q}_{n,h}$, the convergence of the eigenspaces also holds. More precisely, let Π be the projector on $M_1 = N(Q_h - 1)$ along M_2 and Π_n the projector on $M_{n,1}$ along $M_{n,2}$. Statements 2, 3, 5 and 6 of Theorem 19 leads to

$$\|\Pi_n - \Pi\|_W \to 0 \quad a.s. \tag{31}$$

and the dimension of $M_{n,1}$ is equal to ℓ for all n large enough.

Denote by $E_{n,1}$ the subspace of $\mathbb{C}^{j(n)}$ spanned by the eigenvectors of $\mathbf{Q}_{n,h}$ corresponding to the eigenvalues $\lambda_{n,1}, \ldots \lambda_{n,\ell}$. Since

$$M_{n,1} = \bigoplus_{\lambda \in \sigma_1(\widehat{Q}_{n,h})} N(\widehat{Q}_{n,h} - \lambda) \quad \text{and} \quad E_{n,1} = \bigoplus_{\lambda \in \sigma_1(\mathbf{Q}_{n,h})} N(\mathbf{Q}_{n,h} - \lambda),$$

by Proposition 1 the map $\pi_n \Phi_n$ induces an isomorphism between $M_{n,1}$ and $E_{n,1}$. Moreover, Π_n induces a morphism $\tilde{\Pi}_n$ from M_1 to $M_{n,1}$ which converges to the identity map of M_1 in W-norm by (31). Hence, if n is large enough, $\tilde{\Pi}_n$ is invertible and we have the following isomorphisms of vector spaces:

$$\tilde{\Pi}_n : M_1 \xrightarrow{\cong} M_{n,1} \quad \text{and} \quad \pi_n \Phi_n : M_{n,1} \xrightarrow{\cong} E_{n,1}. \tag{32}$$

By Proposition 6, the functions $g_k := \mathbf{1}_{C_k}$, $k = 1, 2 \ldots, \ell$, form a basis of $M_1 = N(Q_h - 1)$. Using the isomorphisms of (32), we may define for all $k \in \{1, \ldots \ell\}$,

$$g_{n,k} := \tilde{\Pi}_n g_k, \qquad \text{and} \qquad \vartheta_{n,k} := \pi_n \Phi_n g_{n,k} = \pi_n \Phi_n \tilde{\Pi}_n g_k.$$

Then the collections $\{g_{n,k}\}_{k=1,\ldots,\ell}$ and $\{\vartheta_{n,k}\}_{k=1,\ldots,\ell}$ are a basis of $M_{n,1}$ and $E_{n,1}$ respectively. Moreover, for all $k \in \{1, \ldots, \ell\}$, $g_{n,k}$ converges to $\mathbf{1}_{C_k}$ in W-norm by (31). And, as $n \to \infty$, if $j \in J(\infty)$,

$$\vartheta_{n,k,j} = \tilde{\Pi}_n(\mathbf{1}_{C_k}) \circ \varphi_n^{-1}(X_j) \to \mathbf{1}_{C_k}(X_j) = \begin{cases} 1 & \text{if } k = k(j), \\ 0 & \text{otherwise.} \end{cases} \tag{33}$$

The eigenvectors $V_{n,1}, \ldots, V_{n,\ell}$ chosen in the algorithm form another basis of $E_{n,1}$. Hence, there exists a matrix ξ_n of dimension $\ell \times \ell$ such that

$$\vartheta_{n,k} = \sum_{i=1}^{\ell} \xi_{n,k,i} V_{n,i}.$$

Hence the j^{th} component of $\vartheta_{n,k}$, for all $j \in J(n)$, may be expressed as

$$\vartheta_{n,k,j} = \sum_{i=1}^{\ell} \xi_{n,k,i} V_{n,i,j}.$$

Since $\rho_n(X_j)$ is the vector of $\mathbb{R}^\ell$ with components $\{V_{n,i,j}\}_{i=1,\ldots,\ell}$, the vector $\vartheta_{n,\bullet,j} = \{\vartheta_{n,k,j}\}_k$ of $\mathbb{R}^\ell$ is related to $\rho_n(X_j)$ by the linear transformation ξ_n, that is,

$$\vartheta_{n,\bullet,j} = \xi_n \rho_n(X_j).$$

The convergence of $\vartheta_{n,\bullet,j}$ to $e_{k(j)}$ then follows from (33) and Theorem 7 is proved. ■

Remark 8 *The last step of the spectral clustering algorithm consists in partitioning the transformed data in the feature space, which can be performed by a standard clustering algorithm, like the k-means algorithm or a hierarchical clustering. Theorem 7 states that there exists a choice for a basis of ℓ eigenvectors such that the transformed data concentrates on the ℓ canonical basis vectors e_k of $\mathbb{R}^\ell$. Consequently, upon choosing a suitable collection $V_{n,1}, \ldots, V_{n,\ell}$ of eigenvectors, for any $\varepsilon > 0$, with probability one, for n large enough, the transformed data $\rho_n(X_j)$'s belong to the union of balls centered at $e_1, \ldots, e_\ell$ and of radius ε. Combining this result with known asymptotic properties of the aforementioned clustering algorithms leads to the desired partition.*

For instance, a hierarchical agglomerative method with single linkage allows to separate groups provided that the minimal distance between the groups is larger than the maximal diameter of the groups. In the preceding display, by choosing ε such that $2\varepsilon < \sqrt{2}$, with probability one for n large enough the points belong to ℓ balls of diameter 2ε which are all at a distance strictly larger than 2ε. Consequently, cutting the dendrogram tree of the single linkage hierarchical clustering at a height 2ε will correctly separate the groups, and the algorithm is consistent.

Similarly, for the k-means algorithm, we may note that, upon choosing a suitable basis of eigenvectors, the empirical measure associated with the transformed data converges to a discrete measure supported by the canonical vectors $e_1, \ldots, e_\ell$. Consistency of the grouping then follows from the well-known properties of the vector quantization method; see Pollard (1981).

The existence of an appropriate choice of eigenvectors is guaranteed by Theorem 7. How to choose such a collection of eigenvectors in practice is left for future research. In this direction, we may note that the two clustering methods considered above (i.e., k-means and hierarchical) are invariant by isometries. So the main question concerns the choice of the normalization of an arbitrary collection of eigenvectors.

Remark 9 *Note that if one is only interested in the consistency property, then this result could be obtained through another route. Indeed, it is shown in Biau et al. (2007) that the neighborhood graph with connectivity radius h has asymptotically the same number of connected components as*

the level set. Hence, splitting the graph into its connected components leads to the desired clustering as well. But Theorem 7, by giving the asymptotic representation of the data when embedded in the feature space $\mathbb{R}^\ell$, provides additional insight into spectral clustering algorithms. In particular, Theorem 7 provides a rationale for the heuristic of Zelnik-Manor and Perona (2004) for automatic selection of the number of groups. Their idea is to quantify the amount of concentration of the points embedded in the feature space, and to select the number of groups leading to the maximal concentration. Their method compared favorably with the eigengap heuristic considered in von Luxburg (2007).

4.3 Further Spectral Convergence

Naturally, the selection of the number of groups is also linked with the choice of the parameter h. In this direction, let us emphasize that the operators $\widehat{Q}_{n,h}$ and Q_h depend continuously on the scale parameter h. Thus, the spectral properties of both operators will be close to the ones stated in Theorem 7 if h is in a neighborhood of the interval $(0; d_{min})$. This follows from the continuity of an isolated set of eigenvalues, as stated in Appendix B. In particular, the sum of the eigenspaces of Q_h associated with the eigenvalues close to 1 is spanned by functions that are close to (in $W(\mathcal{L}(t))$-norm) the characteristic functions of the connected components of $\mathcal{L}(t)$. Hence, the representation of the data set in the feature space $\mathbb{R}^\ell$ still concentrates on some neighborhoods of e_k, $1 \leq k \leq \ell$ and a simple clustering algorithm such as the k-means algorithm will still lead to the desired partition. This is made precise in the following Theorem.

Theorem 10 *Suppose that assumptions 1 and 2 hold. There exists $h_{max} > d_{min}$ which depends only on the density f, such that, for any $h \in (0; h_{max})$, the event "for all n large enough, the representation of the extracted data set in the feature space, namely $\{\rho_n(X_j)\}_{j \in J(n)}$, concentrates in ℓ cubes of $\mathbb{R}^\ell$ that do not overlap" has probability one. Moreover, on this event of probability one, the ℓ cubes are in one-to-one correspondence with the ℓ connected component of $\mathcal{L}(t)$. Hence, for all n large enough, each $\rho_n(X_j)$ with $j \in J(\infty)$ is in the cube corresponding to the $k(j)^{th}$ cluster for all n large enough.*

This result contrasts with the graph techniques used to recover the connected components, as in, for example, Biau et al. (2007), where an unweighted graph is defined by connecting two observations if and only if their distance is smaller than h. The partition is then obtained by the connected components of the graph. However, when h is taken slightly larger than the critical value d_{min}, at least two connected components cannot be separated using the graph partitioning algorithm.

Proof Let us begin with the following consequence of Proposition 6. For all $h \leq d_{min}$ the ℓ largest eigenvalues of Q_h are all equal to 1 and the corresponding eigenspace is spanned by the indicator functions of the connected components of the t-level set. Moreover, 1 is an isolated eigenvalue of $Q_{d_{min}}$, that is, there exists η_0 in the interval $(0;1)$ such that $\sigma(Q_{d_{min}}) \cap \{z \in \mathbb{C} : |z-1| < \eta_0\}$ is the singleton $\{1\}$.

We choose an arbitrary constant C_0 in $(0;1/2)$. Since $h \mapsto Q_h$ is continuous for the topology of the operator norm, Theorem 19 implies that there exists a neighborhood $(h_{min}; h_{max})$ of d_{min} such that, for all h in $(h_{min}; h_{max})$,

(i) Q_h has exactly ℓ eigenvalues in $\{z \in \mathbb{C} : |z-1| < \eta_0\}$;

(ii) the sum of the corresponding eigenspaces of Q_h is spanned by ℓ functions, say $g_1, \ldots, g_\ell$, at distance (in $\|\cdot\|_W$-norm) less than $C_0/2$ from the indicator functions of the connected components

of $\mathcal{L}(t)$:

$$\|g_k - \mathbf{1}_{\mathcal{C}_k}\|_\infty \leq \|g_k - \mathbf{1}_{\mathcal{C}_k}\|_W < C_0/2 \quad \text{for } k = 1, \ldots, \ell. \tag{34}$$

Now, fix h in $(d_{min}; h_{max})$. We follow the arguments leading to Theorem 7. The convergence in (33) becomes

$$\lim_{n\to\infty} \vartheta_{n,k,j} = g_k(X_j) \quad \text{almost surely.}$$

Hence, there exists n_0 such that, for all $n \geq n_0$, $j \in J(n)$ and $k \in \{1, \ldots, \ell\}$, we have $|\vartheta_{n,k,j} - g_k(X_j)| < C_0/2$. With the triangular inequality and (34), we obtain $|\vartheta_{n,k,j} - \mathbf{1}_{\mathcal{C}_k}(X_j)| < C_0$, that is, the representation of the extracted data set in the feature space concentrates in cubes with edge length $2C_0$, centered at e_k, $k = 1, \ldots, \ell$, up to a linear transformation of $\mathbb{R}^\ell$, for all n large enough. Moreover, if X_j with $j \in J(\infty)$ lies in $\mathcal{C}_{k(j)}$, then its representation is in the cube centered at $e_{k(j)}$. Since those cubes have edge length $2C_0 < 1$, they do not overlap. Hence, a classical method such as the k-means algorithm will asymptotically partition the extracted data set as desired. ■

4.4 Generalizations and Open Problems

Our results allow to relate the limit partition of a spectral clustering algorithm with the connected components of either the support of the distribution (case $t = 0$) or of an upper level set of the density (case $t > 0$). This holds for a fixed similarity function with compact support. Interestingly, the scale parameter h of the similarity function may be larger than the minimal distance between two connected components, up to a threshold value h_{max} above which we have no theoretical guarantee that the connected components will be recovered.

Several important questions, though, remain largely open. Among these, interpreting the limit partition of the classical spectral clustering algorithm with the underlying distribution when one asks for more groups than the number of connected components of its support remains largely an unsolved problem. Also in practice, a sequence h_n decreasing to 0 with the number of observations is frequently used for the scale parameter of the similarity function, and to the best of our knowledge, no convergence results have been obtained yet. At last, it would be interesting to alleviate the assumption of compact support on the similarity function. Indeed, a gaussian kernel is a popular choice in practice. In this direction, one possibility would be to consider a sequence of functions with compact support converging towards the gaussian kernel at an appropriate rate.

5. Auxiliary Results for the Operator Norm Convergence

In this section we give technical lemmas that were needed in the proof of our main results. We also recall several facts from empirical process theory in Section 5.2.

5.1 Preliminaries

Let us start with the following simple lemma.

Lemma 11 *Let $\{A_n\}_{n\geq 0}$ be a decreasing sequence of Borel sets in $\mathbb{R}^d$, with limit $A_\infty = \cap_{n\geq 0} A_n$. If $\mu(A_\infty) = 0$, then*

$$\mathbb{P}_n A_n = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{X_i \in A_n\} \to 0 \quad \textit{almost surely as } n \to \infty,$$

where $\mathbb{P}_n$ is the empirical measure associated with the random sample $X_1, \ldots, X_n$.

Proof First, note that $\lim_n \mu(A_n) = \mu(A_\infty)$. Next, fix an integer k. For all $n \geq k$, $A_n \subset A_k$ and so $\mathbb{P}_n A_n \leq \mathbb{P}_n A_k$. But $\lim_n \mathbb{P}_n A_k = \mu(A_k)$ almost surely by the law of large numbers. Consequently $\limsup_n \mathbb{P}_n A_n \leq \mu(A_k)$ almost surely. Letting $k \to \infty$ yields

$$\limsup_n \mathbb{P}_n A_n \leq \mu(A_\infty) = 0,$$

which concludes the proof since $\mathbb{P}_n A_n \geq 0$. ■

5.2 Uniform Laws of Large Number and Glivenko-Cantelli Classes

In this paragraph, we prove that some classes of functions satisfy a uniform law of large numbers. We shall use some facts on empirical processes that we briefly summarize below. For materials on the subject, we refer the reader to Chapter 19 in van der Vaart (1998) and the book by van der Vaart and Wellner (2000).

A collection $\mathcal{F}$ of functions is Glivenko-Cantelli if it satisfies a uniform law of large numbers, that is, if

$$\sup_{g \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^{n} g(X_i) - \mathbb{E}[X] \right| \to 0 \quad \text{almost surely,}$$

where $(X_n)_n$ is an i.i.d. sequence of random variables with the same distribution as the random variable X. That a class $\mathcal{F}$ is Glivenko-Cantelli depends on its size. A simple way of measuring the size of $\mathcal{F}$ is in terms of bracketing numbers.

A *bracket* $[f_l, f_u]$ is the set of functions g in $\mathcal{F}$ such that $f_u \leq g \leq f_u$, and an ε-*bracket in* L^p is a bracket $[f_l, f_u]$ such that $\mathbb{E}[(f_u(X) - f_l(X))^p]^{1/p} < \varepsilon$. The *bracketing number* $N_{[]}(\varepsilon, \mathcal{F}, L^p)$ is the minimal number of ε-brackets of size ε in the L^p norm which are needed to cover $\mathcal{F}$. A sufficient condition for a class $\mathcal{F}$ to be Glivenko-Cantelli is that $N_{[]}(\varepsilon, \mathcal{F}, L^1)$ is finite for all $\varepsilon > 0$ (Theorem 2.4.1, van der Vaart and Wellner, 2000, p. 122).

A bound on the L^1-bracketing number of a class $\mathcal{F}$ may be obtained from a bound on its metric entropy in the uniform norm, if appropriate. An ε-*covering of* $\mathcal{F}$ in the supremum norm is a collection of N balls of radius ε and centered at functions $f_1, \ldots, f_N$ in $\mathcal{F}$ whose union covers $\mathcal{F}$. For ease of notation, an ε-covering of $\mathcal{F}$ is denoted by the centers of the balls $f_1 \ldots, f_N$. The minimal number $\mathcal{N}(\varepsilon, \mathcal{F}, \|.\|_\infty)$ of balls of radius ε in the supremum norm that are needed to cover $\mathcal{F}$ is called the *covering number* of $\mathcal{F}$ in the uniform norm. The *entropy* of the class is the logarithm of the covering number, and $\mathcal{F}$ is said to have *finite entropy* if $\mathcal{N}(\varepsilon, \mathcal{F}, \|.\|_\infty)$ is finite for all ε. If a class $\mathcal{F}$ may be covered by finitely many balls of radius ε in the supremum norm and centered at $f_1, \ldots, f_N$, then the brackets $[f_i - \varepsilon; f_i + \varepsilon]$ have size at most 2ε for the L^1 norm and their union covers $\mathcal{F}$. This argument is used to conclude the proof of Lemma 13 below.

Lemma 12 *The two collections of functions*

$$\mathcal{F}_1 := \left\{ y \mapsto k_h(y-x)\mathbf{1}_{\mathcal{L}(t)}(y) : x \in \mathcal{L}(t - \varepsilon_0) \right\},$$
$$\mathcal{F}_2 := \left\{ y \mapsto D_x k_h(y-x)\mathbf{1}_{\mathcal{L}(t)}(y) : x \in \mathcal{L}(t - \varepsilon_0) \right\},$$

are Glivenko-Cantelli, where $D_x k_h$ denotes the differential of k_h.

Proof Denote by g_x the functions in $\mathcal{F}_1$, for x ranging in $\mathcal{L}(t-\varepsilon_0)$. We proceed by constructing a covering of $\mathcal{F}_1$ by finitely many L^1-brackets of an arbitrary size, as in, for example, Example 19.8 in van der Vaart (1998). Denote by Q a probability measure on $\mathcal{L}(t)$. Let $\delta > 0$. Since $\mathcal{L}(t-\varepsilon_0)$ is compact, it can be covered by finitely many balls of radius δ, that is, there exists an integer N and points $x_1,\ldots,x_N$ in $\mathcal{L}(t-\varepsilon_0)$ such that $\mathcal{L}(t-\varepsilon_0) \subset \bigcup_{i=1}^N B(x_i,\delta)$. Define the functions $g^l_{i,\delta}$ and $g^u_{i,\delta}$ respectively by

$$g^l_{i,\delta}(y) = \inf_{x\in B(x_i,\delta)} g_x(y) \quad \text{and} \quad g^u_{i,\delta}(y) = \sup_{x\in B(x_i,\delta)} g_x(y).$$

Then the union of brackets $[g^l_{i,\delta}, g^u_{i,\delta}]$, for $i = 1,\ldots,N$, covers $\mathcal{F}_1$. Observe that $|g_x(y)| \le \|k_h\|_\infty$ for all $x \in \mathcal{L}(t-\varepsilon_0)$ and all $y \in \mathcal{L}(t)$ since k_h is uniformly bounded, and that for any fixed $y \in \mathcal{L}(t)$, the map $x \mapsto g_x(y)$ is continuous since k is of class $\mathcal{C}^2$ on $\mathbb{R}^d$ under Assumption 2. Therefore the function $g^u_{i,\delta} - g^l_{i,\delta}$ converges pointwise to 0 and $\|g^u_{i,\delta} - g^l_{i,\delta}\|_{L^1(Q)}$ goes to 0 as $\delta \to 0$ by the Lebesgue dominated convergence theorem. Consequently, for any $\varepsilon > 0$, one may choose a finite covering of $\mathcal{L}(t-\varepsilon_0)$ by N balls of radius $\delta > 0$ such that $\max_{i=1,\ldots,N} \|g^u_{i,\delta} - g^l_{i,\delta}\|_{L^1(Q)} \le \varepsilon$. Hence, for all $\varepsilon > 0$ the L^1-bracketing number of $\mathcal{F}_1$ is finite, so $\mathcal{F}_1$ is Glivenko-Cantelli. Since k_h is continuously differentiable, the same arguments apply to each component of $D_x k_h$, and so $\mathcal{F}_2$ is also a Glivenko-Cantelli class. ■

Lemma 13 *Let $r : \mathcal{L}(t) \times \mathbb{R}^d$ be a continuously differentiable function such that*
(i) there exists a compact $\mathcal{K} \subset \mathbb{R}^d$ such that $r(x,y) = 0$ for all $(x,y) \in \mathcal{L}(t) \times \mathcal{K}^c$;
(ii) r is uniformly bounded on $\mathcal{L}(t) \times \mathbb{R}^d$, that is, $\|r\|_\infty < \infty$.
Then the collection of functions

$$\mathcal{F}_3 := \left\{ y \mapsto r(x,y)g(y)\mathbf{1}_{\mathcal{L}(t)}(y) : x \in \mathcal{L}(t),\, \|g\|_{W(\mathcal{L}(t))} \le 1 \right\}$$

is Glivenko-Cantelli.

Proof Set $\mathcal{R} = \{y \mapsto r(x,y) : x \in \mathcal{L}(t)\}$. Since r is continuous on the compact set $\mathcal{L}(t) \times \mathcal{K}$, it is uniformly continuous. So for any $\varepsilon > 0$, there exists $\delta > 0$ such that $|r(x,y) - r(x',y')| \le \varepsilon$ whenever the points (x,y) and (x',y') in $\mathcal{L}(t) \times \mathcal{K}$ are at a distance no more than δ. Since $\mathcal{L}(t)$ is compact, it may be covered by finitely many balls of radius δ centered at N points $x_1,\ldots,x_N$ of $\mathcal{L}(t)$. Denote by g_i the function in $\mathcal{R}$ defined by $g_i(y) = r(x_i,y)$, and let $\mathcal{R}_i = \{y \mapsto r(x,y) : x \in \mathcal{L}(t)\,, \|x - x_i\| \le \delta\}$. Then the union of the $\mathcal{R}_i$'s cover $\mathcal{R}$, and for any g in $\mathcal{R}_i$, $\|g - g_i\|_\infty \le \varepsilon$. This shows that $\mathcal{R}$ has finite entropy in the supremum norm, that is, that $\mathcal{N}(\varepsilon, \mathcal{R}, \|.\|_\infty) < \infty$.

Second, consider the unit ball $\mathcal{G}$ in $W(\mathcal{L}(t))$, that is, $\mathcal{G} = \{g : \mathcal{L}(t) \to \mathbb{C} : \|g\|_{W(\mathcal{L}(t))} \le 1\}$. Denote by $\mathcal{X}$ the convex hull of $\mathcal{L}(t)$, and consider the collection of functions $\tilde{\mathcal{G}} = \{\tilde{g} : \mathcal{X} \to \mathbb{C} : \|\tilde{g}\|_{W(\mathcal{X})} \le 1\}$. Observe that $\tilde{\mathcal{G}}$ is a subset of the Holder space $C^{0,1}(\mathcal{X})$. It is proved in Theorem 2.7.1, p. 155 in the book by van der Vaart and Wellner (2000) that if $\mathcal{X}$ is a convex bounded subset of $\mathbb{R}^d$, then $C^{0,1}(\mathcal{X})$ has finite entropy in the uniform norm (this theorem was established in van der Vaart (1994) using results of Kolmogorov and Tikhomirov (1961). Consequently, for any $\varepsilon > 0$, there exist N functions $\tilde{g}_1,\ldots,\tilde{g}_n$ in $\tilde{\mathcal{G}}$ such that the union of the sets $\{\tilde{g} \in \tilde{\mathcal{G}} : \|\tilde{g} - \tilde{g}_i\|_\infty \le \varepsilon\}$ covers $\tilde{\mathcal{G}}$. By considering the restrictions g_i of each $\tilde{g}_i$ to $\mathcal{L}$, it follows that the union of the sets $\{g \in \mathcal{G} : \|g - g_i\|_\infty \le \varepsilon\}$ covers $\mathcal{G}$. So $\mathcal{N}(\varepsilon, \mathcal{G}, \|.\|_\infty) < \infty$ for any $\varepsilon > 0$.

Now fix $\varepsilon > 0$. Let $r_1, \ldots, r_M \in \mathcal{R}$ be an ε-covering of $\mathcal{R}$ in the supremum norm, and let $g_1, \ldots, g_N \in \mathcal{G}$ be an ε-covering of $\mathcal{G}$ in the supremum norm, for some integers M and N. For any function f in $\mathcal{F}_3$ of the form $f(y) = r(x,y)g(y)\mathbf{1}_{\mathcal{L}(t)}$ for some $x \in \mathcal{L}(t)$ and $g \in W(\mathcal{L}(t))$ with $\|g\|_{W(\mathcal{L}(t))} \leq 1$, there exists $1 \leq i \leq M$ and $1 \leq j \leq N$ such that $\|r(x,.) - r_i\|_\infty \leq \varepsilon$ and $\|g - g_j\|_\infty \leq \varepsilon$. Then

$$\begin{aligned}
\sup_{y \in \mathbb{R}^d} |f(y) - r_i(y)g_j(y)\mathbf{1}_{\mathcal{L}(t)}(y)| &= \sup_{y \in \mathcal{L}(t)} |r(x,y)g(y) - r_i(y)g_j(y)| \\
&= \sup_{y \in \mathcal{L}(t)} \big|(r(x,y) - r_i(y))g(y) + r_i(y)(g(y) - g_j(y))\big| \\
&\leq \sup_{y \in \mathcal{L}(t)} |r(x,y) - r_i(y)|\, \|g\|_\infty + \|r_i\|_\infty \sup_{y \in \mathcal{L}(t)} \big|g(y) - g_j(y)\big| \\
&\leq \varepsilon + \|r\|_\infty \varepsilon,
\end{aligned}$$

since $\|r_i\|_\infty = 1$ for all $i = 1, \ldots, M$ and since $\|g\|_\infty \leq \varepsilon$. So the collection of functions $f_{ij} : y \mapsto r_i(y)g_j(y)\mathbf{1}_{\mathcal{L}(t)}(y)$ form a finite covering of $\mathcal{F}_3$ of size $M \times N$ by balls of radius $(1 + \|r\|_\infty)\varepsilon$ in the supremum norm, and $\mathcal{N}(\varepsilon, \mathcal{F}_3, \|.\|_\infty) < \infty$ for all $\varepsilon > 0$.

To conclude the proof, observe that if $f_1, \ldots, f_N \in \mathcal{F}_3$ is an ε-covering of $\mathcal{F}_3$ in the supremum norm, then the brackets $[f_i - \varepsilon; f_i + \varepsilon]$ have size at most 2ε in the L^1 norm, and their union covers $\mathcal{F}_3$. So for all $\varepsilon > 0$ the L^1-bracketing number of $\mathcal{F}_3$ is finite and $\mathcal{F}_3$ is Glivenko-Cantelli. ■

5.3 Bounds on Kernels

We recall that the limit operator Q_h is given by (18). The following lemma gives useful bounds on K_h and q_h, both defined in (19).

Lemma 14 *1. The function K_h is uniformly bounded from below by some positive number on $\mathcal{L}(t - \varepsilon_0)$, that is,* $\inf\{K_h(x) : x \in \mathcal{L}(t - \varepsilon_0)\} > 0$*;*
2. The kernel q_h is uniformly bounded, that is, $\|q_h\|_\infty < \infty$*;*
3. The differential of q_h with respect to x is uniformly bounded on $\mathcal{L}(t - \varepsilon_0) \times \mathbb{R}^d$, that is, $\sup\big\{\|D_x q_h(x,y)\| : (x,y) \in \mathcal{L}(t - \varepsilon_0) \times \mathbb{R}^d\big\} < \infty$*;*
4. The Hessian of q_h with respect to x is uniformly bounded on $\mathcal{L}(t - \varepsilon_0) \times \mathbb{R}^d$, that is, $\sup\big\{\|D_x^2 q_h(x,y)\| : (x,y) \in \mathcal{L}(t - \varepsilon_0) \times \mathbb{R}^d\big\} < \infty$.

Proof First observe that the statements 2, 3 and 4 are immediate consequences of statement 1 together with the fact that the function k_h is of class $\mathcal{C}^2$ with compact support, which implies that $k_h(y - x)$, $D_x k_h(y - x)$, and $D_x^2 k_h(y - x)$ are uniformly bounded.

To prove statement 1, note that K_h is continuous and that $K_h(x) > 0$ for all $x \in \mathcal{L}(t)$. Set

$$\alpha(\varepsilon_0) = \inf\big\{\|D_x f(x)\|; x \in \mathcal{L}^t_{t - \varepsilon_0}\big\}.$$

Let $(x,y) \in \mathcal{L}^t_{t-\varepsilon_0} \times \partial\mathcal{L}(t)$. Then

$$\varepsilon_0 \geq f(y) - f(x) \geq \alpha(\varepsilon_0)\|y - x\|.$$

Thus, $\|y - x\| \leq \varepsilon_0/\alpha(\varepsilon_0)$ and so

$$\operatorname{dist}\big(x, \mathcal{L}(t)\big) \leq \frac{\varepsilon_0}{\alpha(\varepsilon_0)}, \quad \text{for all } x \in \mathcal{L}^t_{t-\varepsilon_0}.$$

Recall from (4) that $h/2 > \varepsilon_0/\alpha(\varepsilon_0)$. Consequently, for all $x \in \mathcal{L}(t-\varepsilon_0)$, the set $(x+hB/2)\cap\mathcal{L}(t)$ contains a non-empty, open set $U(x)$. Moreover k_h is bounded from below by some positive number on $hB/2$ by Assumption 2. Hence $K_h(x) > 0$ for all x in $\mathcal{L}(t-\varepsilon_0)$ and point 1 follows from the continuity of K_h and the compactness of $\mathcal{L}(t-\varepsilon_0)$. ■

In order to prove the convergence of $\widehat{Q}_{n,h}$ to Q_h, we also need to study the uniform convergence of $K_{n,h}$, given in (2). Lemma 15 controls the difference between $K_{n,h}$ and K_h, while Lemma 16 controls the ratio of K_h over $K_{n,h}$.

Lemma 15 *As $n \to \infty$, almost surely,*

1. $\sup_{x\in\mathcal{L}(t-\varepsilon_0)} \left|K_{n,h}(x) - K_h(x)\right| \to 0$ *and*
2. $\sup_{x\in\mathcal{L}(t-\varepsilon_0)} \left|D_x K_{n,h}(x) - D_x K_h(x)\right| \to 0.$

Proof Let

$$K^{\dagger}_{n,h}(x) := \frac{1}{n\mu(\mathcal{L}(t))}\sum_{i=1}^{n} k_h(X_i - x)\mathbf{1}_{\mathcal{L}_n(t)}(X_i), \quad K^{\dagger\dagger}_{n,h}(x) := \frac{1}{n\mu(\mathcal{L}(t))}\sum_{i=1}^{n} k_h(X_i - x)\mathbf{1}_{\mathcal{L}(t)}(X_i).$$

Let us start with the inequality

$$\left|K_{n,h}(x) - K_h(x)\right| \le \left|K_{n,h}(x) - K^{\dagger}_{n,h}(x)\right| + \left|K^{\dagger}_{n,h}(x) - K_h(x)\right|, \tag{35}$$

for all $x \in \mathcal{L}(t-\varepsilon_0)$. Using the inequality

$$\left|K_{n,h}(x) - K^{\dagger}_{n,h}(x)\right| \le \left|\frac{n}{j(n)} - \frac{1}{\mu(\mathcal{L}(t))}\right| \|k_h\|_\infty$$

we conclude that the first term in (35) tends to 0 uniformly in x over $\mathcal{L}(t-\varepsilon_0)$ with probability one as $n \to \infty$, since $j(n)/n \to \mu\big(\mathcal{L}(t)\big)$ almost surely, and since k_h is bounded on $\mathbb{R}^d$.

Next, for all $x \in \mathcal{L}(t-\varepsilon_0)$, we have

$$\left|K^{\dagger}_{n,h}(x) - K_h(x)\right| \le \left|K^{\dagger}_{n,h}(x) - K^{\dagger\dagger}_{n,h}(x)\right| + \left|K^{\dagger\dagger}_{n,h}(x) - K_h(x)\right|. \tag{36}$$

The first term in (36) is bounded by

$$\begin{aligned}\left|K^{\dagger}_{n,h}(x) - K^{\dagger\dagger}_{n,h}(x)\right| &\le \frac{\|k_h\|_\infty}{\mu\big(\mathcal{L}(t)\big)}\frac{1}{n}\left|\sum_{i=1}^{n}\left\{\mathbf{1}_{\mathcal{L}_n(t)}(X_i) - \mathbf{1}_{\mathcal{L}(t)}(X_i)\right\}\right| \\ &= \frac{\|k_h\|_\infty}{\mu\big(\mathcal{L}(t)\big)}\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{\mathcal{L}_n(t)\Delta\mathcal{L}(t)}(X_i),\end{aligned}$$

where $\mathcal{L}_n(t)\Delta\mathcal{L}(t)$ denotes the symmetric difference between $\mathcal{L}_n(t)$ and $\mathcal{L}(t)$. Recall that, on the event Ω_n, $\mathcal{L}(t-\varepsilon_n) \subset \mathcal{L}_n(t) \subset \mathcal{L}(t-\varepsilon_n)$. Therefore $\mathcal{L}_n(t)\Delta\mathcal{L}(t) \subset \mathcal{L}_{t-\varepsilon_n}^{t+\varepsilon_n}$ on Ω_n, and so

$$0 \le \frac{1}{n}\left|\sum_{i=1}^{n}\left\{\mathbf{1}_{\mathcal{L}_n(t)}(X_i) - \mathbf{1}_{\mathcal{L}(t)}(X_i)\right\}\right|\mathbf{1}_{\Omega_n} \le \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{A_n}(X_i),$$

where $A_n = \mathcal{L}_{t-\varepsilon_n}^{t+\varepsilon_n}$. Hence by Lemma 11, and since $\mathbf{1}_{\Omega_n} \to 1$ almost surely as $n \to \infty$, the first term in (36) converges to 0 with probability one as $n \to \infty$.

Next, since the collection $\{y \mapsto k_h(y-x)\mathbf{1}_{\mathcal{L}(t)}(y) : x \in \mathcal{L}(t-\varepsilon_0)\}$ is Glivenko-Cantelli by Lemma 12, we conclude that

$$\sup_{x \in \mathcal{L}(t-\varepsilon_0)} \left| K_{n,h}^{\dagger\dagger}(x) - K_h(x) \right| \to 0,$$

with probability one as $n \to \infty$. This concludes the proof of the first statement.

The second statement may be proved by developing similar arguments, with k_h replaced by $D_x k_h$, and by noting that the collection of functions $\{y \mapsto D_x k_h(y-x)\mathbf{1}_{\mathcal{L}(t)}(y) : x \in \mathcal{L}(t-\varepsilon_0)\}$ is also Glivenko-Cantelli by Lemma 12. ■

Lemma 16 *As $n \to \infty$, almost surely,*

$$\sup_{x \in \mathcal{L}(t)} \left| \frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} - 1 \right| \to 0, \qquad \textit{and} \qquad \sup_{x \in \mathcal{L}(t)} \left\| D_x \left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} \right] \right\| \to 0.$$

Proof First of all, K_h is uniformly continuous on $\mathcal{L}(t-\varepsilon_0)$ since K_h is continuous and since $\mathcal{L}(t-\varepsilon_0)$ is compact. Moreover, φ_n converges uniformly to the identity map of $\mathcal{L}(t)$ by Lemma 17. Hence

$$\sup_{x \in \mathcal{L}(t)} \left| K_h\big(\varphi_n(x)\big) - K_h(x) \right| \to 0 \quad \text{as } n \to \infty,$$

and since $K_{n,h}$ converges uniformly to K_h with probability one as $n \to \infty$ by Lemma 15, this proves the first convergence result.

We have

$$D_x \left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} \right] = \frac{D_x\varphi_n(x)\Big[K_{n,h}\big(\varphi_n(x)\big)D_xK_h\big(\varphi_n(x)\big) - K_h\big(\varphi_n(x)\big)D_xK_{n,h}\big(\varphi_n(x)\big)\Big]}{\Big[K_{n,h}\big(\varphi_n(x)\big)\Big]^2}.$$

Since $D_x\varphi_n(x)$ converges to the identity matrix I_d uniformly over $x \in \mathcal{L}(t)$ by Lemma 17, $\|D_x\varphi_n(x)\|$ is bounded uniformly over n and $x \in \mathcal{L}(t)$ by some positive constant C_φ. Furthermore the map $x \mapsto K_{n,h}(x)$ is bounded from below over $\mathcal{L}(t)$ by some positive constant k_{min} independent of x because i) $\inf_{x \in \mathcal{L}(t-\varepsilon_0)} K_h(x) > 0$ by Lemma 14, and ii) $\sup_{x \in \mathcal{L}(t-\varepsilon_0)} \left| K_{n,h}(x) - K_h(x) \right| \to 0$ by Lemma 15. Hence

$$\left| D_x \left[\frac{K_h\big(\varphi_n(x)\big)}{K_{n,h}\big(\varphi_n(x)\big)} \right] \right| \leq \frac{C_\varphi}{k_{min}^2} \left| K_{n,h}(y)D_xK_h(y) - K_h(y)D_xK_{n,h}(y) \right|,$$

where we have set $y = \varphi_n(x)$ which belongs to $\mathcal{L}(t-\varepsilon_n) \subset \mathcal{L}(t-\varepsilon_0)$. At last, Lemma 15 gives

$$\sup_{y \in \mathcal{L}(t-\varepsilon_0)} \left| K_{n,h}(y)D_xK_h(y) - K_h(y)D_xK_{n,h}(y) \right| \to 0 \quad \text{almost surely,}$$

as $n \to \infty$ which proves the second convergence result. ■

Acknowledgments

This work was supported by the French National Research Agency (ANR) under grant ANR-09-BLAN-0051-01. We thank the referees and the Associate Editor for valuable comments and insightful suggestions that led to an improved version of the paper.

Appendix A. Geometry of Level Sets

The proof of the following result is adapted from Theorem 3.1 in (Milnor, 1963, p. 12) and Theorem 5.2.1 in (Jost, 1995, p. 176)

Lemma 17 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a function of class $\mathcal{C}^2$. Let $t \in \mathbb{R}$ and suppose that there exists $\varepsilon_0 > 0$ such that $f^{-1}\big([t-\varepsilon_0; t+\varepsilon_0]\big)$ is non empty, compact and contains no critical point of f. Let $\{\varepsilon_n\}_n$ be a sequence of positive numbers such that $\varepsilon_n < \varepsilon_0$ for all n, and $\varepsilon_n \to 0$ as $n \to \infty$. Then there exists a sequence of diffeomorphisms $\varphi_n : \mathcal{L}(t) \to \mathcal{L}(t-\varepsilon_n)$ carrying $\mathcal{L}(t)$ to $\mathcal{L}(t-\varepsilon_n)$ such that:*
1. $\sup_{x \in \mathcal{L}(t)} \|\varphi_n(x) - x\| \to 0$ *and*
2. $\sup_{x \in \mathcal{L}(t)} \|D_x\varphi_n(x) - I_d\| \to 0$,
as $n \to \infty$, where $D_x\varphi_n$ denotes the differential of φ_n and where I_d is the identity matrix on $\mathbb{R}^d$.

Proof Recall first that a one-parameter group of diffeomorphisms $\{\varphi_u\}_{u\in\mathbb{R}}$ of $\mathbb{R}^d$ gives rise to a vector field V defined by

$$V_x g = \lim_{u\to 0} \frac{g\big(\varphi_u(x)\big) - g(x)}{u}, \quad x \in \mathbb{R}^d,$$

for all smooth function $g : \mathbb{R}^d \to \mathbb{R}$. Conversely, a smooth vector field which vanishes outside of a compact set generates a unique one-parameter group of diffeomorphisms of $\mathbb{R}^d$; see Lemma 2.4 in (Milnor, 1963, p. 10) and Theorem 1.6.2 in (Jost, 1995, p. 42)

Denote the set $\{x \in \mathbb{R}^d : a \le f(x) \le b\}$ by $\mathcal{L}_a^b$, for $a \le b$. Let $\eta : \mathbb{R}^d \to \mathbb{R}$ be the non-negative differentiable function with compact support defined by

$$\eta(x) = \begin{cases} 1/\|D_x f(x)\|^2 & \text{if } x \in \mathcal{L}_{t-\varepsilon_0}^t, \\ (t+\varepsilon_0 - f(x))/\|D_x f(x)\|^2 & \text{if } x \in \mathcal{L}_t^{t+\varepsilon_0}, \\ 0 & \text{otherwise.} \end{cases}$$

Then the vector field V defined by $V_x = \eta(x) D_x f(x)$ has compact support $\mathcal{L}_{t-\varepsilon_0}^{t+\varepsilon_0}$, so that V generates a one-parameter group of diffeomorphisms

$$\varphi_u : \mathbb{R}^d \to \mathbb{R}^d, \quad u \in \mathbb{R}.$$

We have

$$D_u\big[f\big(\varphi_u(x)\big)\big] = \langle V, D_x f\rangle_{\varphi_u(x)} \ge 0,$$

since η is non-negative. Furthermore,

$$\langle V, D_x f\rangle_{\varphi_u(x)} = 1, \quad \text{if } \varphi_u(x) \in \mathcal{L}_{t-\varepsilon_0}^t$$

Consequently the map $u \mapsto f\big(\varphi_u(x)\big)$ has constant derivative 1 as long as $\varphi_u(x)$ lies in $\mathcal{L}^t_{t-\varepsilon_0}$. This proves the existence of the diffeomorphism $\varphi_n := \varphi_{-\varepsilon_n}$ which carries $\mathcal{L}(t)$ to $\mathcal{L}(t-\varepsilon_n)$.

Note that the map $u \in \mathbb{R} \mapsto \varphi_u(x)$ is the integral curve of V with initial condition x. Without loss of generality, suppose that $\varepsilon_n \leq 1$. For all x in $\mathcal{L}^{t+\varepsilon_0}_{t-\varepsilon_0}$, we have

$$\|\varphi_n(x) - x\| \leq \int_{-\varepsilon_n}^{0} \big\|D_u\big(\varphi_u(x)\big)\big\|\, du \leq \varepsilon_n/\beta(\varepsilon_n) \leq \varepsilon_n/\beta(\varepsilon_0)$$

where we have set

$$\beta(\varepsilon) := \inf\big\{\, \|D_x f(x)\| \,:\, x \in \mathcal{L}^{t+\varepsilon}_{t-\varepsilon} \big\} > 0.$$

This proves the statement 1, since $\varphi_n(x) - x$ is identically 0 on $\mathcal{L}(t+\varepsilon_0)$.

For the statement 2, observe that $\varphi_u(x)$ satisfies the relation

$$\varphi_u(x) - x = \int_0^u D_v\big(\varphi_v(x)\big)dv = \int_0^u V\big(\varphi_v(x)\big)\big)dv.$$

Differentiating with respect to x yields

$$D_x\varphi_u(x) - I_d = \int_0^u D_x\varphi_v(x) \circ D_x V\big(\varphi_v(x)\big)dv.$$

Since f is of class $\mathcal{C}^2$, the two terms inside the integral are uniformly bounded over $\mathcal{L}^{t+\varepsilon_0}_{t-\varepsilon_0}$, so that there exists a constant $C > 0$ such that

$$\|D_x\varphi_n - I\|_x \leq C\varepsilon_n,$$

for all x in $\mathcal{L}^{t+\varepsilon_0}_{t-\varepsilon_0}$. Since $\|D_x\varphi_n - I\|_x$ is identically zero on $\mathcal{L}(t+\varepsilon_0)$, this proves the statement 2. ■

Appendix B. Continuity of an Isolated Finite Set of Eigenvalues

In brief, the spectrum $\sigma(T)$ of a bounded linear operator T on a Banach space is upper semi-continuous in T, but not lower semi-continuous; see Kato (1995), IV§3.1 and IV§3.2. However, an isolated finite set of eigenvalues of T is continuous in T, as stated in Theorem 19 below.

Let T be a bounded operator on the $\mathbb{C}$-Banach space E with spectrum $\sigma(T)$. Let $\sigma_1(T)$ be a finite set of eigenvalues of T. Set $\sigma_2(T) = \sigma(T) \setminus \sigma_1(T)$ and suppose that $\sigma_1(T)$ is separated from $\sigma_2(T)$ by a rectifiable, simple, and closed curve Γ. Assume that a neighborhood of $\sigma_1(T)$ is enclosed in the interior of Γ. Then we have the following theorem; see Kato (1995), III.§6.4 and III.§6.5.

Theorem 18 (Separation of the spectrum) *The Banach space E decomposes into a pair of supplementary subspaces as $E = M_1 \oplus M_2$ such that T maps M_j into M_j ($j = 1,2$) and the spectrum of the operator induced by T on M_j is $\sigma_j(T)$ ($j = 1,2$). If additionally the total multiplicity m of $\sigma_1(T)$ is finite, then* $\dim(M_1) = m$.

Moreover, the following theorem states that a finite system of eigenvalues of T, as well as the decomposition of E of Theorem 18, depends continuously of T, see Kato (1995), IV.§3.5. Let $\{T_n\}_n$ be a sequence of operators which converges to T in norm. Denote by $\sigma_1(T_n)$ the part of the spectrum of T_n enclosed in the interior of the closed curve Γ, and by $\sigma_2(T_n)$ the remainder of the spectrum of T_n.

Theorem 19 (Continuous approximation of the spectral decomposition) *There exists a finite integer n_0 such that the following holds true.*
1. Both $\sigma_1(T_n)$ and $\sigma_2(T_n)$ are nonempty for all $n \geq n_0$ provided this is true for T.
2. For each $n \geq 0$, the Banach space E decomposes into two subspaces as $E = M_{n,1} \oplus M_{n,2}$ in the manner of Theorem 18, that is, T_n maps $M_{n,j}$ into itself and the spectrum of T_n on $M_{n,j}$ is $\sigma_j(T_n)$.
3. For all $n \geq n_0$, $M_{n,j}$ is isomorphic to M_j.
4. If $\sigma_1(T)$ is a singleton $\{\lambda\}$, then every sequence $\{\lambda_n\}_n$ with $\lambda_n \in \sigma_1(T_n)$ for all $n \geq n_0$ converges to λ.
5. If Π is the projector on M_1 along M_2 and Π_n the projector on $M_{n,1}$ along $M_{n,2}$, then Π_n converges in norm to Π.
6. If the total multiplicity m of $\sigma_1(T)$ is finite, then, for all $n \geq n_0$, the total multiplicity of $\sigma_1(T_n)$ is also m and $\dim(M_{n,1}) = m$.

Appendix C. Background Materials on Markov Chains

For the reader not familiar with Markov chains on a general state space, we begin by summarizing the relevant part of the theory.

Let $\{\xi_i\}_{i\geq 0}$ be a Markov chain with state space $\mathcal{S} \subset \mathbb{R}^d$ and transition kernel $q(x,dy)$. We write P_x for the probability measure when the initial state is x and E_x for the expectation with respect to P_x. The Markov chain is called *(strongly) Feller* if the map

$$x \in \mathcal{S} \mapsto Qg(x) := \int_{\mathcal{S}} q(x,dy)g(y) = \mathbb{E}_x f(\xi_1)$$

is continuous for every bounded, measurable function g on $\mathcal{S}$; see (Meyn and Tweedie, 1993, p. 132). This condition ensures that the chain behaves nicely with the topology of the state space $\mathcal{S}$. The notion of irreducibility expresses the idea that, from an arbitrary initial point, each subset of the state space may be reached by the Markov chain with a positive probability. A Feller chain is said *open set irreducible* if, for every points x, y in $\mathcal{S}$, and every $\eta > 0$,

$$\sum_{n\geq 1} q^n(x, y+\eta B) > 0,$$

where $q^n(x,dy)$ stands for the n-step transition kernel; see (Meyn and Tweedie, 1993, p. 135). Even if open set irreducible, a Markov chain may exhibit a periodic behavior, that is, there may exist a partition $\mathcal{S} = \mathcal{S}_0 \cup \mathcal{S}_1 \cup \ldots \cup \mathcal{S}_N$ of the state space such that, for every initial state $x \in \mathcal{S}_0$,

$$P_x[\xi_1 \in \mathcal{S}_1, \xi_2 \in \mathcal{S}_2, \ldots, \xi_N \in \mathcal{S}_N, \xi_{N+1} \in \mathcal{S}_0, \ldots] = 1.$$

Such a behavior does not occur if the Feller chain is *topologically aperiodic*, that is, if for each initial state x, each $\eta > 0$, there exists n_0 such that $q^n(x, x+\eta B) > 0$ for every $n \geq n_0$; see (Meyn and Tweedie, 1993, p. 479).

Next we come to ergodic properties of the Markov chain. A Borel set A of $\mathcal{S}$ is called *Harris recurrent* if the chain visits A infinitely often with probability 1 when started at any point x of A, that is,

$$P_x\left(\sum_{i=0}^{\infty} \mathbf{1}_A(\xi_i) = \infty\right) = 1$$

for all $x \in A$. The chain is then said to be *Harris recurrent* if every Borel set A with positive Lebesgue measure is Harris recurrent; see (Meyn and Tweedie, 1993, p. 204). At least two types of behavior, called evanescence and non-evanescence, may occur. The event $[\xi_n \to \infty]$ denotes the fact that the sample path visits each compact set only finitely many often, and the Markov chain is called *non-evanescent* if $P_x(\xi_n \to \infty) = 0$ for each initial state $x \in \mathcal{S}$. Specifically, a Feller chain is Harris recurrent if and only if it is non-evanescent; see (Meyn and Tweedie, 1993, p. 122), Theorem 9.2.2.

The ergodic properties exposed above describe the long time behavior of the chain. A measure ν on the state space is said *invariant* if

$$\nu(A) = \int_{\mathcal{S}} q(x,A)\nu(dx)$$

for every Borel set A in $\mathcal{S}$. If the chain is Feller, open set irreducible, topologically aperiodic and Harris recurrent, it admits a unique (up to constant multiples) invariant measure ν; see (Meyn and Tweedie, 1993, p. 235), Theorem 10.0.1. In this case, either $\nu(\mathcal{S}) < \infty$ and the chain is called *positive*, or $\nu(\mathcal{S}) = \infty$ and the chain is called *null*. The following important result provides one with the limit of the distribution of ξ_n when $n \to \infty$, whatever the initial state is. Assuming that the chain is Feller, open set irreducible, topologically aperiodic and positive Harris recurrent, the sequence of distribution $\{q^n(x,dy)\}_{n\geq 1}$ converges in total variation to $\nu(dy)$, the unique invariant probability distribution; see Theorem 13.3.1 of (Meyn and Tweedie, 1993, p. 326). That is to say, for every x in $\mathcal{S}$,

$$\sup_g \left\{ \left| \int_{\mathcal{S}} g(y)q^n(x,dy) - \int_{\mathcal{S}} g(y)\nu(dy) \right| \right\} \to 0 \quad \text{as } n \to \infty,$$

where the supremum is taken over all continuous functions g from $\mathcal{S}$ to $\mathbb{R}$ with $\|g\|_\infty \leq 1$.

References

M.R. Anderberg. *Cluster Analysis for Applications*. Academic Press, New-York, 1973.

A. Azzalini and N. Torelli. Clustering via nonparametric estimation. *Stat. Comput.*, 17:71–80, 2007.

M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In *Learning Theory*, volume 3559 of *Lecture Notes in Comput. Sci.*, pages 486–500. Springer, Berlin, 2005.

M. Belkin, I. Matveeva, and P. Niyogi. Regularization and semi-supervised learning on large graphs. In *Learning Theory*, volume 3120 of *Lecture Notes in Comput. Sci.*, pages 624–638. Springer, Berlin, 2004.

G. Biau, B. Cadre, and B. Pelletier. A graph-based estimator of the number of clusters. *ESAIM Probab. Stat.*, 11:272–280, 2007.

F.R.K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 1997.

R.R. Coifman and S. Lafon. Diffusion maps. *Appl. Comput. Harmon. Anal.*, 21:5–30, 2006.

A. Cuevas, M. Febrero, and R. Fraiman. Estimating the number of clusters. *Canadian Journal of Statistics*, 28:367–382, 2000.

A. Cuevas, M. Febrero, and R. Fraiman. Cluster analysis: a further approach based on density estimation. *Comput. Statist. Data Anal.*, 36:441–459, 2001.

R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley Interscience, New-York, 2000.

M. Fillipone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recognition*, 41(1):176–190, 2008.

L.A. García-Escudero, A. Gordaliza, C. Matrán, and A. Mayo-Iscar. A general trimming approach to robust cluster analysis. *Ann. Statis.*, 36(3):1324–1345, 2008.

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.

J.A. Hartigan. *Clustering Algorithms*. Wiley, New-York, 1975.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New-York, 2001.

M. Hein, J.-Y. Audibert, and U. von Luxburg. Graph Laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8:1325–1368, 2007.

J. Jost. *Riemannian Geometry and Geometric Analysis*. Universitext. Springer-Verlag, Berlin, 1995.

T. Kato. *Perturbation Theory for Linear Operators*. Classics in Mathematics. Springer-Verlag, Berlin, 1995. Reprint of the 1980 edition.

A.N. Kolmogorov and V.M. Tikhomirov. ε-entropy and ε-capacity of sets in functional space. *Amer. Math. Soc. Transl. (2)*, 17:277–364, 1961.

V.I. Koltchinskii. Asymptotics of spectral projections of some random matrices approximating integral operators. In *High Dimensional Probability (Oberwolfach, 1996)*, volume 43 of *Progr. Probab.*, pages 191–227. Birkhäuser, Basel, 1998.

T. Linder. Learning-theoretic methods in vector quantization. In *Principles of Nonparametric Learning (Udine, 2001)*, volume 434 of *CISM Courses and Lectures*, pages 163–210. Springer, Vienna, 2002.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkely Symp. Math. Statist. Prob.*, volume 1, pages 281–297, 1967.

G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley, New-York, 2000.

S.P. Meyn and R.L. Tweedie. *Markov Chains and Stochastic Stability*. Communications and Control Engineering Series. Springer-Verlag, London, 1993.

J.W. Milnor. *Morse Theory*. Annals of Mathematics Studies, No. 51. Princeton University Press, Princeton, N.J., 1963.

B. Nadler, S. Lafon, R.R. Coifman, and I.G. Kevrekidis. Difusion maps, spectral clustering and reaction coordinates of dynamical systems. *Appl. Comput. Harmon. Anal.*, 21(1):113–127, 2006.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 849–856. MIT Press, 2002.

D. Pollard. Consistency of k-means clustering. *Ann. Statis.*, 9(1):135–140, 1981.

D. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag. New-York, 1984.

B. L. S. Prakasa Rao. *Nonparametric Functional Estimation*. Probability and Mathematical Statistics. Academic Press Inc., New York, 1983.

L. Rosasco, M. Belkin, and E. De Vito. On learning with integral operators. *Journal of Machine Learning Research*, 11:905–934, 2010.

A.W. van der Vaart. Bracketing smooth functions. *Stochastic Process. Appl.*, 52(1):93–105, 1994.

A.W. van der Vaart. *Asymptotic Statistics*, volume 3 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, Cambridge, 1998.

A.W. van der Vaart and J.A. Wellner. *Weak Convergence and Empirical Processes*. Springer Series in Statistics, Second Edition. Springer, New-York, 2000.

U. von Luxburg. A tutorial on spectral clustering. *Stat. Comput.*, 17(4):395–416, 2007.

U. von Luxburg and S. Ben-David. Towards a statistical theory of clustering. In *PASCAL Workshop on Statistics and Optimization of Clustering*, 2005.

U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *Ann. Statis.*, 36 (2):555–586, 2008.

L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Eighteenth Annual Conference on Neural Information Processing Systems (NIPS)*, 2004.

Journal of Machine Learning Research 12 (2011) 417-454 Submitted 7/10; Revised 10/10; Published 2/11

Approximate Marginals in Latent Gaussian Models

Botond Cseke B.CSEKE@SCIENCE.RU.NL
Tom Heskes T.HESKES@SCIENCE.RU.NL
Institute for Computing and Information Sciences
Faculty of Science, Radboud University Nijmegen
Heyendaalseweg 135, 6525 AJ, The Netherlands

Editor: Manfred Opper

Abstract

We consider the problem of improving the Gaussian approximate posterior marginals computed by expectation propagation and the Laplace method in latent Gaussian models and propose methods that are similar in spirit to the Laplace approximation of Tierney and Kadane (1986). We show that in the case of sparse Gaussian models, the computational complexity of expectation propagation can be made comparable to that of the Laplace method by using a parallel updating scheme. In some cases, expectation propagation gives excellent estimates where the Laplace approximation fails. Inspired by bounds on the correct marginals, we arrive at factorized approximations, which can be applied on top of both expectation propagation and the Laplace method. The factorized approximations can give nearly indistinguishable results from the non-factorized approximations and their computational complexity scales linearly with the number of variables. We experienced that the expectation propagation based marginal approximations we introduce are typically more accurate than the methods of similar complexity proposed by Rue et al. (2009).

Keywords: approximate marginals, Gaussian Markov random fields, Laplace approximation, variational inference, expectation propagation

1. Introduction

Following Rue et al. (2009), we consider the problem of computing marginal probabilities over single variables in (sparse) latent Gaussian models. Probabilistic models with latent Gaussian variables are of interest in many areas of statistics, such as spatial data analysis (Rue and Held, 2005), and machine learning, such as Gaussian process models (e.g., Kuss and Rasmussen, 2005). The general setting considered in this paper is as follows: the prior distribution over the latent variables is a Gaussian random field with a sparse precision (inverse covariance) matrix and the likelihood factorizes into a product of terms depending on just a single latent variable. Both the prior and the likelihood may depend on a small set of hyper-parameters. We are interested in the posterior marginal probabilities over single variables given all observations.

Rue et al. (2009) propose an integrated nested Laplace approximation to approximate these posterior marginal distributions. Their procedure consists of three steps. 1) Approximate the posterior of the hyper-parameters given the data and use this to determine a grid of hyper-parameter values. 2) Approximate the posterior marginal distributions given the data and the hyper-parameters values on the grid. 3) Numerically integrate the product of the two approximations to obtain the posterior

©2011 Botond Cseke and Tom Heskes.

marginals of interest. The crucial contribution is the improved marginal posterior approximation in step 2), based on the approach of Tierney and Kadane (1986), that goes beyond the Gaussian approximation and takes into account higher order characteristics of (all) likelihood terms. Comparing their approach with Monte Carlo sampling techniques on several high-dimensional models, they show that their procedure is remarkably fast and accurate.

The main objective of the current paper is to see whether we can improve upon the approach of Rue et al. (2009). Expectation propagation (Minka, 2001), a method for approximate inference developed and studied mainly in the machine learning community, is then an obvious candidate. It is well-known to yield approximations that are more accurate than the Laplace method (e.g., Minka, 2001; Kuss and Rasmussen, 2005). Furthermore, expectation propagation can still be applied in cases where the Laplace method is out of the question, for example, when the log-posterior is not twice-differentiable (Seeger, 2008). The typical price to be paid is that of higher computational complexity. However, we will see that, using a parallel instead of a sequential updating scheme, expectation propagation is at most a small constant factor slower than the Laplace method in applications on sparse Gaussian models with many latent variables. Moreover, along the way we will arrive at further approximations (both for expectation propagation and the Laplace method) that yield an order of magnitude speed-up, with hardly any degradation in performance.

The paper is structured as follows. In Sections 1.1 and 2 we specify the model and briefly present the Laplace method and expectation propagation. In Section 3, we introduce and compare several methods for correcting marginals given a fixed setting of the hyper-parameters. In Section 4.6, we discuss the computational complexity of these methods when applied to sparse models. In Section 5, we introduce a method for numerical integration over hyper-parameters and finally in Section 6, we show that the proposed methods are competitive both in computational complexity and accuracy with the methods introduced in Rue et al. (2009).

In order to increase the readability of the paper we include a schematic figure (Figure 13) and an explanatory list (Section D of the Appendix) of the marginal approximation methods we introduce or refer to. In the following we define the model and give a short outline of how we proceed to approximate the marginal densities.

1.1 Latent Gaussian Models

In this section, we introduce notation and define the model under consideration. Let $p(\boldsymbol{y}|\boldsymbol{x},\theta_l)$ be the conditional probability of the observations $\boldsymbol{y} = (y_1,\ldots,y_n)^T$ given the latent variables $\boldsymbol{x} = (x_1,\ldots,x_n)^T$ and the hyper-parameters θ_l. We assume that $p(\boldsymbol{y}|\boldsymbol{x},\theta_l)$ factorizes as

$$p(\boldsymbol{y}|\boldsymbol{x},\theta_l) = \prod_{i=1}^{n} p(y_i|x_i,\theta_l).$$

The prior $p(\boldsymbol{x}|\theta_p)$ over the latent variables is taken to be Gaussian with canonical parameters $\boldsymbol{h}(\boldsymbol{\theta}_p)$ and $\boldsymbol{Q}(\boldsymbol{\theta}_p)$, that is,

$$p(\boldsymbol{x}|\boldsymbol{\theta}_p) \propto \exp\left(\boldsymbol{x}^T\boldsymbol{h}(\boldsymbol{\theta}_p) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}(\boldsymbol{\theta}_p)\,\boldsymbol{x}\right).$$

Examples for $p(\boldsymbol{x}|\theta_p)$ include Gaussian process models, where $\boldsymbol{Q}^{-1}(\boldsymbol{\theta}_p)$ is the covariance matrix at the corresponding input and Gaussian Markov random fields, where the elements of $\boldsymbol{Q}(\boldsymbol{\theta}_p)$ are the interactions strengths $Q_{ij}(\boldsymbol{\theta}_p)$ between the latent variables x_i and x_j. The prior $p(\boldsymbol{\theta}_l,\boldsymbol{\theta}_p)$ over the hyper-parameters is typically taken to be non-informative—uniform for location variables and

log-uniform for scale variables—and factorizes w.r.t. $\boldsymbol{\theta}_l$ and $\boldsymbol{\theta}_p$. In order to simplify the notation, we use the proxy $\boldsymbol{\theta} = (\boldsymbol{\theta}_l, \boldsymbol{\theta}_p)$ to denote the hyper-parameters of the model.

The joint distribution of the variables in the model we study is

$$p(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}) \propto \prod_{i=1}^{n} p(y_i|x_i, \theta) \exp\left(\boldsymbol{x}^T \boldsymbol{h}(\boldsymbol{\theta}) - \frac{1}{2}\boldsymbol{x}^T \boldsymbol{Q}(\boldsymbol{\theta})\boldsymbol{x}\right) p(\boldsymbol{\theta}).$$

We take $\boldsymbol{y}$ fixed and we consider the problem of computing accurate approximations of the posterior marginal densities of the latent variables $p(x_i|\boldsymbol{y}, \boldsymbol{\theta})$, given a fixed hyper-parameter value. Then we integrate these marginals over the approximations of the hyper-parameters posterior density $p(\boldsymbol{\theta}|\boldsymbol{y})$. The exact quantities are given by the formulas

$$p(x_i|\boldsymbol{y}, \boldsymbol{\theta}) = \frac{1}{p(\boldsymbol{y}|\boldsymbol{\theta})} p(y_i|x_i, \boldsymbol{\theta}) \int d\boldsymbol{x}_{\setminus i}\, p(\boldsymbol{x}|\boldsymbol{\theta}) \prod_{j \neq i} p(y_j|x_j, \boldsymbol{\theta}), \tag{1}$$

$$p(\boldsymbol{\theta}|\boldsymbol{y}) \propto p(\boldsymbol{\theta})\, p(\boldsymbol{y}|\boldsymbol{\theta}). \tag{2}$$

We use the term *evidence* for $p(\boldsymbol{y}|\boldsymbol{\theta}) = \int d\boldsymbol{x}\, p(\boldsymbol{y}, \boldsymbol{x}|\boldsymbol{\theta})$. In the following we omit $p(\boldsymbol{y}|\boldsymbol{x}, \theta)$'s and $p(\boldsymbol{x}|\theta)$'s dependence on θ whenever it is not relevant and use $t_i(x_i)$ as an alias of $p(y_i|x_i, \theta)$ and $p_0(\boldsymbol{x})$ as an alias of $p(\boldsymbol{x}|\theta)$. We use the notation $p(\boldsymbol{x}) = Z_p^{-1} p_0(\boldsymbol{x}) \prod_i t_i(x_i)$, with $Z_p(\boldsymbol{\theta}) \equiv p(\boldsymbol{y}|\boldsymbol{\theta})$. A Gaussian approximation of p will be denoted by q and Z_q will denote its normalization constant.

1.2 An Outline of the Main Methods Presented in the Paper

In this paper, we will discuss a variety of methods for approximating marginals in latent Gaussian models. To assist the reader, we give an outline of these methods, leaving the technical details for later sections. We advise the reader to consult Figure 13 and Section D in the Appendix for a schematic figure and the corresponding explanatory list.

The posterior probability density $p(\boldsymbol{x})$ is proportional to a (sparse) multivariate Gaussian distribution over all latent variables and a product of non-Gaussian terms $t_j(x_j)$, each of which depends on just a single latent variable. The first step is to find a global, Gaussian approximation of this posterior. There is obviously no need to approximate the Gaussian prior part, but we then do have to approximate the non-Gaussian terms $t_j(x_j)$ by Gaussian forms $\tilde{t}_j(x_j)$, thus constructing an approximation $q(\boldsymbol{x}) \propto p_0(\boldsymbol{x}) \prod_j \tilde{t}_j(x_j)$. Here we consider two choices, which we refer to as the Laplace method and expectation propagation.

The Laplace method (LM) finds the maximum of the (log) posterior and then makes a second order Taylor approximation around this maximum. It is easy to see that the Hessian, the matrix with second derivatives in this second order Taylor approximation, consists of the (sparse) precision matrix resulting from the Gaussian prior plus a diagonal term consisting of second derivatives of the logarithm of each of the terms $t_j(x_j)$. Hence, the approximation resulting from the Laplace method can always be written as the original prior $p_0(\boldsymbol{x})$ times a product of so-called term approximations $\tilde{t}_j(x_j)$, each of which has a Gaussian form (not necessarily normalizable) depending on just a single latent variable.

Expectation propagation (EP) aims to iteratively refine these term approximations $\tilde{t}_j(x_j)$. It works as follows. In the product of Gaussian prior times term approximations, we replace the term approximation that we aim to refine by the corresponding original non-Gaussian term. The resulting distribution $t_j(x_j)\tilde{t}_j(x_j)^{-1} q(\boldsymbol{x})$ is referred to as the tilted distribution: a Gaussian form $\tilde{t}_j(x_j)^{-1} q(\boldsymbol{x})$

times a non-Gaussian term $t_j(x_j)$ depending on a single latent variable. We then compute the zeroth, first, and second moments of this tilted distribution (e.g., through one-dimensional numerical integration) and determine the term approximation $\tilde{t}_j^{new}(x_j)$ which results in the same zeroth, first, and second moments. In the Gaussian approximation, we replace the old term approximation $\tilde{t}_j(x_j)$ by this new term approximation $\tilde{t}_j^{new}(x_j)$. In its original setting, expectation propagation refines term approximations $\tilde{t}_j(x_j)$ sequentially. In this particular setting of sparse models with many latent variables., a tremendous speed-up speed-up can be obtained by using a batch-mode version, that is, updating the term approximations in parallel.

Whichever procedure one prefers, Laplace or expectation propagation, this first step yields a global Gaussian approximation $q(\boldsymbol{x})$ of the original non-Gaussian posterior. We can then write the exact non-Gaussian posterior as this Gaussian approximation $q(\boldsymbol{x})$ times a product of correction terms, where each correction term is nothing but the original term $t_j(x_j)$ divided by its term approximation $\tilde{t}_j(x_j)$. Any further approximation is based on the assumption that these correction terms are close to 1 in average w.r.t. q, that is, that the Gaussian term approximation is indeed a sensible approximation of the original non-Gaussian term in the region where the main mass of q lies.

We are interested in accurate approximations of marginals $p(x_i)$ on a single variable, say x_i. For this, we have to integrate out all variables except x_i. Decomposing the global Gaussian approximation $q(\boldsymbol{x})$ into the product of $q(x_i)$ and the conditional $q(\boldsymbol{x}_{\setminus i}|x_i)$, we can take both $q(x_i)$ and the correction term depending on x_i outside of the integral over $\boldsymbol{x}_{\setminus i}$. The remaining integrand is then the conditional Gaussian $q(\boldsymbol{x}_{\setminus i}|x_i)$ times the product of all correction terms, except the one for x_i. The crucial observation here is that this integrand is of exactly the same form as the problem we started with: a (sparse) Gaussian prior (here the conditional $q(\boldsymbol{x}_{\setminus i}|x_i)$) times a product of non-Gaussian terms (here the correction terms). In principle, we could again use the Laplace method or expectation propagation to approximate the integral. Doing this for the Laplace method yields the Laplace approximation of Tierney and Kadane (1986) (LA-TK) (Section 3.1). Doing the same in conjunction with expectation propagation leads to the approximation in Section 3.2.

However, both easily become very expensive, since we have to apply the Laplace method or run a full expectation propagation for each setting of x_i. Luckily, we now have an additional property that we can try to exploit: the non-Gaussian correction terms in the integrand have been constructed such that they are somehow close to 1.

The first, obvious approximation is to replace these correction terms within the integral by 1, leaving only the product of $q(x_i)$ and the correction term depending on x_i. We will refer to this type of approximation as a local approximation. In the case of expectation propagation it is exactly the corresponding marginal of the tilted distribution and we refer to it by EP-L (Section 3). The same approximation, but then in conjunction with the Laplace method is referred to as LM-L (Section 3).

The method proposed by Rue and co-workers can be viewed as a compromise between applying the expensive Laplace method (LA-TK) and the cheap local approximation (LM-L). Instead of finding the optimum of the integrand (conditional Gaussian times correction terms) and expanding around that, Rue et al. (2009) propose to expand the integrand around the optimum of conditional Gaussian only. Essentially, in the computation of the optimum of the integrand they hereby ignore the correction terms and simply set them to 1. Their method is referred to as LA-CM (see Section 4.1.2), where CM stands for conditional mean. It is straightforward and from the computational point of view relatively inexpensive to correct for the fact that the Taylor expansion is not done at the maximum of the integrand. The method which takes this into account is called LA-CM2 (Section 4.1.2).

In conjunction with EP we can use a similar argument. The term approximations inside the integral are optimized for the global Gaussian approximation, that is, when averaging over x_i. A full run of expectation propagation would give the term approximations that are optimal conditioned upon x_i, instead of marginalized over x_i. This difference is likely to be rather small and hence we expect that the main difference can be picked up by doing just one (parallel) iteration of expectation propagation. This approximation is referred to as EP-1STEP (Section 4.1.1). Iterating EP until convergence would lead to an approximation that will be referred to as EP-FULL (Section 4.1.1).

Another line of reasoning, followed by Opper et al. (2009), is to Taylor expand the correction terms around 1 (or their logarithm around 0). This is referred to as EP-OPW (Section 4.3). In their original work, they apply this Taylor expansion not only for the correction terms inside the integral, but also for the correction term depending on x_i outside of the integral, which is unnecessary in the current context. The interesting observation here is that, in a first-order Taylor expansion, the correlations within $q(\boldsymbol{x}_{\setminus i}|x_i)$ become irrelevant and the integral over $\boldsymbol{x}_{\setminus i}$ factorizes into a product of one-dimensional integrals. This (and also the existence of variational bounds on the marginals) suggests the approximation EP-FACT (Section 4.2), which corresponds to EP-1STEP, but then with $q(\boldsymbol{x}_{\setminus i}|x_i)$ replaced by its factorization $\prod_j q(x_j|x_i)$. The same replacement for LA-CM gives a method we refer to as LA-FACT (Section 4.2). Both EP-FACT and LA-FACT are an order of magnitude faster than their counterparts based on the non-factorized conditional distribution since they do not require computing the log-determinant of a high-dimensional (sparse) matrix for each setting of x_i. By a recursive application of the factorization principle one can obtain better approximations, which will be detailed in a future report. We use EP-FACTN to denote these approximations. In the following we expand the above mentioned ideas. We start with the presentation of the global approximation methods.

2. Global Gaussian Approximations

A close inspection of (1) and (2) shows that computing $p(x_i|\boldsymbol{y},\boldsymbol{\theta})$ leads to computing similar integrals as for $p(\boldsymbol{y}|\boldsymbol{\theta})$. In this section, we review two approximation schemes that approximate such integrals: the Laplace method and expectation propagation (Minka, 2001). There are other approximation schemes, such as the variational approximation (e.g., Opper and Archambeau, 2009). The marginal approximation methods we propose for expectation propagation in Section 3 can be, under mild conditions, translated to the variational approximation in Opper and Archambeau (2009). For this reason, we will not discuss the details of this method.

2.1 The Laplace Method

The Laplace method approximates the evidence Z_p and, as a side product, it provides Gaussian approximation that is characterized by the local properties of the distribution at its mode $\boldsymbol{x}^* = \operatorname{argmax}_{\boldsymbol{x}} \log p(\boldsymbol{x})$. The mean parameter of the corresponding approximating Gaussian density is $\boldsymbol{m} = \boldsymbol{x}^*$ while the inverse of the covariance parameter $\boldsymbol{V}$ is the Hessian of $-\log p$ at $\boldsymbol{x}^*$.

The idea behind the method is the following. Let $f = \log p$. Expanding f in second order at an arbitrary value $\tilde{\boldsymbol{x}}$, we get

$$\begin{aligned} f(\boldsymbol{x}) &= f(\tilde{\boldsymbol{x}}) + (\boldsymbol{x}-\tilde{\boldsymbol{x}})^T \nabla_{\boldsymbol{x}} f(\tilde{\boldsymbol{x}}) \\ &\quad + \frac{1}{2}(\boldsymbol{x}-\tilde{\boldsymbol{x}})^T \nabla^2_{\boldsymbol{x}\boldsymbol{x}} f(\tilde{\boldsymbol{x}})(\boldsymbol{x}-\tilde{\boldsymbol{x}}) + R_2[f](\boldsymbol{x};\tilde{\boldsymbol{x}}), \end{aligned} \tag{3}$$

where $R_2[f](\boldsymbol{x};\tilde{\boldsymbol{x}})$ is the residual term of the expansion at $\tilde{\boldsymbol{x}}$ with $R_2[f](\tilde{\boldsymbol{x}};\tilde{\boldsymbol{x}}) = 0$. By using the change of variables $\boldsymbol{s} = \boldsymbol{x} - \tilde{\boldsymbol{x}}$, we have

$$\begin{aligned} \log \int d\boldsymbol{x} e^{f(\boldsymbol{x})} &= f(\tilde{\boldsymbol{x}}) - \frac{1}{2}\nabla_{\boldsymbol{x}} f(\tilde{\boldsymbol{x}})^T \left[\nabla^2_{\boldsymbol{xx}} f(\tilde{\boldsymbol{x}})\right]^{-1} \nabla_{\boldsymbol{x}} f(\tilde{\boldsymbol{x}}) \\ &\quad -\frac{1}{2}\log|-\nabla^2_{\boldsymbol{xx}} f(\tilde{\boldsymbol{x}})| + \log \mathrm{E}_{\boldsymbol{s}}\left[e^{R_2[f](\boldsymbol{s}+\tilde{\boldsymbol{x}};\tilde{\boldsymbol{x}})}\right], \end{aligned} \tag{4}$$

where $|\cdot|$ denotes the determinant and the expectation w.r.t. $\boldsymbol{s}$ is taken over a normal distribution with canonical parameters $\nabla_{\boldsymbol{x}} f(\tilde{\boldsymbol{x}})$ and $-\nabla^2_{\boldsymbol{xx}} f(\tilde{\boldsymbol{x}})$.

A closer look at (3) and (4) suggests that choosing $\tilde{\boldsymbol{x}} = \boldsymbol{x}^*$ and using the approximation $R_2[\log p](\boldsymbol{x};\tilde{\boldsymbol{x}}) \approx 0$ yields an approximation of the log evidence

$$\log Z_p \approx \log p(\boldsymbol{x}^*) - \frac{1}{2}\log|-\nabla^2_{\boldsymbol{xx}} \log p(\boldsymbol{x}^*)|. \tag{5}$$

In the meantime, p can be approximated by the Gaussian

$$q(\boldsymbol{x}) = N\left(\boldsymbol{x}|\boldsymbol{x}^*, -\left[\nabla^2_{\boldsymbol{xx}} \log p(\boldsymbol{x}^*)\right]^{-1}\right).$$

Note that any reasonably good approximation of $\mathrm{E}_{\boldsymbol{s}}\left[e^{R_2[f](\boldsymbol{s}+\tilde{\boldsymbol{x}};\tilde{\boldsymbol{x}})}\right]$ can improve the accuracy of the approximation in (5).

The Laplace method requires the second order differentiability of $\log p$ at $\boldsymbol{x}^*$, thus a sufficient condition for the applicability of this approximation scheme is the second order differentiability of $\log p$. The necessary condition is the second order differentiability at the mode $\boldsymbol{x}^*$. A distribution p for which the method fails to give any meaningful information about the variances is, for example, when $p(y_j|x_j) = \lambda \exp(-\lambda|y_j - x_j|)/2$. In this case, the Hessian of $\log p$ at an arbitrary point $\tilde{\boldsymbol{x}}$ is either equal to the precision $\boldsymbol{Q}$ of the prior or it is undefined. Since the Laplace method captures the characteristics of the modal configuration, it often gives poor estimates of the normalization constant (e.g., Kuss and Rasmussen, 2005). The example in Section 4.1 shows how this behavior influences the approximation of the marginals in case of a two dimensional toy model. However, compared to other methods, the main advantage of the Laplace method is its speed. The optimization of $\log p$ w.r.t. $\boldsymbol{x}$ for computing $\boldsymbol{m} = \boldsymbol{x}^*$ requires only a few Newton steps.

2.2 Expectation Propagation

Expectation propagation (EP) approximates the integral for computing the evidence in the following way. Let us assume that q is a Gaussian approximation of p constrained to have the form $q(\boldsymbol{x}) = Z_q^{-1} p_0(\boldsymbol{x}) \prod_j \tilde{t}_j(x_j)$. Then the evidence can be approximated as

$$\begin{aligned} Z_p &= \int d\boldsymbol{x}\, p_0(\boldsymbol{x}) \prod_j t_j(x_j), \\ &= Z_q \int d\boldsymbol{x}\, q(\boldsymbol{x}) \prod_j \frac{t_j(x_j)}{\tilde{t}_j(x_j)}, \\ &\approx Z_q \prod_j \int dx_j\, q(x_j) \frac{t_j(x_j)}{\tilde{t}_j(x_j)} \end{aligned} \tag{6}$$

and we are left with choosing the appropriate $\tilde{t}_j(x_j)$s that yield both a good approximation of the evidence and of $p(\boldsymbol{x})$. EP computes the terms $\tilde{t}_j(x_j)$ by iterating

$$\tilde{t}_j^{new}(x_j) \propto \frac{\text{Collapse}\left(t_j(x_j)\tilde{t}_j(x_j)^{-1}q(\boldsymbol{x})\right)}{q(\boldsymbol{x})}\tilde{t}_j(x_j), \text{ for all } j = 1,\ldots,n, \tag{7}$$

where $\text{Collapse}(r) = \text{argmin}_{r'\in\mathcal{N}} \mathrm{D}\left[r||r'\right]$ is the Kullback-Leibler (KL) projection of the distribution r into the family of Gaussian distributions $\mathcal{N}$. In other words, it is the Gaussian distribution that matches the first two moments of r. Using the properties of the KL divergence, one can check that when the terms t_j depend only on the variables x_j then $\text{Collapse}\left(t_j(x_j)\tilde{t}_j(x_j)^{-1}q(\boldsymbol{x})\right)/q(\boldsymbol{x}) = \text{Collapse}\left(t_j(x_j)\tilde{t}_j(x_j)^{-1}q(x_j)\right)/q(x_j)$, therefore, the iteration in (7) is well defined. At any fixed point of this iteration, we have a set of $\tilde{t}_j(x_j)$ terms for which $\text{Collapse}\left(t_j(x_j)\tilde{t}_j(x_j)^{-1}q(\boldsymbol{x})\right) = q(\boldsymbol{x})$ for any $j \in \{1,\ldots,n\}$. By defining the cavity distribution $q^{\backslash j}(\boldsymbol{x}) \propto \tilde{t}_j(x_j)^{-1}q(\boldsymbol{x})$ and scaling the terms t_j, the above fixed point condition can be rewritten as

$$\int dx_j \left\{1, x_j, x_j^2\right\} q^{\backslash j}(x_j)\tilde{t}_j(x_j) = \int dx_j \left\{1, x_j, x_j^2\right\} q^{\backslash j}(x_j)t_j(x_j),\ j = 1,\ldots,n,$$

and so, the approximation for Z_p has the form

$$Z_p \approx \int d\boldsymbol{x}\, p_0(\boldsymbol{x}) \prod_j \tilde{t}_j(x_j).$$

Expectation propagation, can be viewed as a generalization of loopy belief propagation (e.g., Murphy et al., 1999) to probabilistic models with continuous variables and also as an iterative application of the assumed density filtering procedure (e.g., Csató and Opper, 2001). An equivalent algorithm for Gaussian process classification based on statistical physics methods was derived in Opper and Winther (2000). A close inspection of the parametric form of the iteration in Section C of the Appendix shows that the convexity of $\log \int d\boldsymbol{x} N(\boldsymbol{x}|\boldsymbol{m},\boldsymbol{V})\, t_j(x_j)$ w.r.t. $\boldsymbol{m}$ or the concavity of $\log t_j(x_j)$ (Seeger, 2008) is a sufficient condition for the terms $\tilde{t}_j$s to be normalizable and thus for the existence of q^{new}. However, this alone does not guarantee convergence. To our knowledge, the issue of EP's convergence in case of the models we study in this paper is still an open question. The iteration in (7) can also be derived by using variational free energies (e.g., Heskes et al., 2005; Minka, 2005). It can be relaxed such that the projections are taken on $t_j(x_j)^{\alpha}\tilde{t}_j(x_j)^{-\alpha}q(\boldsymbol{x})$, with $\alpha \in (0,1]$. The limit $\alpha \to 0$ corresponds to the variational approximation of Opper and Archambeau (2009).

In a personal correspondence, H. Rue emphasized that in many real world models, linear constraints of the form $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ have to be considered and expressed the concern that EP might not be suited to handle such constraints. Incorporating these constraints into EP would require to define updates for terms of the form $\delta_0(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$. In the following we propose a way to deal with such terms. First we start out by deriving a sampling distributions for the Gaussian random variables $\boldsymbol{x}|\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, where we assume that $\boldsymbol{A}$ is a $k \times n$ matrix with $k < n$. Let $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{m},\boldsymbol{V})$ and $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} + \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, v\boldsymbol{I})$. Then the conditional density of $\boldsymbol{x}$ given $\boldsymbol{y}$ is a Gaussian with parameters $\boldsymbol{m} + \boldsymbol{V}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{V}\boldsymbol{A}^T + v\boldsymbol{I})^{-1}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m} + \boldsymbol{b})$ and $\boldsymbol{V} - \boldsymbol{V}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{V}\boldsymbol{A}^T + v\boldsymbol{I})^{-1}\boldsymbol{A}\boldsymbol{V}$. Setting $\boldsymbol{y} = \boldsymbol{0}$ and taking the limit $v \to 0$ we find that

$$\boldsymbol{x}|\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \sim \mathcal{N}\left(\boldsymbol{m} - \boldsymbol{V}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{V}\boldsymbol{A}^T)^{-1}(\boldsymbol{A}\boldsymbol{m} - \boldsymbol{b}), \boldsymbol{V} - \boldsymbol{V}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{V}\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{V}\right). \tag{8}$$

As a consequence, we propose the following procedure: 1) first we perform the term updates for all "regular" terms, then we project the new moment parameters of q according to (8), 2) the value of the corresponding factor in (6) is $N(\mathbf{0}|\boldsymbol{Am}-\boldsymbol{b},\boldsymbol{AVA}^T)$ and it corresponds to a Bayesian update in the limit $v \to 0$.

3. Approximation of the Posterior Marginals

The global approximations provide Gaussian approximations q of p and approximations of the evidence Z_p. The Gaussian approximation q can be used to compute Gaussian approximations of posterior marginals. In case of the Laplace method this only requires linear algebraic methods (computing the diagonal elements of the Hessian's inverse), while in the case of EP, the approximate marginals are a side product of the method itself. We refer to the corresponding Gaussian marginal approximations by LM-G (Laplace method) and EP-G (EP). Moreover, one can make use of the approximation method at hand in order to improve the Gaussian approximate marginals.

In case of the Laplace method, one can easily check that the residual term in (3) decomposes as $R_2[\log p](\boldsymbol{x};\tilde{\boldsymbol{x}}) = \sum_j R_2[\log t_j](x_j;\tilde{x}_j)$, thus, when approximating the marginal of x_i it is sufficient to assume $R_2[\log t_j](x_j;\tilde{x}_j) \approx 0$ only for $j \neq i$. This yields a locally improved approximation $q(x_i) \times \exp R_2[\log t_i](x_i;x_i^*)$ to which we refer by LM-L.

As shown in Section 2.2, EP is built on exploiting the low-dimensionality of $t_i(x_i)$ and approximating the *tilted* marginals $t_i(x_i)q^{\setminus i}(x_i)$. These are known to be better approximations of the marginals $p(x_i)$ than $q(x_i)$ (e.g., Opper and Winther, 2000; Opper et al., 2009). We refer to this approximation by EP-L.

These observations show that there are ways to improve the marginals of the global approximation q by exploiting the properties of the methods. For the moment, however, we postpone this to Section 4 and first try to compute the marginals from scratch. This gives us some insight into where to look for further improvements.

The exact marginals can be computed as

$$p(x_i) = \frac{1}{Z_p} t_i(x_i) \int d\boldsymbol{x}_{\setminus i}\, p_0\left(\boldsymbol{x}_{\setminus i}, x_i\right) \prod_{j \neq i} t_j(x_j), \tag{9}$$

thus, as mentioned earlier, computing the marginal for a fixed x_i leads to computing the normalization constant of the distribution $p_0\left(\boldsymbol{x}_{\setminus i}|x_i\right)\prod_{j\neq i} t_j(x_j)$. Therefore, we can use our favorite method to approximate it. In the following, we present the details of these procedures for the Laplace method and EP.

3.1 Laplace Approximation

We use the same line of argument as in Section 2.1, but now we fix x_i and expand $\log p$ w.r.t. $\boldsymbol{x}_{\setminus i}$ at an arbitrary $\tilde{\boldsymbol{x}}_{\setminus i}$. The expression is identical to (3) with $\boldsymbol{x} = (x_i, \boldsymbol{x}_{\setminus i}^T)^T$ and $\tilde{\boldsymbol{x}} = (x_i, \tilde{\boldsymbol{x}}_{\setminus i}^T)^T$. Let $\boldsymbol{x}_{\setminus i}^*(x_i) = \operatorname{argmax}_{\boldsymbol{x}_{\setminus i}} \log p\left(x_i, \boldsymbol{x}_{\setminus i}\right)$ and let $\tilde{\boldsymbol{x}}_{\setminus i} = \boldsymbol{x}_{\setminus i}^*(x_i)$. Then the approximation of (4) simplifies to a form similar to (5), that is, the approximation of the marginal density, up to the constant $\log Z_p$, is given by

$$\log \int d\boldsymbol{x}_{\setminus i} p(\boldsymbol{x}) \approx \log p\left(x_i, \boldsymbol{x}_{\setminus i}^*(x_i)\right) - \frac{1}{2}\log\left|-\nabla^2_{\boldsymbol{x}_{\setminus i}\boldsymbol{x}_{\setminus i}} \log p\left(x_i, \boldsymbol{x}_{\setminus i}^*(x_i)\right)\right|. \tag{10}$$

This approximation is known in statistics as the Laplace approximation (Tierney and Kadane, 1986) and we will refer to it as $\tilde{p}_i^{\text{LA-TK}}(x_i)$.

The error of the approximation can be characterized in terms of the residual of the second order expansion. The residual decomposes as

$$R_2\left[\log p\right](\boldsymbol{x};\tilde{\boldsymbol{x}}) = \sum_{j\neq i} R_2\left[\log t_j\right]\left(s_j + x_j^*(x_i); x_j^*(x_i)\right)$$

and the expectation (see Equation (4)) is taken w.r.t. $\boldsymbol{s} \in \mathbb{R}^{(n-1)}$ having a normal density with mean $\mathbf{0}$ and inverse covariance $-\nabla^2_{\boldsymbol{x}_{\setminus i}\boldsymbol{x}_{\setminus i}} \log p(x_i, \boldsymbol{x}^*_{\setminus i}(x_i))$. This means that in principle we have exact estimates of the error and that any reasonable approximation of the integral can improve the quality of the approximation in (10).

3.2 Expectation Propagation

The integral in (9) can also be approximated using EP. As mentioned above EP typically provides better approximations of $\log Z_p$ than the Laplace method. For this reason, the marginals computed by approximating (9) using EP are expected to be more accurate. The procedure is as follows: (1) fix x_i and compute the canonical parameters of $p_0(\boldsymbol{x}_{\setminus i}|x_i)$ given by $\boldsymbol{h}_{\setminus i} - \boldsymbol{Q}_{\setminus i,i}x_i$ and $\boldsymbol{Q}_{\setminus i,\setminus i}$ and (2) use EP to approximate the integral in (9). Thus we approximate the integral by leaving out $p_0(x_i)$ and $t_i(x_i)$ and applying EP using the prior $p_0(\boldsymbol{x}_{\setminus i}|x_i)$ and the terms $t_j(x_j)$, $j \neq i$.

4. Approximation of the Posterior Marginals by Correcting the Global Approximations

As we have seen in the previous section, computing the marginal for a given fixed x_i value can be as expensive as the global procedure itself. On the other hand, however, there are ways to improve the marginals of the global approximation. In this section, we start from the "direct" approach and try to re-use the results of the global approximation to improve on the locally improved marginals LM-L and EP-L.

We start with the observation that for all the presented approximation methods, we can write the approximating distribution q as

$$q(\boldsymbol{x}) = \frac{1}{Z_q} p_0(\boldsymbol{x}) \prod_j \tilde{t}_j(x_j). \tag{11}$$

In case of the Laplace method, the canonical parameters of the Gaussian functions $\tilde{t}_j$ are defined by the parameters of the Taylor expansion of $\log t_j$ at x_i^*, while in case of EP, they are the parameters corresponding to EP's fixed point.

In the following, we do not keep track of the normalization constants that are independent of x_i. In order to avoid overloading the notation and to express that a distribution is approximated as proportional to an expression on the right hand side of the $\approx$ relation, we occasionally use Z as a proxy for unknown normalization constants. One can keep track of these constants, but in most cases, from the practical point of view, it is easier to perform a univariate numerical interpolation followed by numerical quadrature and (re)normalization.

4.1 Improving the Marginals of the Global Approximations

Given a global Gaussian approximation $q(\boldsymbol{x})$ of the form (11) with corresponding term approximations $\tilde{t}_i(x_i)$, we can rewrite $p(x_i)$ as

$$\begin{aligned} p(x_i) &= \frac{Z_q}{Z_p}\frac{t_i(x_i)}{\tilde{t}_i(x_i)}\int d\boldsymbol{x}_{\setminus i}\, q(\boldsymbol{x})\prod_{j\neq i}\frac{t_j(x_j)}{\tilde{t}_j(x_j)}, \\ &= \frac{Z_q}{Z_p}\frac{t_i(x_i)}{\tilde{t}_i(x_i)}q(x_i)\int d\boldsymbol{x}_{\setminus i}\, q\left(\boldsymbol{x}_{\setminus i}|x_i\right)\prod_{j\neq i}\frac{t_j(x_j)}{\tilde{t}_j(x_j)}, \\ &= \frac{Z_q}{Z_p}\varepsilon_i(x_i)q(x_i)\int d\boldsymbol{x}_{\setminus i}\, q\left(\boldsymbol{x}_{\setminus i}|x_i\right)\prod_{j\neq i}\varepsilon_j(x_j)\,, \end{aligned} \tag{12}$$

where we define $\varepsilon_i(x_i)\equiv t_i(x_i)/\tilde{t}_i(x_i)$. In case of EP, the term approximations $\tilde{t}_i(x_i)$ are chosen to be close to the terms $t_i(x_i)$ in average w.r.t. $q(x_i)$. For this reason, we expect the $\varepsilon_i(x_i)$'s to be close to 1 in average w.r.t $q(x_i)$.

Equation (12) is still exact and it shows that there are two corrections to the Gaussian approximation $q(x_i)$: one direct, local correction through $\varepsilon_i(x_i)$ and one more indirect correction through the (weighted integral over) $\varepsilon_j(x_j)$s for $j\neq i$. The direct, local correction comes without additional cost and suggests the above-mentioned (Section 3) local approximation

$$p(x_i)\approx\frac{1}{Z}\varepsilon_i(x_i)q(x_i)\,.$$

We use the notations $\tilde{p}_i^{\text{EP-L}}(x_i)$ and $\tilde{p}_i^{\text{LM-L}}(x_i)$ for the approximations following the global Gaussian approximations by EP and Laplace method, respectively.

To improve upon this approximation, we somehow have to get a handle on the indirect correction

$$c_i(x_i)\equiv\int d\boldsymbol{x}_{\setminus i}\, q\left(\boldsymbol{x}_{\setminus i}|x_i\right)\prod_{j\neq i}\varepsilon_j(x_j)\,. \tag{13}$$

Again, for each x_i, we are in fact back to the form (9): we have to estimate the normalization constant of a latent Gaussian model, where $q\left(\boldsymbol{x}_{\setminus i}|x_i\right)$ now plays the role of an $(n-1)$-dimensional Gaussian prior and the $\varepsilon_j(x_j)$s are terms depending on a single variable. Running a complete procedure, be it EP or Laplace, for each x_i—as described in Sections 3.1 and 3.2—is often computationally too intensive and further approximations are needed to reduce the computational burden.

4.1.1 IMPROVING THE MARGINALS RESULTING FROM EP

Let us write $\tilde{\varepsilon}_j(x_j;x_i)$ for the term approximation of $\varepsilon_j(x_j)$ in the context of approximating $c_i(x_i)$. A full run of EP for each x_i may be too expensive, so instead we propose to perform just one simultaneous EP step for all $j\neq i$. Since the term approximations of the global EP approximation are tuned to make $\tilde{t}_j(x_j)$ close to $t_j(x_j)$ w.r.t. $q(x_i)$, it is plausible to initialize $\tilde{\varepsilon}_j(x_j;x_i)$ to 1. Following EP, computing the new term approximation for term j then amounts to choosing $\tilde{\varepsilon}_j(x_j;x_i)$ such that

$$\int dx_j\,\{1,x_j,x_j^2\}q(x_j|x_i)\tilde{\varepsilon}_j(x_j;x_i)=\int dx_j\,\{1,x_j,x_j^2\}q(x_j|x_i)\varepsilon_j(x_j), \tag{14}$$

that is, we get $\tilde{\varepsilon}_j(x_j;x_i)$ by *collapsing* $\varepsilon_j(x_j;x_i)q(x_j|x_i)$ into a Gaussian and dividing it by $q(x_j|x_i)$. As we have seen in Section 2.2, EP computes $\tilde{t}_j$ such that

$$\int dx_j\,\{1,x_j,x_j^2\}q(x_j)=\int dx_j\,\{1,x_j,x_j^2\}q(x_j)\varepsilon_j(x_j),$$

thus, the difference here is made by the conditioning on x_i and $\tilde{\varepsilon}_j(x_j;x_i)$ can be viewed as an update $\tilde{t}_j(x_j;x_i)$ of $\tilde{t}_j(x_j)$ that accounts "locally" for this difference—up to second order. Replacing the terms $\varepsilon_j(x_j)$ in (13) by their term approximations $\tilde{\varepsilon}_j(x_j;x_i)$ yields an estimate for $c_i(x_i)$. The corresponding approximation

$$p(x_i) \approx \frac{1}{Z}\varepsilon_i(x_i)q(x_i)\int d\boldsymbol{x}_{\setminus i}\, q\left(\boldsymbol{x}_{\setminus i}|x_i\right)\prod_{j\neq i}\tilde{\varepsilon}_j(x_j;x_i)$$

is referred to as $\tilde{p}_i^{\text{EP-1STEP}}(x_i)$. By performing further EP steps, one can refine the term approximations $\tilde{\varepsilon}_j(x_j;x_i)$. Iterating the EP steps until convergence (as mentioned above) leads to a similar (costly) approximation as in Section 3.2. We refer to the resulting approximation as EP-FULL.

4.1.2 IMPROVING THE MARGINALS RESULTING FROM THE LAPLACE METHOD

According to the Laplace approximation presented in Section 3.1 one has to recompute the conditional mode $\boldsymbol{x}^*_{\setminus i}(x_i)$ for every choice of x_i. In order to lessen the computational burden, Rue et al. (2009) propose to re-use the global approximation by approximating the conditional mode with the conditional mean, that is, $\boldsymbol{x}^*_{\setminus i}(x_i) \approx \boldsymbol{m}_{\setminus i} + \boldsymbol{V}_{\setminus i,i}V_{i,i}^{-1}(x_i - m_i)$, where $\boldsymbol{m} = \boldsymbol{x}^*(= \operatorname{argmax}_{\boldsymbol{x}} \log p(\boldsymbol{x}))$. This approximation often performs reasonably well when p is close to a Gaussian.

In our setting, the approximation proposed by Rue et al. (2009) can be understood as follows. The error terms ε_j can be identified with the residual terms, that is, $\log\varepsilon_i(x_i) = R_2\left[\log t_i\right](x_i;m_i)$. In order to assess $c_i(x_i)$, one could, in principle, apply the Laplace method to

$$f(\boldsymbol{x}_{\setminus i};x_i) \equiv q\left(\boldsymbol{x}_{\setminus i}|x_i\right)\prod_{j\neq i}\varepsilon_j(x_j)\,.$$

This would be identical to the direct method of Tierney and Kadane (1986) presented in Section 3.1. Using the conditional mean as an approximation of the conditional mode leads to ignoring the terms $\varepsilon_j(x_j)$ and using the mode of $q(\boldsymbol{x}_{\setminus i}|x_i)$. The corresponding approximation is of the form (4.1.1), where now $\tilde{\varepsilon}_j(x_j;x_i)$ follows from a second-order Taylor expansion of $\log\varepsilon_j(x_j)$ around the mode or mean of $q(x_j|x_i)$ instead of the mode of $f(\boldsymbol{x}_{\setminus i};x_i)$. We refer to this approximation as $\tilde{p}_i^{\text{LA-CM}}(x_i)$.

Taking a closer look at (4) and using our assumptions in Section 3.1, we can easily see that when we are not evaluating the normalization constant at the conditional mode, we can refine the approximation by adding $-\frac{1}{2}\nabla_{\boldsymbol{x}_{\setminus i}}f(\tilde{\boldsymbol{x}}_{\setminus i})[\nabla^2_{\boldsymbol{x}_{\setminus i},\boldsymbol{x}_{\setminus i}}f(\tilde{\boldsymbol{x}}_{\setminus i})]^{-1}\nabla_{\boldsymbol{x}_{\setminus i}}f(\tilde{\boldsymbol{x}}_{\setminus i})$, which is not identical to zero when the expansion in not made at the mode, that is, $\tilde{\boldsymbol{x}}_{\setminus i} \neq \boldsymbol{x}^*_{\setminus i}(x_i)$. As we will see in Section 4.7, this correction adds no significant computational burden to the method proposed in Rue et al. (2009). We refer to this approximation as $\tilde{p}_i^{\text{LA-CM2}}(x_i)$.

In order to further reduce computational effort, Rue et al. (2009) suggest additional approximations. Because they can only be expected to reduce the accuracy of the final approximation, we will not consider them in our experiments in Sections 4.5 and 6. Below we propose another EP-related approximation, motivated by theoretical bounds on the corrections $c_i(x_i)$.

4.2 Bounds and Factorized Approximations

The computational bottleneck in the above procedures for approximating the correction $c_i(x_i)$ is not computing appropriate approximations of the terms $\varepsilon_j(x_j)$, either through EP or Laplace, but instead computing the normalization of the resulting Gaussian form in (4.1.1), which leads to the

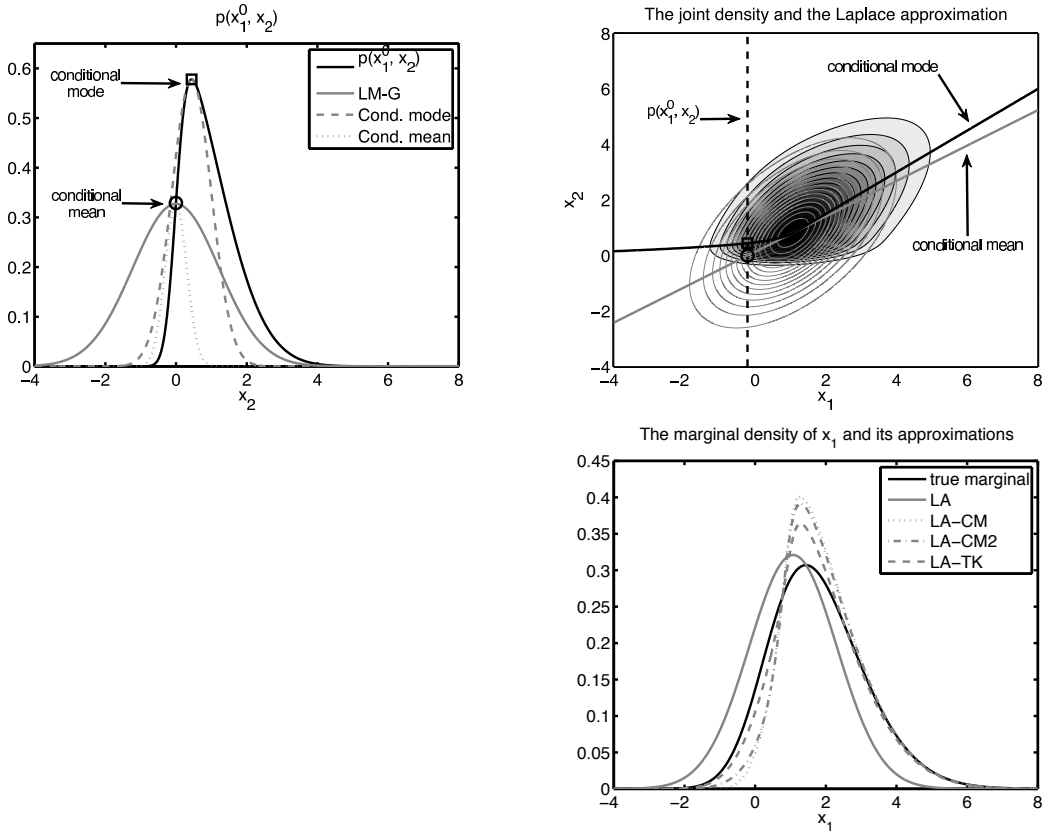


Figure 1: A two-dimensional example, illustrating how the Laplace approximation works and why it can fail. In the top-right panel, the black contour curves show the true distribution, the gray contour curves stand for the global Laplace approximation, and the black and gray curves show the conditional modes and the conditional means w.r.t. x_1. The square and circle outline these quantities for a fixed x_1^0. The dashed vertical line emphasizes the "slice" $p(x_1^0, x_2)$ at x_1^0. The top-left panel shows $p(x_1^0, x_2)$ and the approximations for computing its area under the curve. The areas under the Gaussian curves corresponding to the conditional mode (square) and the conditional mean (circle) are the approximations of $p(x_1^0) = \int dx_2\, p(x_1^0, x_2)$. The bottom-right panel shows the marginal of $p(x_1)$ and its approximations. The conditional mean can severely underestimate the mass for $x_1 = x_1^0$.

computation of a Gaussian normalization constant. Here we propose a simplification, which we motivate through its connection to bounds on the marginal correction $c_i(x_i)$.

Using Jensen's inequality, we obtain the lower bound on (13)

$$c_i(x_i) \geq \exp\left[\sum_{j\neq i}\int dx_j\, q(x_j|x_i)\log \varepsilon_j(x_j)\right] \equiv c_i^{\text{lower}}(x_i)\,.$$

Following Minka (2005), we can also get an upper bound:

$$c_i(x_i) \leq \prod_{j\neq i}\left[\int dx_j\, q(x_j|x_i)\varepsilon_j(x_j)^{n-1}\right]^{1/(n-1)} \equiv c_i^{\text{upper}}(x_i)\,.$$

This upper bound will in many cases be useless because the integral often does not exist. The lower bound, which corresponds to a mean-field-type approximation, does not have this problem, but may still be somewhat conservative. We therefore propose the general family of approximations

$$c_i^{(\alpha)}(x_i) = \prod_{j\neq i}\left[\int dx_j\, q(x_j|x_i)\varepsilon_j(x_j)^{\alpha}\right]^{1/\alpha}.$$

It is easy to show that

$$c_i^{\text{lower}}(x_i) \leq c_i^{(\alpha)}(x_i) \leq c_i^{\text{upper}}(x_i) \quad \forall\, 0 \leq \alpha \leq n-1\,,$$

where $\alpha = 0$ is interpreted as the limit $\alpha \to 0$. Furthermore, for any α we obtain exactly the same Taylor expansion in terms of $\varepsilon_j(x_j) - 1$ (see Opper et al., 2009 and Section 4.3 below). The most sensible choice seems to be $\alpha = 1$, because it gives exact results when $n = 2$ as well as in the case when all x_js are indeed conditionally independent given x_i. We refer to the corresponding approximation as $\tilde{p}_i^{\text{EP-FACT}}(x_i)$. Note that when EP converges, this approximation always exists, because $q(x_j|x_i)\varepsilon_j(x_j)$ is proportional to the conditional marginal of the so called *tilted* distributions $t_j(x_j)\tilde{t}_j(x_j)^{-1}q(\boldsymbol{x})$.

Using (14), it is easy to see that $\tilde{p}_i^{\text{EP-FACT}}(x_i)$ corresponds to $\tilde{p}_i^{\text{EP-1STEP}}(x_i)$ if in (4.1.1) we would replace $q(\boldsymbol{x}_{\setminus i}|x_i)$ by the factorization $\prod_{j\neq i} q(x_j|x_i)$, that is, as if the variables x_j in the global Gaussian approximation are conditionally independent given x_i. A similar replacement in the Laplace approximation yields the approximation referred to as $\tilde{p}_i^{\text{LA-FACT}}(x_i)$. Here, we compute the univariate integrals with the Laplace method and using the approximation $x_j^*(x_i) \approx \mathrm{E}_q[x_j|x_i]$, with $q(\boldsymbol{x})$ being the global approximation resulting from the Laplace method.

The factorization principle can be applied to groups of variables $\boldsymbol{x}_I$ by factorizing $q(\boldsymbol{x}_{\setminus I}|\boldsymbol{x}_I)$. Another way to make use of the factorization is by applying it recursively. In this way, we can obtain higher order corrections of the approximate marginals and the evidence approximation. We will detail these methods in a future report.

An advantage of the bounding arguments is that we can extend the factorized approximation to cases when t_j depends on more variables, say, $\boldsymbol{x}_{I_j}$, with $I_j \in \{1,\ldots,n\}$. In this case, the factorization is unfeasible since $\prod_j t_j(\boldsymbol{x}_{I_j})$ may not factorize w.r.t. x_j. By using the bounding argument (Minka, 2005), we can still compute a "factorized" approximation

$$c_i^{(\alpha)}(x_i) = \prod_{j\neq i}\left[\int d\boldsymbol{x}_{I_j}\, q(\boldsymbol{x}_{I_j}|x_i)\varepsilon_j(\boldsymbol{x}_{I_j})^{\alpha}\right]^{1/\alpha}.$$

An example illustrating this idea is the logistic regression model presented in Section 4.5.

4.3 Connection to the Taylor Expansion in Opper et al. (2009)

The line of argument in Opper et al. (2009) when applied to approximating the marginals can be explained in our notation as follows. By expanding $p(\boldsymbol{x}) = Z_q Z_p^{-1} q(\boldsymbol{x}) \prod_j \varepsilon_j(x_j)$ in first order w.r.t. all $\varepsilon_j(x_j) - 1$, they obtain a first order approximation of the exact p in terms of the global approximation q and the tilted distributions $t_j(x_j)q^{\setminus i}(\boldsymbol{x})$. The marginalization of this expansion yields the marginal approximation

$$\tilde{p}_i^{\text{EP-OPW}}(x_i) \equiv \frac{Z_q}{Z_p} q(x_i) \left[1 + \sum_j \int dx_j q(x_j|x_i)[\varepsilon_j(x_j) - 1] \right].$$

Since the goal of Opper et al. (2009) was to provide improved approximations of the posterior distribution $p(\boldsymbol{x})$, and not only of its marginals, a natural adaptation of their approach would be to expand w.r.t. to all $j \neq i$ and not i itself. This leads to the approximation

$$p(x_i) \approx q(x_i)\varepsilon_i(x_i) \left[1 + \sum_{j \neq i} \int dx_j q(x_j|x_i)[\varepsilon_j(x_j) - 1] \right],$$

which is also the first order expansion of $\tilde{p}_i^{\text{EP-FACT}}(x_i)$ w.r.t. $\varepsilon_j(x_j) - 1$, $j \neq i$. A further expansion w.r.t $\varepsilon_i(x_i) - 1$ leads to $\tilde{p}_i^{\text{EP-OPW}}(x_i)$, thus the two approximations are equal in first order. An advantage of $\tilde{p}_i^{\text{EP-FACT}}(x_i)$ is that it is non-negative by construction, while $\tilde{p}_i^{\text{EP-OPW}}(x_i)$ can take on negative values.

4.4 Approximating Predictive Densities in Gaussian Process models

In many real-world problems, the prior $p_0(\boldsymbol{x})$ is defined as a Gaussian process—most often in terms of moment parameters—and besides marginals, one is also interested in computing accurate approximations of the predictive densities

$$p(\boldsymbol{x}_*|\boldsymbol{y}) = Z_p^{-1} \int d\boldsymbol{x}\, p_0(\boldsymbol{x}_*|\boldsymbol{x}) p_0(\boldsymbol{x}) \prod_j t_j(x_j),$$

where $\boldsymbol{x}_*$ is a set of latent variables of which distribution we want to approximate. By defining the $\hat{q}(\boldsymbol{x}, \boldsymbol{x}_*) \propto p_0(\boldsymbol{x}_*|\boldsymbol{x}) q(\boldsymbol{x})$ and using the same line of argument as in (12), one can derive similar approximations as EP-FACT or EP-1STEP. For example, $\tilde{p}^{\text{EP-FACT}}$ has the form

$$\tilde{p}^{\text{EP-FACT}}(\boldsymbol{x}_*) \propto \hat{q}(\boldsymbol{x}_*) \prod_j \int dx_j \hat{q}(x_i|\boldsymbol{x}_*)\varepsilon_j(x_j).$$

One can check that the marginalization and the conditioning of $\hat{q}$ leads to rank k updates, where k is the dimensionality of $\boldsymbol{x}_*$. For $k = 1$, the complexity $\tilde{p}^{\text{EP-FACT}}(x_*)$ roughly scales with the complexity of $\tilde{p}_i^{\text{EP-FACT}}(x_i)$.

4.5 Comparisons on Toy Models

In the following, we compare the performance of the marginal approximations on a few low-dimensional toy models; complex real-world models are considered in Section 6. For most of the models presented below, we use a prior p_0 with a symmetric covariance matrix

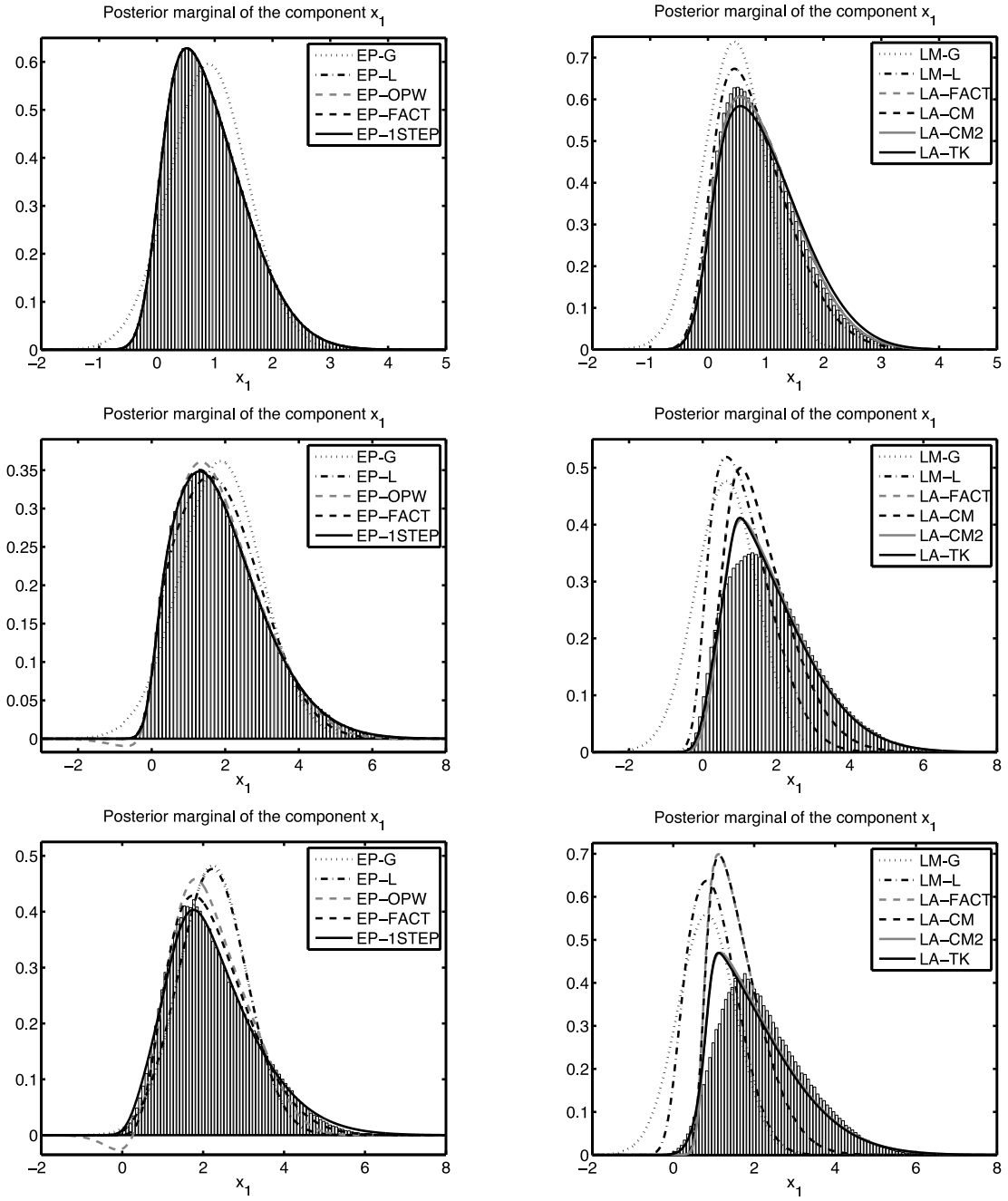


Figure 2: Various marginal corrections for a probit model with $t_i(x_i) = \Phi(4x_i)$ and identical variances and correlations in the prior p_0, using expectation propagation (left column) and Laplace approximations (right column). The panels show the corrections for a 3-dimensional model with prior variances and correlations $(v,c) = (1,0.25)$ (top), $(v,c) = (4,0.9)$ (center) and for a 32-dimensional model $(v,c) = (4,0.95)$ (bottom). Note how, the accuracy of the approximations decreases as the correlation, the prior variance and the dimension of the model increases.

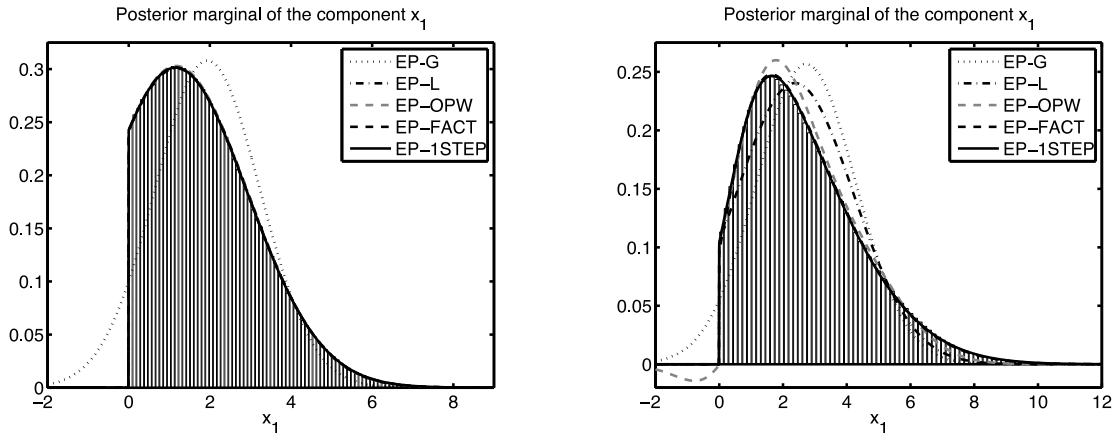


Figure 3: The posterior marginals of the first components of a 3-dimensional model with Heaviside terms with $(v,c) = (4,0.5)$ (left) and $(v,c) = (9,0.95)$ (right). The EP based approximations perform well even when the Laplace method is not applicable. The approximations have a similar behavior as in case of the probit model.

$\boldsymbol{V} = v\left[(1-c)\boldsymbol{I} + c\mathbf{1}\mathbf{1}^T\right]$, where we vary the variance v and the correlation c. We have chosen the models below, because they are often used in practice, and they lead to sufficiently non-Gaussian posterior marginals.

Probit terms. The terms t_j are defined as $t_j(x_j) = \Phi(y_j x_j)$, where Φ is the standard Gaussian cumulative density function. This choice of terms is typically made in binary classification models, where $y_j \in \{-1,1\}$. In order to obtain skewed marginals, in this example we set $y_j = 4$. The top and center panels in Figure 2 show the marginal corrections of the first component for a three-dimensional model with $(v,c) = (1,0.25)$ and $(v,c) = (4,0.9)$, respectively. The bars, in this and all other figures, correspond to a large number of Monte Carlo samples, either obtained through Gibbs or Metropolis sampling, and are supposed to represent the gold standard. The local correction EP-L yields sufficiently accurate approximations when the correlations are weak (top), but is clearly insufficient when they are strong (center). The corrections EP-1STEP and EP-FACT yield accurate estimates and are almost indistinguishable even for strong prior correlations. Only when we increase the number of dimensions (here from 3 to 32) and use strong prior correlations with moderate prior variances $(v,c) = (4,0.95)$, we can see small differences (top-right). As we can see in Figure 2, EP-OPW performs slightly worse than EP-FACT and can indeed turn negative.

It is known that the Laplace method does not perform well on this model (e.g., Kuss and Rasmussen, 2005). The approximations it yields tend to be acceptable for weak correlations (top), with LA-CM and LA-FACT clearly outperforming LM-G and LM-L, but are far off when the correlations are stronger (center, bottom). These corrections suffer from essentially the same problems as the global Gaussian approximation based on Laplace's method: the mode and the inverse Hessian represent the mean and the covariance badly and fail to sufficiently improve it. It is interesting to see that LA-CM2 can be almost as accurate as LA-TK, while its computational complexity scales with LA-CM. The examples suggest that, at least in case of this model, LA-CM2 has the best accuracy/complexity tradeoff when compared to LA-CM and LA-TK.

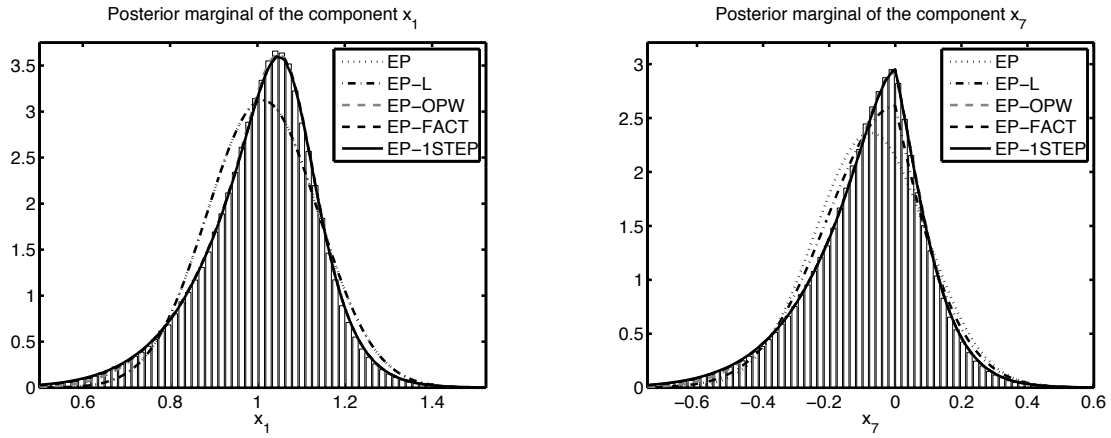


Figure 4: The posterior densities of a non-zero and a zero coefficient in a toy linear regression model with double exponential prior on the coefficients. It is interesting to compare the effects of the double exponential prior terms centered a zero on the quality of the local approximation EP-L. The effect is insignificant in the case the non-zero coefficient while in the case of the zero coefficient it has a strong effect, but the EP-L might still be quite inaccurate. We considered $n = 8$ coefficients the first two being 1 and the rest 0 and we generated $m = 8$ observables according to the model.

Step-function terms. Expectation propagation can still be applied when the Laplace method is not applicable. One such example is when the terms t_j are defined as $t_j(x_j) = \Theta(y_j x_j)$, where Θ is the step-function $\Theta(z) = \text{sign}(z)$ for $z \neq 0$ and $\Theta(0) = 1$. We chose $y_i = 1$. The plots on the left of Figure 3 show the marginals of the first component of a three dimensional model with $(v,c) = (4,0.5)$ (left) and $(v,c) = (9,0.95)$ (right). The performance of the approximations is similar to those of the previous model, except that in this case, we are dealing with discontinuous marginals.

Linear regression with sparsifying prior. Another model where the Laplace method is not applicable is the linear regression model with double exponential prior on the coefficients. We choose a model with $n = 8$ coefficients and $m = 8$ observations—m being close to n led to the most interesting posterior marginals. The elements of the design matrix $\boldsymbol{U}$ are sampled according to the standard normal density and renormalized such that every column vector has unit length. The regression coefficients are chosen as $\boldsymbol{x} = [1,1,0,\ldots,0]^T$ and the observations y_j are generated by $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{x} + \boldsymbol{\epsilon}$, where ε_j is normal with variance $v = 0.01$. We take zero centered independent double exponential priors on the x_j coefficients. The panels of Figure 4 show a few posterior marginals of the regression coefficients x_j given the maximum a posteriori (MAP) hyper-parameters v and λ. The priors on the hyper-parameters are taken as independent and log-uniform. The approximations are accurate but in this case, the local approximations EP-L fail dramatically when the mass of the distribution is not close to zero.

A logistic regression model. We can try to use EP-FACT to approximate the marginal probability densities even when the terms t_i, $i \in \{1,\ldots,m\}$ depend on more than one variable or a linear transformation of the variables. As an example, we define the terms as $t_i(\boldsymbol{x}) = \Phi(\boldsymbol{u}_i^T \boldsymbol{x})$. In this case, the factorization principle does not apply, but we can still use the line of argument in Section 4.2 and

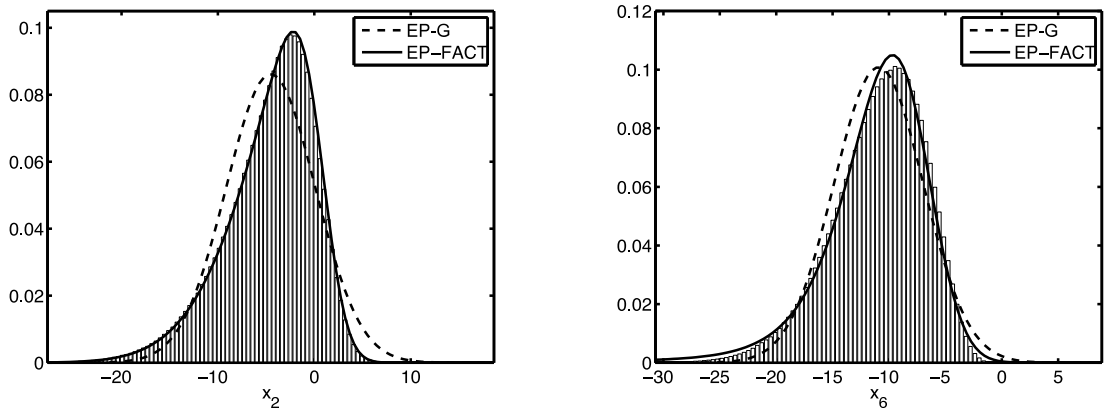


Figure 5: The posterior marginal approximation EP-FACT of the coefficients in a toy logistic regression model with Gaussian prior on the coefficients and moderate posterior correlations.. The panels show that even when the non-Gaussian terms depend on more than one variable and the posterior the approximation EP-FACT might still be accurate. We generated $n = 8$ coefficients and $m = 8$ observable variables.

evaluate how EP-FACT performs. The panels of Figure 5 show a few marginals of a model where we have chosen $u_i^j \sim \mathcal{N}(0, 10)$ and an independent Gaussian prior $p_0(\boldsymbol{x}) = \prod_j N(x_j|0, v^{-1})$ with $v = 0.01$. We used $n = 8$ and $m = 8$. Although one would expect that the factorization might lead to poor approximations, EP-FACT seems to approximate the marginals significantly better than the global approximation EP-G.

4.6 Computational Complexities of the Global Approximations in Sparse Gaussian Models

In this section, we review the computational complexities of the Laplace method and expectation propagation when applied to sparse Gaussian models, that is, models for which the n-dimensional precision matrix $\boldsymbol{Q}$ of the Gaussian prior is sparse. This is common in many practical applications in which the prior p_0 can be defined as a Gaussian Markov random field (e.g., van Gerven et al., 2009, 2010). We explore whether EP is indeed orders of magnitude slower, as suggested in Rue et al. (2009).

The computational complexity for both the (global) Laplace method and expectation propagation is dominated by several operations. 1) Computing the *Cholesky factor*, $\tilde{\boldsymbol{L}}$ of a matrix $\tilde{\boldsymbol{Q}}$, for example, corresponding to the posterior approximation $\tilde{p}^{\text{EP-G}}$ or $\tilde{p}^{\text{LM-G}}$, with the same sparsity structure as the prior precision matrix $\boldsymbol{Q}$. The computational complexity, denoted c_{chol}, scales typically with $\texttt{nnzeros}(\boldsymbol{Q})^2/n$, with $\texttt{nnzeros}(\boldsymbol{Q})$ being the number of non-zeros in the precision matrix $\boldsymbol{Q}$. 2) Computing the *diagonal elements of the inverse* of $\tilde{\boldsymbol{Q}}$. For sparse matrices, these can be computed efficiently by solving the Takahashi equations (Takahashi et al., 1973; Erisman and Tinney, 1975), which take the Cholesky factor $\tilde{\boldsymbol{L}}$ as input. A detailed description of solving the Takahasi equations can be found in Section A of the Appendix. The computational complexity, denoted c_{taka}, scales with n^3 in the worst case, but typically scales with $\texttt{nnzeros}(\boldsymbol{L})^2/n$. In practice, we experienced that it is significantly more expensive than the Cholesky factorization, possibly due to the additional

covariance values one has to compute during the process.[1] 3) Solving a *triangular system* of the form $\tilde{\boldsymbol{L}}\boldsymbol{a} = \boldsymbol{b}$, with corresponding computational complexity $c_{\text{tria}} \propto \texttt{nnzeros}(\boldsymbol{L})$.

The complexity of the latter two operations strongly depends on the number of non-zeros in the Cholesky factor, which should be kept to a minimum. There are various methods to achieve this by reordering the variables of the model. The approximate minimum degree reordering algorithm (Amestoy et al., 1996) seems to be the one with the best average performance (Ingram, 2006). Since the sparsity structure is fixed, the reordering algorithm has to be run only once, prior to running any other algorithm.

4.6.1 THE LAPLACE METHOD

To compute the global Gaussian approximation using the Laplace method, we first have to find the maximum a-posteriori solution. This can be done using, for example, the Newton method. Each Newton step requires one Cholesky factorization and solving two triangular systems. The off-diagonal elements of the posterior precision matrix $\tilde{\boldsymbol{Q}}$ are by construction equal to the off-diagonal elements of the prior precision matrix, so we only have to compute the n diagonal elements. To arrive at the lowest-order marginals $\tilde{p}_i^{\text{LM-G}}$ for all nodes i, we need the diagonal elements of the covariance matrix, the inverse of the precision matrix. These can be computed by solving the Takahashi equations, for which we can use the Cholesky factor computed in the last Newton step. Thus, computing the lowest order (Gaussian) marginals $\tilde{p}_i^{\text{LM-G}}$ for all variables x_i, $i = 1, \ldots, n$ by the Laplace method scales in total with $n_{\text{steps}}^{\text{Newton}} \times (c_{\text{chol}} + 2 \times c_{\text{tria}}) + c_{\text{taka}}$.

4.6.2 EXPECTATION PROPAGATION

In order to update a term approximation $\tilde{t}_j(x_j)$, we compute $q^{\backslash j}(x_j)$ using the marginals $q(x_j)$ from the current global approximation $q(\boldsymbol{x})$ and re-estimate the normalization constant and the first two moments of $t_j(x_j)\, q^{\backslash j}(x_j)$. In standard practice, the term approximations $\tilde{t}_j$ are updated sequentially and all marginal means and variances are recomputed using rank one updates after each term update. Instead, we adopt a parallel strategy, that is, we recompute marginal means and variances only after we have updated *all* term approximations $\tilde{t}_j$, $j = 1, \ldots, n$.

A parallel EP step consists of: 1) compute the Cholesky factorization of the current precision matrix, 2) solve two triangular systems to compute the current posterior mean and solve the Takahashi equations to compute the diagonal elements of the covariance matrix, and 3) if necessary, use univariate Gauss-Hermite numerical quadrature with n_{quad} nodes to compute the moments of $\varepsilon_j(x_j) q(x_j)$ for all $j = 1, \ldots, n$. This adds up to a computational complexity that scales with $n_{\text{steps}}^{\text{EP}} \times \big(c_{\text{chol}} + 2 \times c_{\text{tria}} + c_{\text{taka}} + n \times n_{\text{quad}}\big)$. After convergence, EP yields the lowest order marginals $\tilde{p}_i^{\text{EP-G}}$ for all variables x_i, $i = 1, \ldots, n$.

Because of the parallel schedule, we can make use of exactly the same computational tricks as with the Laplace method (Cholesky, Takahashi). Since solving the Takahashi equations for large n dominates all other operations, the main difference between the Laplace method and EP is that for EP we have to solve these equations a number of times, namely the number of EP steps, yet for Laplace only once. Initializing the term approximations in EP to the terms obtained by the Laplace method and then performing a few EP steps to obtain better estimates of the probability mass, makes EP just a (small) constant factor slower than Laplace. For efficient sequential updating

1. We used the MATLAB implementation of the sparse Cholesky factorization and a C implementation for solving the Takahashi equations.

steps \ methods	LA-CM	LA-FACT	EP-1STEP	EP-FACT
$q(x_j\|x_i)$	$c_{\text{tria}} + n \times n_{\text{grid}}$	$\boxed{c_{\text{tria}} + n \times n_{\text{grid}}}$	$c_{\text{tria}} + n \times n_{\text{grid}}$	$c_{\text{tria}} + n \times n_{\text{grid}}$
$\tilde{\varepsilon}(x_j;x_i)$	$n \times n_{\text{grid}}$	$n \times n_{\text{grid}}$	$n \times n_{\text{grid}} \times n_{\text{quad}}$	$\boxed{n \times n_{\text{grid}} \times n_{\text{quad}}}$
Norm. or det.-s	$\boxed{c_{\text{chol}} \times n_{\text{grid}}}$	$n \times n_{\text{grid}}$	$\boxed{c_{\text{chol}} \times n_{\text{grid}}}$	$n \times n_{\text{grid}}$

Table 1: Computational complexities of the steps for computing an improved marginal approximation for a particular node i using the various methods. The frames highlight the complexities that typically dominate the computation time. c_{tria}, c_{chol}, and c_{taka} refer to solving a sparse triangular system, a Cholesky factorization, and Takahashi equations, respectively. n_{grid} refers to the number of grid points and n_{quad} to the number of Gauss-Hermite quadrature nodes for x_i.

of EP, we would need a fast one-rank Takahashi update (or something similar), which, to the best of our knowledge, does not exist yet.

It is interesting to realize that since for any $Q_{ij} \neq 0$ the Takahashi equations also provide $\left[\boldsymbol{Q}^{-1}\right]_{ij}$, we can run EP using the factors $t_{ij}(x_i,x_j) = t_i(x_i)^{1/n_i} t_j(x_j)^{1/n_j}$ where n_k is the number of neighbors of node k according to the adjacency matrix defined by the structure of $\boldsymbol{Q}$. This increases the amount of computation, but the approximation might be more accurate.

4.7 Computational Complexities of Marginal Approximations

After running the global approximation to obtain the lowest order approximation, we are left with some Gaussian $q(\boldsymbol{x})$ with known precision matrix, a corresponding Cholesky factor and single-node marginals $q(x_i)$. We now consider the complexity of computing a corrected marginal through the various methods for a single node i, using n_{grid} grid points (see the summary in Table 1).

The local corrections $\tilde{p}_i^{\text{LM-L}}$ and $\tilde{p}_i^{\text{EP-L}}$ we get more or less for free. All other correction methods require the computation of the conditional densities $q(x_j|x_i)$. The conditional variance is independent of x_i, the conditional mean is a linear function of x_i. Computing $q(x_j|x_i)$ at all grid points for each j then amounts to solving two sparse triangular systems and $(n-1) \times n_{\text{grid}}$ evaluations. To arrive at the term approximations $\tilde{\varepsilon}(x_j;x_i)$, we need to compute second order derivatives for the Laplace approximation and numerical quadratures for EP, which is about n_{quad} times more expensive. For LA-FACT, EP-OPW and EP-FACT, we then simply have to compute a product or sum of n normalization terms. For LA-CM and EP-1STEP, we need to compute the determinant of an $(n-1)$-dimensional sparse matrix, which costs a Cholesky factorization. For LA-CM2 an additional c_{tria} has to be added for each x_i.

5. Inference of the Hyper-parameters

Until now, we considered estimating single-node marginals conditioned upon the hyper-parameters. In this section, we consider the estimation of the posterior marginals that follow by integrating over the hyper-parameters. For this, we need the posterior density of the hyper-parameters given the observations, which is approximated by $\tilde{p}(\theta|\boldsymbol{y}) \propto \tilde{p}(\boldsymbol{y}|\theta)\, p(\theta)$, where $\tilde{p}(\boldsymbol{y}|\theta)$ is the evidence

approximation provided by the Laplace method or expectation propagation. For the moment we assume that the approximate posterior density of the hyper-parameters is unimodal.

We propose a slight modification of the method used by Rue et al. (2009). Their method explores the space of the hyper-parameters in the eigen-space corresponding to the modal configuration and can be described briefly as: (1) compute the modal configuration (μ, Σ) of $\log \tilde{p}(\theta|\boldsymbol{y})$, (2) starting from the mode μ, select a set of uniformly spaced nodes $\mathcal{X}_i$ along the scaled eigenvectors $\sqrt{\lambda_i}\boldsymbol{u}_i$— here $\Sigma = \boldsymbol{U}\Lambda\boldsymbol{U}^T$ —by thresholding at both ends according to $\log \tilde{p}(\mu|\boldsymbol{y}) - \log \tilde{p}\left(\mu + k_i\Delta\sqrt{\lambda_i}\boldsymbol{u}_i|\boldsymbol{y}\right) < \delta, k_i \in \mathbb{Z}$, and finally (3) use all hyper-parameters corresponding to the nodes of the product grid $\mathcal{X}_1 \times \ldots \times \mathcal{X}_d, d = \dim(\theta)$ and satisfying the latter thresholding condition, to perform numerical quadrature using the rectangle rule.

Since the computational bottleneck of the procedure is the evaluation of the approximate evidence, we propose to improve this method by selecting the nodes—step (2) from above—in a different way: we keep the thresholding condition but we do a breadth-first search with regard to $(k_1, \ldots, k_d)$ on the grid graph $\mathbb{Z}^d$. We start from the origin and the hyper-parameter values that do not satisfy the thresholding condition are not included in the set of nodes whose neighbors we search. This simple modification proves to be very economical, since when exploring the volume around the mode, only the hyper-parameters that form the boundary surface are explored, but not selected. Thus, the proportion of useless computational time is the ratio of surface to volume. Although the boundary nodes do not satisfy the thresholding conditions, we can still use them in the numerical procedure. The number of grid points to be evaluated grows exponentially, as it does for the method in Rue et al. (2009). The difference is that in our method it roughly grows proportional to the volume of a d-dimensional sphere, whereas in the case of the method in Rue et al. (2009) it relates to the (larger) volume of a d-dimensional cube. Figure 6 illustrates the methods on a two-dimensional example. When the posterior density is not unimodal then we suggest to use a d-dimensional uniformly spaced grid, that is, $\boldsymbol{\Sigma} = \boldsymbol{I}$ and choose a well suited $\boldsymbol{\mu}$ and threshold δ which allows the exploration of the most significant modes. Once the hyper-parameters $\{\theta_1, \ldots, \theta_m\}$ are selected, the integration of the corrected approximate marginals over the hyper-parameter's approximate posterior density can be written as

$$\tilde{p}(x_i|\boldsymbol{y}) = \frac{\sum_{j=1}^{m} \tilde{p}(x_i|\boldsymbol{y}, \theta_j)\, \tilde{p}(\theta_j|\boldsymbol{y})}{\sum_{j=1}^{m} \tilde{p}(\theta_j|\boldsymbol{y})},$$

implying that the proposed procedure is similar to a reasonably efficient sampling procedure.

6. Examples

As real-world examples, we chose four models: a stochastic volatility model (Zoeter and Heskes, 2005; Rue et al., 2009), a log Gaussian Cox process model (Rue et al., 2009), a Gaussian process binary classification model (Kuss and Rasmussen, 2005) and a ranking model (Birlutiu and Heskes, 2007). Our aim is to show that the EP based correction methods can be as accurate as the Laplace approximation based ones and given that we have a sparse Gaussian prior, EP can be considered as an alternative to the Laplace method even when the number of variables is of the order of tens of thousands.

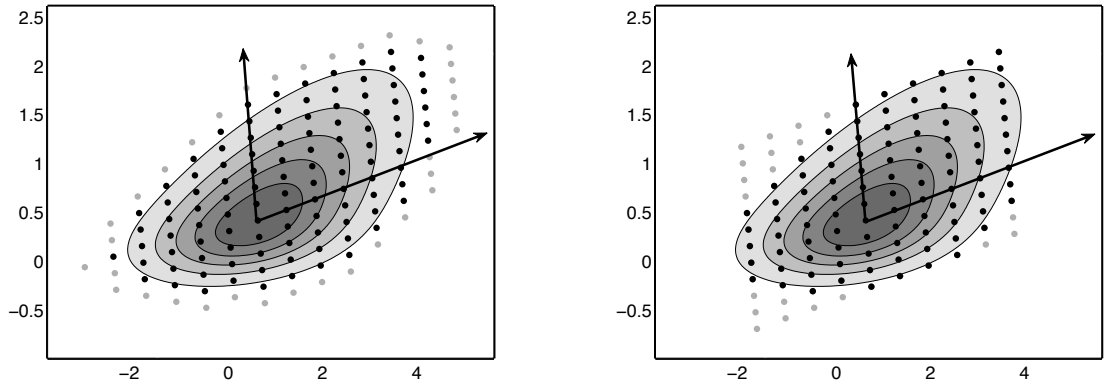


Figure 6: A comparison of the points selected by the thresholding breadth-first search procedure (left panel) and the method proposed by Rue et al. (2009) (right panel) when exploring in the eigen-space corresponding to the modal configuration. The black dots show the selected points while the gray ones stand for the ones that do not satisfy the thresholding condition. The principal axes on the figure are not perpendicular because of the different scaling of the axes. The number of evaluations in our method roughly grows proportional to the volume of a d-dimensional sphere, whereas the method of Rue et al. (2009) relates to the (larger) volume of a d-dimensional cube.

6.1 A Stochastic Volatility Model

As a first example for a sparse Gaussian model, we implemented the stochastic volatility model presented in Zoeter and Heskes (2005) where the authors used a sequential (global) EP algorithm to approximate the posterior density. The same model was used by Rue et al. (2009) to show that the global Laplace approximation is by magnitudes faster in sparse models than a sequential EP algorithm. They also showed that their marginal approximations work well on this model.

The data set consists of 945 samples of the daily difference of the pound-dollar exchange rate from October 1^{st}, 1981, to June 28^{th}, 1995. The observations y_t given the latent variables η_t are taken to be distributed independently according to $p(y_t|\eta_t) = N(y_t|0,e^{\eta_t})$. The quantity η_t governing the volatility is a linear predictor defined to be the sum $\eta_t = f_t + \mu$ of a first-order auto-regressive Gaussian process $p(f_t|f_{t-1},\phi,\tau) = N(f_t|\phi f_{t-1},1/\tau)$, with $|\phi| < 1$, and an additional Gaussian bias term with a prior $\mu \sim N(\mu|0,1)$. Thus the prior on $(f_1,\ldots,f_T,\mu)$ is a sparse latent Gaussian field. The prior on the hyper-parameter τ is taken to be $p(\tau) = \Gamma(\tau|1,10)$ and a Gaussian prior $\mathcal{N}(0,3)$ is taken over $\phi' = \log((1+\phi)/(1-\phi))$.

The joint density of the stochastic volatility model is

$$p(\boldsymbol{y},\boldsymbol{f},\mu,\tau,\phi) = \prod_{t=1}^{T} N\left(y_t|0,e^{f_t+\mu}\right) N(f_1|0,1) \prod_{t=2}^{T} N(f_t|\phi f_{t-1},1/\tau)$$
$$\times N(\mu|0,1)\,\Gamma(\tau|1,10)\,N\left(\log\left(\frac{1+\phi}{1-\phi}\right)|0,3\right)\left(\frac{2}{1-\phi^2}\right),$$

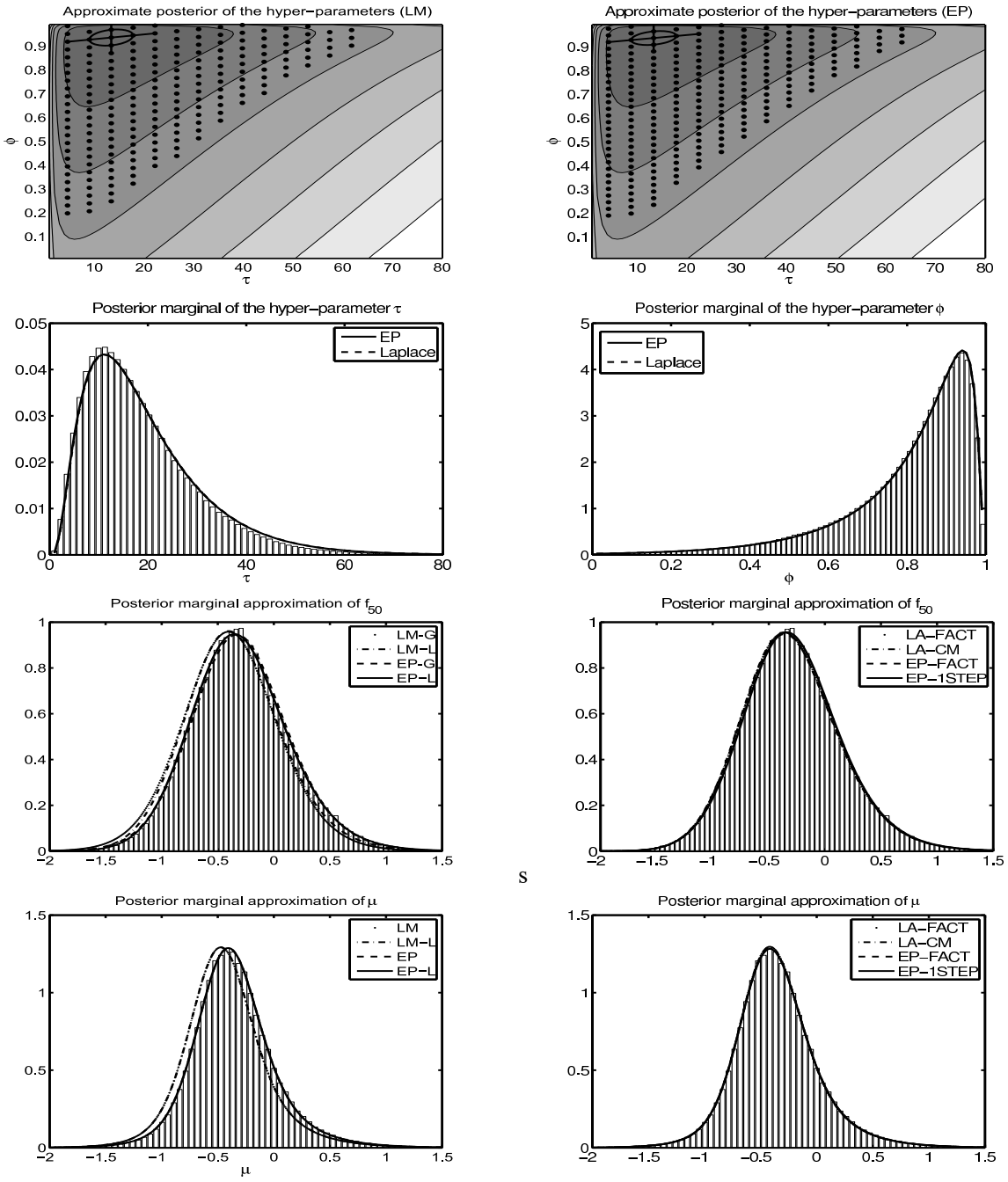


Figure 7: Plots of the posterior densities in the stochastic volatility model in Section 6.1. Figure panels show the logarithm of the approximate posterior density of the hyper-parameters using EP (top-right) and the Laplace method (top-left), their marginals (second row) and the posterior marginal approximations of f_{50} and μ (bottom rows) when integrated over the corresponding approximations of the hyper-parameters' posterior density. Dots show the hyper-parameters used for numerical integration; ellipses visualize the Hessian at the approximate posterior density's mode. The rest of the panels show the posterior density approximations of f_{50} and μ.

where $\Gamma(\cdot|k,\theta)$ denotes the Gamma density with mean value $k\theta$. Rue et al. (2009) propose to use the first 50 observations, both because of using the whole data set makes the approximation problem easier and because of comparison to Zoeter and Heskes (2005). For comparison, we used the same number of observations.

The results are shown in Figure 7. The Laplace and EP approximation of the evidence are nearly indistinguishable (top-row), as are the posterior marginals of the hyper-parameters (second row). Here EP is around a factor 5 slower than Laplace. The posterior marginals of f_{50} and μ obtained using the more involved methods (bottom rows) are practically indistinguishable from each other and the gold (sampling) standard. This is not the case for the cheaper variants LM-G, EP-G, and LM-L, but *is* the case for EP-L (third row): apparently to obtain excellent posterior marginals on this model, there is no need for (computationally expensive) corrections, but it suffices to compute a single global EP approximation per hyper-parameter setting and correct this for the (non-Gaussian) local term.

6.2 A log-Gaussian Cox Process Model

As a large sized example, we implemented the Laplace approximation and expectation propagation for the log-Gaussian Cox process model applied to the tropical rainforest bio-diversity data as presented in Rue et al. (2009). The observational data used in Rue et al. (2009) is the number of trees y_{ij} form a certain species in a small rectangular rainforest area indexed by $i = 1,\ldots,201$ and $j = 1,\ldots,101$ with mean altitude a_{ij} and gradient g_{ij}. The data is modeled by a discretized Poisson point process in two dimensions and the log of the mean parameter η_{ij} is defined as a Gaussian field. This means that the observations y_{ij} are taken to be Poisson distributed with mean $w_{ij}e^{\eta_{ij}}$, where the parameters w_{ij} are proportional to the size of the area where y_{ij} is measured. Since Rue et al. (2009) consider rectangular areas of the equal size, in their model w_{ij} is constant.

The latent Gaussian field η_{ij} modeling the log of the mean is defined as

$$\eta_{ij} = \beta_a a_{ij} + \beta_g g_{ij} + \beta_0 + f_{ij}^{(s)} + f_{ij}^{(u)}$$

where a_{ij} and g_{ij} are scalar quantities specifying altitude and gradient data, β_a and β_g are the corresponding linear coefficients and β_0 is a bias parameter. The latent fields $\boldsymbol{f}^{(s)}$ and $\boldsymbol{f}^{(u)}$ are defined as follows: $\boldsymbol{f}^{(s)}$ is a second-order polynomial intrinsic Gaussian Markov random field with precision parameter e^{v_s} constructed to mimic a thin plate spline on a uniform two dimensional grid, while $\boldsymbol{f}^{(u)}$ is an independent field with $f_{ij}^{(u)} \sim \mathcal{N}(0, e^{-v_\eta})$ included to model the noise. The fields $\boldsymbol{f}^{(s)}$ and $\boldsymbol{f}^{(u)}$ are modeling the unobserved spatially structured or unstructured covariates. Independent wide priors $\mathcal{N}(0, v_\beta^{-1})$ are taken on β_a, β_g and β_0, with $v_\beta^{-1} = 10^3$, thus the field $\boldsymbol{f}^{(s)}$ explains the assumed a-priori correlation in $\boldsymbol{\eta}$. We worked with the data set used in the INLA software package (Martino and Rue, 2009). The data set contains the corresponding a_{ij}, g_{ij}, w_{ij} and y_{ij} for a grid size of 101×201. We also used the same modeling approach, that is, we have taken $(\eta^T, {\boldsymbol{f}^{(s)}}^T, \beta_a, \beta_g, \beta_0)^T$ as latent variable, thus having an inference problem of dimension 40605. The joint density of the

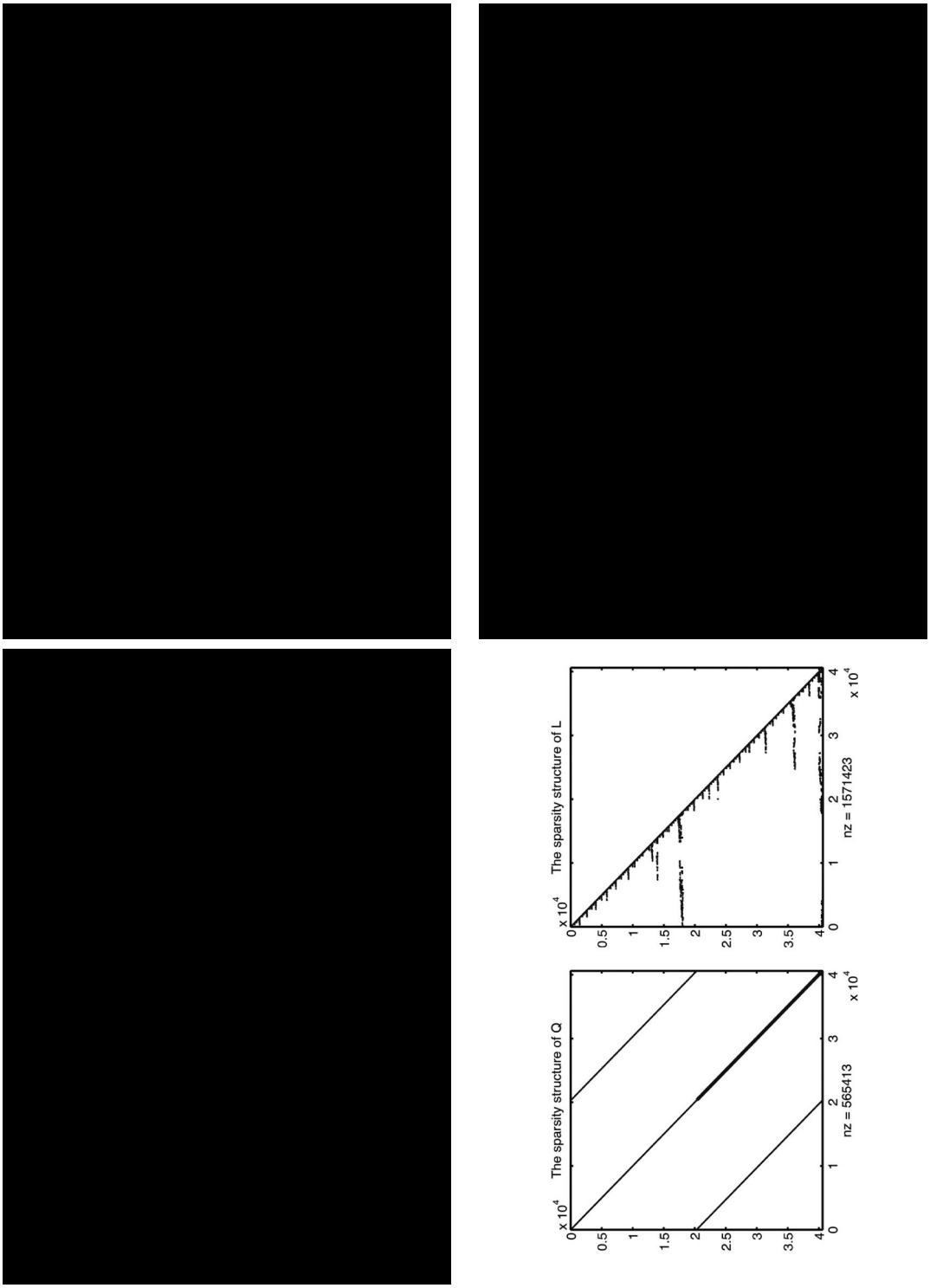


Figure 8: The panels show the altitude a_{ij}, gradient g_{ij} and the non-zero observation y_{ij} data for the log-Gaussian Cox process model in Section 6.2 together with the sparsity structure of $\boldsymbol{Q}$ and the Cholesky factor $\boldsymbol{L}$ of its approximate minimum degree reordering.

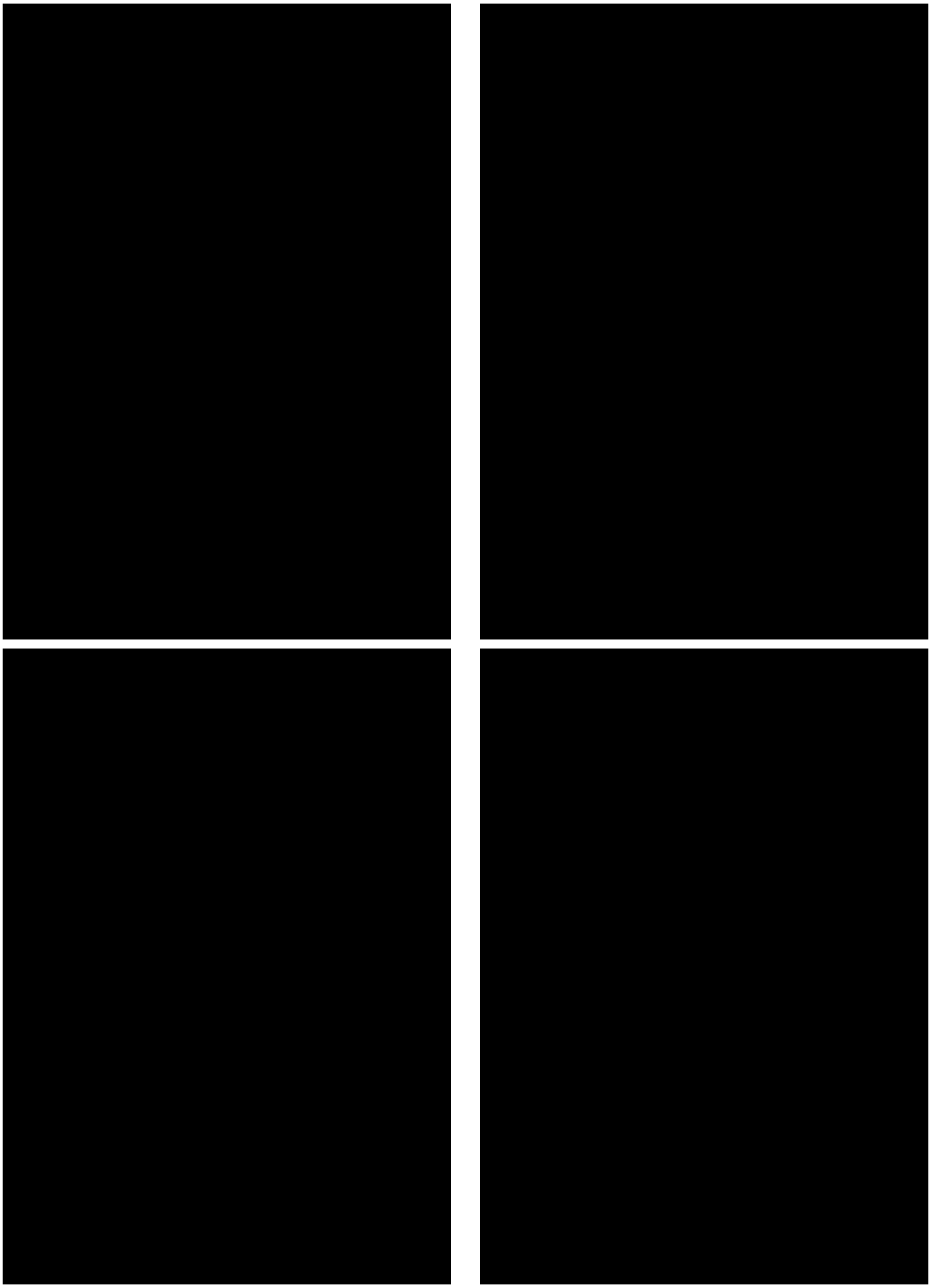


Figure 9: The approximate posterior mean and variance of the Gaussian random field $\boldsymbol{\eta}$ from the log-Gaussian Cox process model in Section 6.2. The top figures show the approximation obtained by the EP algorithm. The bottom panels show the comparison of the former to the approximation obtained by the Laplace method. The black contour curve in the bottom-left panel corresponds to the zero value.

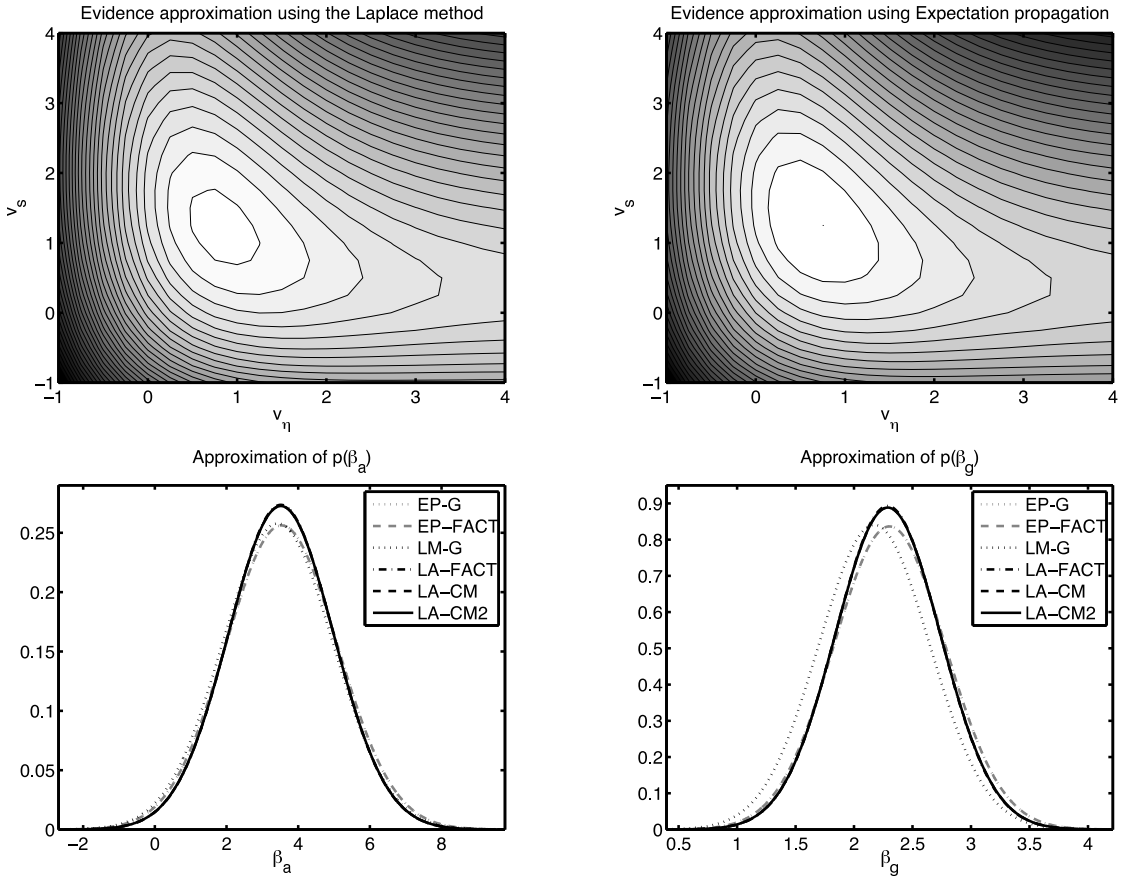


Figure 10: The posterior approximations of the evidence (top) and β_a and β_g (bottom). The Laplace method results in similar evidence estimates as EP (the level curves on the top panels show identical levels). The marginal approximations show marginals for the approximate MAP hyper-parameters.

log-Gaussian Cox process model is

$$
\begin{aligned}
p(\boldsymbol{y},\eta,\boldsymbol{f}^{(s)},\beta_a,\beta_g,\beta_0|v_\eta,v_s,\boldsymbol{a},\boldsymbol{g},\boldsymbol{w}) = \\
= \prod_{ij} \text{Poisson}\left(y_{ij}|w_{ij}e^{\eta_{ij}}\right) N\left(\eta_{ij}|f_{ij}^{(s)} + a_{ij}\beta_a + g_{ij}\beta_g + \beta_0, e^{-v_\eta}\right) \\
\times \left(\frac{v_s}{2\pi}\right)^{N/2} |\boldsymbol{S}|_*^{1/2} \exp\left\{-\frac{1}{2}e^{v_s}\boldsymbol{f}^{(s)^T}\boldsymbol{S}\boldsymbol{f}^{(s)}\right\} N\left(\beta_a,\beta_g,\beta_0|\boldsymbol{0},10^3\boldsymbol{I}\right),
\end{aligned}
$$

where $|\boldsymbol{S}|_*$ is the generalized determinant—an irrelevant constant—of the structure matrix $\boldsymbol{S}$ consisting of the finite difference coefficients of a second order improper polynomial Gaussian Markov random field on a uniform two dimensional grid—with the corresponding boundary conditions (Rue and Held, 2005). We used uninformative priors for v_η and v_s. The bottom-right panels of Figure 8 show the sparsity structure of the precision matrix $\boldsymbol{Q}$ corresponding to the Gaussian random vector $(\eta^T, \boldsymbol{f}^{(s)^T}, \beta_a, \beta_g, \beta_0)^T$ and the sparsity structure of its Cholesky factor $\boldsymbol{L}$ when $\boldsymbol{Q}$ is reordered with the AMD algorithm.

Expectation propagation was initialized using the term approximations corresponding to the Laplace method. Figure 8 shows the data we used and Figure 9 shows the mean values and standard deviations of the log intensity $\boldsymbol{\eta}$ when using the EP algorithm and the Laplace method with the hyper-parameter fixed to their corresponding approximate a posteriori (MAP) value.

The top panels of Figure 10 show the evidence approximations while the bottom panels show the marginal approximations for the corresponding MAP hyper-parameters. For β_a, there is a slight difference in variance between the Laplace approximation and the EP based methods, while for β_g, besides a similar effect, the approximation methods also improve on the mean of LM-G. It seems that EP-G is a sufficiently good approximation and EP-FACT does not really improve on it.

6.3 A Gaussian Process Model for Binary Classification

In this section we revisit and detail the probit model presented in Section 4.5. We use it in a binary classification problem with a Gaussian process prior on the latent variables. The data consists of the inputs $\boldsymbol{u}_j \in \mathbb{R}^d$, $j = 1,\ldots,n$ and the corresponding binary outputs $y_j \in \{-1,1\}$, $j = 1,\ldots,n$.

The model is defined as follows. The binary observables y_i are assumed to be Bernoulli distributed and conditionally independent given a set of latent variables $x_j \in \mathbb{R}$, $j = 1,\ldots,n$ which are controlling the parameters of the distribution through the cumulative density function Φ of the standard normal distribution, that is, $p(y_j|x_j) = \Phi(x_j)^{(1+y_j)/2}(1-\Phi(x_j))^{(1-y_j)/2}$. The latent variables $x_j = x(\boldsymbol{u}_j)$ are modeled as the values at locations $\boldsymbol{u}_i$ of a zero mean Gaussian process x with a (positive definite) covariance function $c(\cdot,\cdot)$. The joint density of the model can then be written as

$$
p(\boldsymbol{y},\boldsymbol{x}|\left\{\boldsymbol{u}_j\right\}_j, c) \propto \prod_j \Phi(y_j x_j) N(\boldsymbol{x}|\boldsymbol{0}, [c(\boldsymbol{u}_i,\boldsymbol{u}_k)]_{i,k}),
$$

where $[c(\boldsymbol{u}_i,\boldsymbol{u}_k)]_{i,k}$ denotes the matrix formed by the covariance values $c(\boldsymbol{u}_i,\boldsymbol{u}_k)$, $i,k = 1,\ldots,n$. We chose the Ionosphere[2] data set and a zero mean Gaussian process with covariance function $c(\boldsymbol{u}_i,\boldsymbol{u}_k) = \exp(a - e^v||\boldsymbol{u}_i - \boldsymbol{u}_k||^2)$. This setting yields a non-sparse precision matrix, therefore, the speed-up arguments do not apply. However, the parallel updating scheme is still applicable and it does not have higher complexity than the serial one.

2. The data set is publicly available at `http://archive.ics.uci.edu/ml/datasets/Ionosphere`.

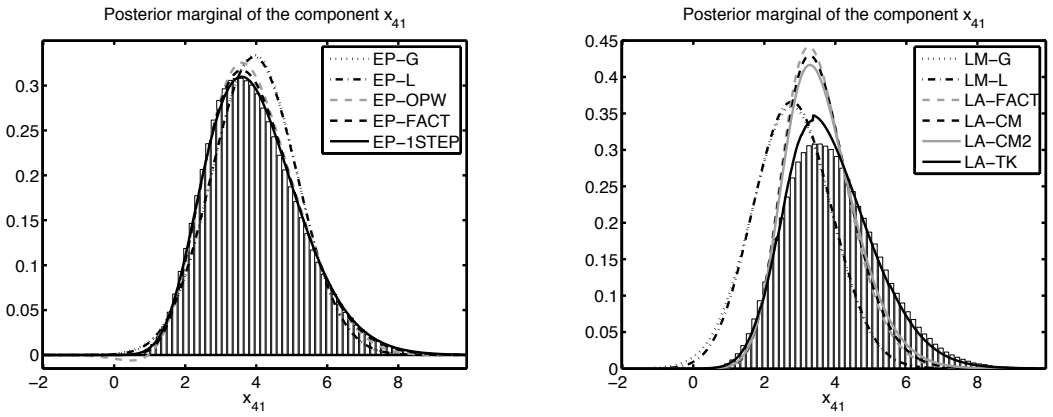


Figure 11: The approximate posterior marginal approximations of a Gaussian process binary classification model, with the hyper-parameters set to the approximate MAP values yielded by EP. The behavior of the marginals is similar to that in Figure 2, however, in this case, the correction provided by LA-CM2 is not that significant as in Figure 2.

Kuss and Rasmussen (2005) showed that on this model and data set EP leads to accurate approximations of the evidence while the Laplace method is substantially less accurate. We propose to illustrate how this behaviour manifests itself when approximating marginals. We use the whole set of $n = 351$ data points and compare the resulting marginals with the histograms obtained from 1.5×10^6 samples by using elliptical slice sampling (Murray et al., 2010). The hyper-parameters are set to the approximate MAP values obtained from EP's evidence approximation with uniform priors on a and v.

It turned out that many posterior marginal densities are skewed, however, most of the skewed marginals are well approximated by EP-L (the marginal of EP's tilted distribution). The panels of Figure 11 show the approximate posterior marginal densities of the latent variable with $j = 41$. These approximate posterior marginals exhibit a similar behavior like the ones in Figure 2.

6.4 A Ranking Model

To show that we can implement linear constraints with EP and that the factorization principle might work even in is cases when the non-Gaussian terms depend on more than one variable, we use a ranking model for rating players in sports competitions. The model is a simplified version of the models presented in Dangauthier et al. (2008) and Birlutiu and Heskes (2007) and we only consider it as an example to support the above mentioned claims. We assume that a player j is characterized by his/her strength which at time t is $x_t^{(j)}$. The prior on the evolution of the players' strength $\boldsymbol{x}_t = (x_t^{(1)}, \ldots, x_t^{(n)})$ is taken to be a factorizing $AR(1)$ model. Each game between two players is represented by the triple (i, j, t) and the collection of these triples is denoted by G. We assume that the outcomes of the games are a binary variables $y_{i,j,t} \in \{-1, 1\}$, the games are conditionally independent given the players strengths and the probability of player i winning the game against player j at time t is $\Phi(x_t^{(i)} - x_t^{(j)})$, where Φ is the standard normal cumulative density function. To

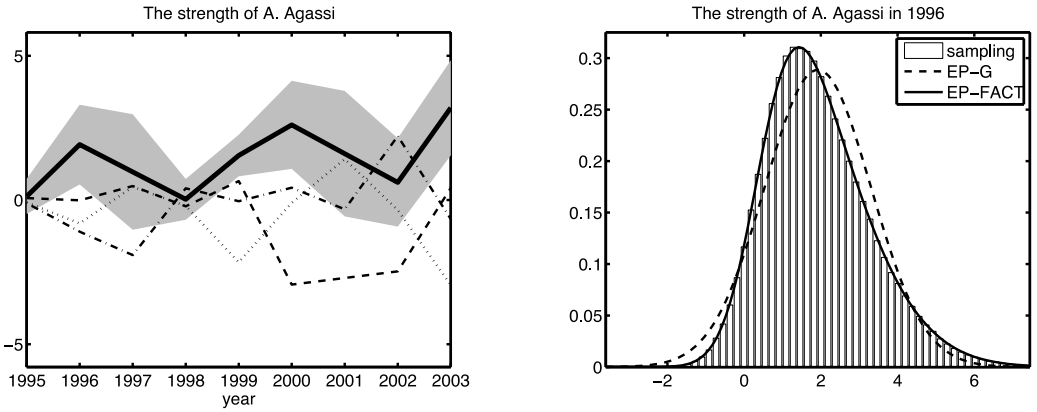


Figure 12: The left panel shows the mean strengths of the players A. Agassi (cont.), Y. Kafelnikov (dashed), C. Moya (dashed-dotted), and T. Henman (dotted) with the standard deviations of A. Agassi's strength based on the ranking model presented in Section 6.4. The data set consists of the games played by these players against each other in the years 1995-2003. We implemented linear constraints such that the players strength sum to zero in every year. The left panel shows that this indeed holds for the means. The right panel shows A. Agassi's strength distribution in 1996 which is a non-Gaussian density and can be well approximated using EP-FACT.

implement linear constraints, we constrain the players' strength to sum to zero at any given time t. These constraints are purely artificial and are only considered for illustration purposes.

The joint posterior density of the players' strength is given by

$$p(\boldsymbol{x}^1,\ldots,\boldsymbol{x}^T|\boldsymbol{y},v_1,v,a) \propto \prod_{t=1}^{T}\delta_0(\mathbf{1}^T\boldsymbol{x}_t)\prod_{(i,j,t)\in G}\Phi(y_{i,j,t}(x_t^{(i)}-x_t^{(j)})) \times\prod_{j=1}^{n}N(x^{(j)}|0,v_1)\prod_{t=1}^{T-1}N(x_{t+1}^{(j)}|ax_t^{(j)},v).$$

We approximate this density with a Gaussian density using EP and we use the factorized corrections EP-FACT, to improve on the Gaussian marginals. The prior on the players strengths is a sparse Gaussian Markov random field, thus we can apply the methods presented in Section 4.6.2.

We have chosen a data set consisting of four[3] tennis players and their ATP tournament games played against each other form 1995 to 2003. There was a total of 45 games. To obtain reasonably skewed marginals, we chose $v_1 = 1, a = 1$ and $v = 9$. The left panel in Figure 12 shows the evolution of the players' mean strengths and the corresponding standard deviations for the best player. Note that the players' mean strengths average to zero at all times. The right panel shows that the factorized approximations EP-FACT, can indeed improve on the Gaussian marginal approximations computed by EP even in models where non-Gaussian terms depend on more than one variable. This might be due to the relatively sparse interaction between the variables $x_t^{(j)}, t = 1,\ldots,T,\ j = 1,\ldots,n$.

3. We have chosen A. Agassi, Y. Kafelnikov, C. Moya and T Henman.

7. Discussion

We introduced several methods to improve on the marginal approximations obtained by marginalizing the global approximations. The approximation denoted by EP-FACT seems to be, in most cases, both accurate and fast. An improvement in accuracy can be achieved with some additional computational cost by using EP-1STEP. We showed that by using a parallel EP scheduling the computational complexity of EP in sparse Gaussian model can scale with the computational complexity of the Laplace method.

There are many options for further improvement, in particular with respect to efficiency. The ideas behind the simplified Laplace approximation of Rue et al. (2009), which aims to prevent the expensive computation of a determinant for each x_i, are applicable to expectation propagation. However, if the computation of the determinant in EP-1STEP dominates the computation time, the factorized approximation EP-FACT may be a faster but less accurate alternative.

One of the main problems of expectation propagation is that it is not guaranteed to converge and may run into numerical problems. There were no problems with the convergence of EP in the problems considered in this paper, but even when there are, it can still be useful to start from the Laplace solution and perform a few EP steps to get closer to the main mass of the probability instead of relying on the mode and the curvature.

For models with weak correlations and smooth nonlinearities, any approximation method gives reasonably good results. However, it is possible to come up with cases (strong correlations, hard nonlinearities), where any deterministic approximation method fails. The most interesting problems are somewhere in between, and for those we can hardly tell how advanced and computationally intensive an approximation method we need. The heuristic suggested in Rue et al. (2009), to systematically increase the complexity and stop when no further changes can be obtained, appears to be risky. In particular when going from the factorized to the non-factorized approximations, it is often hard to see changes, but still both approximations can be far off. It would be interesting to obtain a better theoretical understanding of the (asymptotic) approximation errors implied by the different approaches.

Acknowledgments

This research was supported by VICI grant 639.023.604 from the Netherlands Organization for Scientific Research (NWO). We would like to thank Håvard Rue and the anonymous reviewers for their valuable comments on earlier versions of the manuscript.

Appendix A. Solving the Takahashi Equations

The Takahashi equations (Takahashi et al., 1973) aim to compute certain elements of the inverse of a positive definite matrix from its Cholesky factor. The derivation of the equations or the algorithm can be found in many papers (e.g., Erisman and Tinney, 1975; Rue et al., 2009). In the following we present the line of arguments in Rue et al. (2009). Let $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, $\boldsymbol{z} \sim N(\boldsymbol{0}, \boldsymbol{I})$ and $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$. Then using the notation $\boldsymbol{V} = \boldsymbol{Q}^{-1}$ we find that $\boldsymbol{x} \sim N(\boldsymbol{0}, \boldsymbol{V})$. The equations $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$ can be rewritten as $L_{ii}x_i = z_i - L_{ii}^{-1}\sum_{k=i+1}^{n} L_{ki}x_k$. Multiplying both sides with x_j, $j \geq n$, using $\boldsymbol{z} = \boldsymbol{L}^{-T}\boldsymbol{x}$ and taking expectations we arrive at the Takahashi equations $V_{ij} = \delta_{ij}L_{ii}^{-2} - L_{ii}^{-1}\sum_{k=i+1}^{n} L_{ki}V_{kj}$. Since we only

want to compute the diagonal of $\boldsymbol{V}$ or the elements V_{ij} for which $L_{ij} \neq 0$, the algorithm can be written in the following MATLAB friendly form

1: **function** $\boldsymbol{V}$ = `SolveTakahashi`($\boldsymbol{L}$)
2: **for** $i = n : -1 : 1$
3: $I = \{ j : L_{ij} \neq 0,\ j > i \}$
4: $\boldsymbol{V}_{I,i} = -\boldsymbol{V}_{I,I}\boldsymbol{L}_{I,i}/L_{i,i}$
5: $\boldsymbol{V}_{i,I} = \boldsymbol{V}_{I,i}^T$
6: $V_{i,i} = 1/L_{i,i}^2 - \boldsymbol{V}_{i,I}\boldsymbol{L}_{I,i}/L_{ii}$
7: **end**

The complexity of this algorithm scales with `nonzeros`$(\boldsymbol{Q})^2/n$.

Appendix B. Gaussian Formulas

The first and second moments of a distribution $p(\boldsymbol{x}) = Z^{-1}(\boldsymbol{m}, \boldsymbol{V}) f(\boldsymbol{x}) q(\boldsymbol{x})$ with $q(\boldsymbol{x}) = N(\boldsymbol{x}|\boldsymbol{m}, \boldsymbol{V})$ are given by

$$\begin{aligned} \mathrm{E}_p[\boldsymbol{x}] &= \boldsymbol{m} + \boldsymbol{V}\nabla_{\boldsymbol{m}} \log Z(\boldsymbol{m}, \boldsymbol{V}), \\ \mathrm{V}_p[\boldsymbol{x}] &= \boldsymbol{V} + \boldsymbol{V}\nabla^2_{\boldsymbol{mm}} \log Z(\boldsymbol{m}, \boldsymbol{V})\boldsymbol{V}. \end{aligned}$$

Applying integration by parts, one can show that the moments of p can also be written in the form

$$\begin{aligned} \mathrm{E}_p[\boldsymbol{x}] &= \boldsymbol{m} + \frac{1}{Z}\boldsymbol{V}\mathrm{E}_q[\nabla_{\boldsymbol{x}} f], \\ \mathrm{V}_p[\boldsymbol{x}] &= \boldsymbol{V} + \frac{1}{Z^2}\boldsymbol{V}\left[Z\mathrm{E}_q\left[\nabla^2_{\boldsymbol{xx}} f\right] - \mathrm{E}_q[\nabla_{\boldsymbol{x}} f]\,\mathrm{E}_q[\nabla_{\boldsymbol{x}} f]^T\right]\boldsymbol{V}, \end{aligned}$$

provided that $f(\boldsymbol{x})\, e^{-\boldsymbol{x}^T\boldsymbol{x}}$ and $\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} e^{-\boldsymbol{x}^T\boldsymbol{x}}$ vanish at infinity and the required differentials and integrals exist.

Appendix C. Details of EP in Latent Gaussian Models

Assume the distribution has the form

$$p(\boldsymbol{x}) \propto p_0(\boldsymbol{x}) \prod_i t_i(\boldsymbol{U}_i \boldsymbol{x}),$$

where $\boldsymbol{U}_i$ are linear transformations. This formulation includes both the representations when t_j depend only on a subset of parameters, that is, $t_i(\boldsymbol{x}) = t_i(\boldsymbol{x}_{I_i})$ with $\boldsymbol{U}_i = \boldsymbol{I}_{\cdot,I_i}$ and the representation used in logistic regression, where $\boldsymbol{U}_i$ is the i^{th} row of the design matrix. Here we present the details of the α-fractional or power EP where the updates are performed on $\tilde{t}_i^{\alpha}(\boldsymbol{x})$.

C.1 Computing $\tilde{t}_i^{new}$

First we compute the form of the term approximations, and show that $\tilde{t}_i$ has a low rank representation. Let $q(\boldsymbol{x}) = N(\boldsymbol{x}|\boldsymbol{m}, \boldsymbol{V})$ and let $\tilde{\boldsymbol{h}} = \boldsymbol{V}^{-1}\boldsymbol{m}$, $\tilde{\boldsymbol{Q}} = \boldsymbol{V}^{-1}$ the canonical parameters of $q(\boldsymbol{x})$. We use $q^{\setminus i}(\boldsymbol{x}) = N\left(\boldsymbol{x}|\boldsymbol{m}^{\setminus i}, \boldsymbol{V}^{\setminus i}\right)$ to denote the distribution $q^{\setminus i}(\boldsymbol{x}) \propto q(\boldsymbol{x})/\tilde{t}_i^{\alpha}(\boldsymbol{x})$. After

some calculus one can show that the moment matching Gaussian $q^{new}(\boldsymbol{x}) = N(\boldsymbol{x}|\boldsymbol{m}^{new}, \boldsymbol{V}^{new})$ of $q_i(\boldsymbol{x}) \propto t_i^{\alpha}(\boldsymbol{x}) q^{\setminus i}(\boldsymbol{x})$ is given by

$$\begin{aligned}
\boldsymbol{m}^{new} &= \boldsymbol{m}^{\setminus i} + \boldsymbol{V}^{\setminus i}\boldsymbol{U}_i^T \left[\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right]^{-1} \left[\mathrm{E}\left[\boldsymbol{z}_i\right] - \boldsymbol{U}_i \boldsymbol{m}^{\setminus i}\right], \\
\boldsymbol{V}^{new} &= \boldsymbol{V}^{\setminus i} + \boldsymbol{V}^{\setminus i}\boldsymbol{U}_i^T \left[\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right]^{-1} \left[\mathrm{V}\left[\boldsymbol{z}_i\right] - \boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right] \left[\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right]^{-1} \boldsymbol{U}_i \boldsymbol{V}^{\setminus i},
\end{aligned}$$

where $\boldsymbol{z}_i$ is a random variable distributed as $\boldsymbol{z}_i \sim t(\boldsymbol{z}_i)^{\alpha} N\left(\boldsymbol{z}_i | \boldsymbol{U}_i \boldsymbol{m}^{\setminus i}, \boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right)$. The update for the term approximation $\tilde{t}_i(\boldsymbol{x})$ is given by $\left(\tilde{t}_i^{new}(\boldsymbol{x})\right)^{\alpha} \propto q^{new}(\boldsymbol{x}) / q^{\setminus i}(\boldsymbol{x})$. The latter division yields

$$\left[\boldsymbol{V}^{new}\right]^{-1} - \left[\boldsymbol{V}^{\setminus i}\right]^{-1} = \boldsymbol{U}_i^T \left[\mathrm{V}\left[\boldsymbol{z}_i\right]^{-1} - \left[\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right]^{-1}\right] \boldsymbol{U}_i, \tag{15}$$

$$\left[\boldsymbol{V}^{new}\right]^{-1} \boldsymbol{m}^{new} - \left[\boldsymbol{V}^{\setminus i}\right]^{-1} \boldsymbol{m}^{\setminus i} = \boldsymbol{U}_i^T \left[\mathrm{V}\left[\boldsymbol{z}_i\right]^{-1} \mathrm{E}\left[\boldsymbol{z}_i\right] - \left[\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T\right]^{-1} \boldsymbol{U}_i \boldsymbol{m}^{\setminus i}\right] \tag{16}$$

leading to

$$\tilde{t}_i^{new}(\boldsymbol{x}) \propto \exp\left((\boldsymbol{U}_j \boldsymbol{x})^T \tilde{\boldsymbol{h}}^j - \frac{1}{2} (\boldsymbol{U}_j \boldsymbol{x})^T \tilde{\boldsymbol{Q}}^j (\boldsymbol{U}_j \boldsymbol{x})\right),$$

where $\tilde{\boldsymbol{h}}^i$ and $\tilde{\boldsymbol{Q}}^i$ are given by the corresponding quantities in (15) and (16). The approximating distribution q is defined by the canonical parameters

$$\begin{aligned}
\tilde{\boldsymbol{h}} &= \boldsymbol{h} + \sum_i \boldsymbol{U}_i^T \tilde{\boldsymbol{h}}^i, \\
\tilde{\boldsymbol{Q}} &= \boldsymbol{Q} + \sum_i \boldsymbol{U}_i^T \tilde{\boldsymbol{Q}}^i \boldsymbol{U}_i,
\end{aligned}$$

that is, the sum over the parameters of $\tilde{t}_i$ and the parameters of the prior $p_0(\boldsymbol{x}) \propto \exp\left(\boldsymbol{h}^T \boldsymbol{x} - \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}/2\right)$.

C.2 Computing the Cavity Distribution $q^{\setminus i}$

Now, we turn our attention to the computation of the distribution $q^{\setminus i}$. The quantities we are interested in are $\boldsymbol{U}_i \boldsymbol{m}^{\setminus i}$ and $\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_j^T$. After some calculus, one can show that these are given by

$$\begin{aligned}
\boldsymbol{U}_i \boldsymbol{V}^{\setminus i} \boldsymbol{U}_i^T &= \boldsymbol{U}_i \left(\tilde{\boldsymbol{Q}} - \alpha \boldsymbol{U}_i^T \tilde{\boldsymbol{Q}}^i \boldsymbol{U}_i\right)^{-1} \boldsymbol{U}_i^T \\
&= \left(\boldsymbol{U}_i \boldsymbol{V} \boldsymbol{U}_i^T\right) \left(\boldsymbol{I} - \alpha \tilde{\boldsymbol{Q}}^i \left(\boldsymbol{U}_i \boldsymbol{V} \boldsymbol{U}_i^T\right)\right)^{-1}, \\
\boldsymbol{U}_i \boldsymbol{m}^{\setminus i} &= \boldsymbol{U}_i \left(\tilde{\boldsymbol{Q}} - \alpha \boldsymbol{U}_i^T \tilde{\boldsymbol{Q}}^i \boldsymbol{U}_i\right)^{-1} \left(\tilde{\boldsymbol{h}} - \alpha \boldsymbol{U}_i^T \tilde{\boldsymbol{h}}^i\right) \\
&= \left(\boldsymbol{I} - \alpha \tilde{\boldsymbol{Q}}^i \left(\boldsymbol{U}_i \boldsymbol{V} \boldsymbol{U}_i^T\right)\right)^{-1} \left(\boldsymbol{U}_i \boldsymbol{m} - \alpha \left(\boldsymbol{U}_i \boldsymbol{V} \boldsymbol{U}_i^T\right) \tilde{\boldsymbol{h}}^i\right).
\end{aligned}$$

Therefore, the computational bottleneck of EP reduces to the computation of the quantities $\boldsymbol{U}_i \boldsymbol{m}$ and $\boldsymbol{U}_i \boldsymbol{V} \boldsymbol{U}_i^T$. These can be computed from the canonical representation of q by $\boldsymbol{U}_i \tilde{\boldsymbol{Q}}^{-1} \boldsymbol{h}$ and $\boldsymbol{U}_i \tilde{\boldsymbol{Q}}^{-1} \boldsymbol{U}_i^T$.

C.3 Computing EP's Evidence Approximation

Let us define

$$\log Z(\boldsymbol{m},\boldsymbol{V}) \equiv \frac{1}{2}\boldsymbol{m}^T\boldsymbol{V}^{-1}\boldsymbol{m} + \frac{1}{2}\log\det\boldsymbol{V} + \frac{n}{2}\log(2\pi)$$

and

$$\log Z_i(\boldsymbol{m},\boldsymbol{V}) \equiv \log\int d\boldsymbol{x}\, N(\boldsymbol{x}|\boldsymbol{m},\boldsymbol{V})\, t_i^{\alpha}(\boldsymbol{U}_i\boldsymbol{x}).$$

Expectation propagation approximates the evidence $p(\boldsymbol{y}|\theta)$ by $Z_{ep} = Z^{1-n/\alpha}\prod_i Z_i^{\alpha}$. Using the above introduced notation this can be written as

$$\begin{aligned}\log Z_{EP} = &\log Z(\boldsymbol{m},\boldsymbol{V}) \\ &+ \frac{1}{\alpha}\sum_i\left[\log Z_j\left(\boldsymbol{m}^{\setminus i},\boldsymbol{V}^{\setminus i}\right) + \log Z\left(\boldsymbol{m}^{\setminus i},\boldsymbol{V}^{\setminus i}\right) - \log Z(\boldsymbol{m},\boldsymbol{V})\right],\end{aligned}$$

which in the case when t_i depends on $\boldsymbol{U}_i\boldsymbol{x}$ leads to

$$\begin{aligned}\log Z_{EP} = &\log Z(\boldsymbol{m},\boldsymbol{V}) + \frac{1}{\alpha}\sum_i \log Z_j\left(\boldsymbol{U}_i\boldsymbol{m}^{\setminus i},\boldsymbol{U}_i\boldsymbol{V}^{\setminus i}\boldsymbol{U}_i^T\right) \\ &+ \frac{1}{\alpha}\sum_i\left[\log Z\left(\boldsymbol{U}_i\boldsymbol{m}^{\setminus i},\boldsymbol{U}_i\boldsymbol{V}^{\setminus i}\boldsymbol{U}_i^T\right) - \log Z\left(\boldsymbol{U}_i\boldsymbol{m},\boldsymbol{U}_i\boldsymbol{V}\boldsymbol{U}_i^T\right)\right].\end{aligned}$$

Appendix D. A Summary of the Marginal Approximations

An explanatory list of the approximation methods in Figure 13.

- LA-TK. The Laplace approximation of Tierney and Kadane (1986). The approximation $\tilde{p}^{\text{LA-TK}}(x_i)$ is computed by using the Laplace method to approximate $c_i(x_i)$ (Section 3.1).
- EP-FULL. The full EP approximation of the marginal. This approximation is computed by using EP to approximate $c_i(x_i)$ (Section 4.1.1).
- EP-L. EP local. The approximation $\tilde{p}^{\text{EP-L}}(x_i) \propto \varepsilon_i(x_i)q(x_i)$ is obtained from $c_{x_i}(x_\rangle \approx 1$, where $\varepsilon_i(x_i) = t_i(x_i)/\tilde{t}_i(x_i)$ and $q(\boldsymbol{x})$ are computed by EP (Section 3).
- LM-L. Lapace method local. EP local. The approximation $\tilde{p}^{\text{EP-L}}(x_i) \propto \varepsilon_i(x_i)q(x_i)$ is obtained from $c_{x_i}(x_\rangle \approx 1$, where $\varepsilon_i(x_i) = t_i(x_i)/\tilde{t}_i(x_i)$ and $q(\boldsymbol{x})$ are computed by the Laplace method (Section 3). In this case $\log\varepsilon_i(x_i) = R_2[\log t_i](x_i)$.
- LA-CM. The Laplace approximation with the conditional mode approximated by the conditional mean. The approximation $\tilde{p}^{\text{LA-CM}}(x_i)$ is computed as proposed in Rue et al. (2009), that is, by using the approximation $\boldsymbol{x}^*_{\setminus i}(x_i) \approx \mathrm{E}_q\left[\boldsymbol{x}_{\setminus i}|x_i\right]$ where $q(\boldsymbol{x})$ is given by the Laplace method (Section 4.1.2).
- LA-CM2. The similar approximation as LA-CM, but with an additional term added to account for $\boldsymbol{x}^*_{\setminus i}(x_i) \approx \mathrm{E}_q\left[\boldsymbol{x}_{\setminus i}|x_i\right]$ (Section 4.1.2).

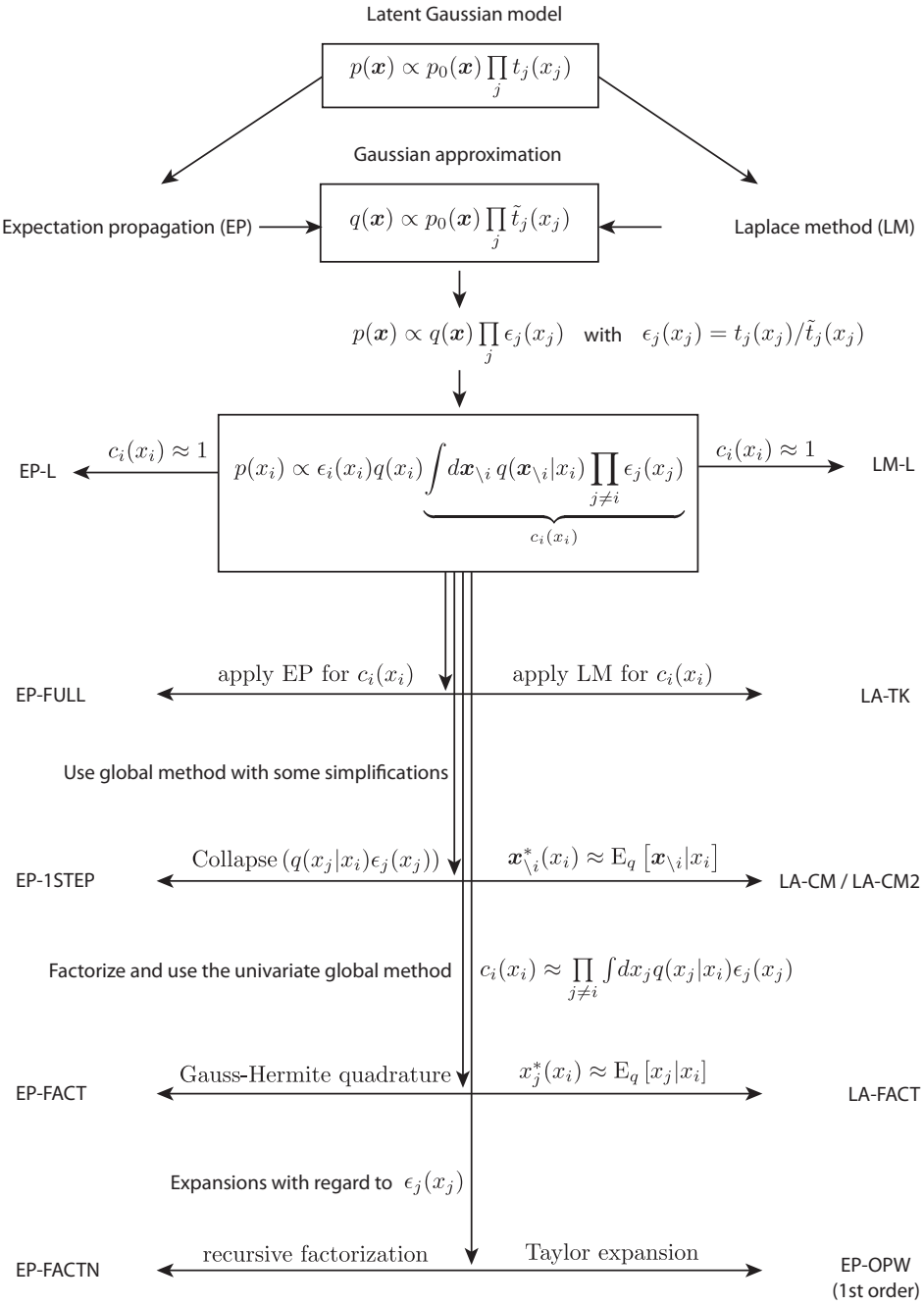


Figure 13: A schematic view of the approximation methods introduced or referred to in this paper. For details see Section D of the Appendix.

- EP-1STEP. The one step EP approximation. The approximation $\tilde{p}^{\text{EP-1STEP}}(x_i)$ is computed by defining $\tilde{\varepsilon}_j(x_j;x_i) \equiv \text{Collapse}(q(x_j|x_i)\varepsilon_j(x_j))/q(x_j|x_i)$ and using the approximation $c_i(x_i) \approx \int d\boldsymbol{x}_{\setminus i} q(\boldsymbol{x}_{\setminus i}|x_i)\prod_{j\neq i}\tilde{\varepsilon}_j(x_j;x_i)$ (see Section 4.1.1). This corresponds to one EP step for computing $c_i(x_i)$ with the initialization $\tilde{\varepsilon}_j(x_j;x_i) = 1$.

- EP-OPW. The Taylor expansion of Opper et al. (2009). The approximation $\tilde{p}^{\text{EP-OPW}}(x_i)$ is computed by expanding $p(\boldsymbol{x}) \propto p_0(\boldsymbol{x})\prod_j \varepsilon_j(x_j)$ in first order with regard to $\varepsilon_j(x_j) - 1$ for all $j = 1,\ldots,n$ and integrating with regard to $\boldsymbol{x}_{\setminus i}$. When expanding only for $j \neq i$ the approximation is equal in first order to $\tilde{p}^{\text{EP-FACT}}(x_i)$ (Section 4.3).

- EP-FACT. The factorized EP approximation. The approximation $\tilde{p}^{\text{EP-FACT}}(x_i)$ is computed using the approximation $c_i(x_i) \approx \prod_{j\neq i}\int dx_j\, q(x_j|x_i)\varepsilon_j(x_j)$, where the univariate integrals are computed numerically or analytically, if it is the case. For further details see Section 4.2.

- LA-FACT. A similar approximation as EP-FACT, but here, the univariate integrals are computed with the Laplace method and using the approximation $x_j^*(x_i) \approx \text{E}_q[x_j|x_i]$, with $q(\boldsymbol{x})$ being the global approximation resulting from the Laplace method. For further details see Section 4.2.

- EP-FACTN. Higher order approximations obtained by using the factorization recursively. For further details see Section 4.2.

References

P. R. Amestoy, T. A. Davis, and Iain S. D. An approximate minimum degree ordering algorithm. *SIAM Journal on Matrix Analysis and Applications*., 17(4):886–905, October 1996.

A. Birlutiu and T. Heskes. Expectation propagation for rating players in sports competitions. In Joost N. Kok, Jacek Koronacki, Ramon López de Mántaras, Stan Matwin, Dunja Mladenic, and Andrzej Skowron, editors, *Proceedings ECML/PKDD*, volume 4702 of *Lecture Notes in Computer Science*, pages 374–381. Springer, 2007.

L. Csató and M. Opper. Sparse representation for Gaussian process models. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, Cambridge, MA, USA, 2001. MIT Press.

P. Dangauthier, R. Herbrich, T. Minka, and T. Graepel. Trueskill through time: Revisiting the history of chess. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 337–344. MIT Press, Cambridge, MA, 2008.

A. M. Erisman and W. F. Tinney. On computing certain elements of the inverse of a sparse matrix. *Communications of the ACM*, 18(3):177–179, 1975. ISSN 0001-0782.

T. Heskes, M. Opper, W. Wiegerinck, O. Winther, and O. Zoeter. Approximate inference techniques with expectation constraints. *Journal of Statistical Mechanics: Theory and Experiment*, 2005: P11015, 2005.

S. Ingram. Minimum degree reordering algorithms: A tutorial, 2006. URL `http://www.cs.ubc.ca/~sfingram/cs517_final.pdf`.

M. Kuss and C. E. Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005. ISSN 1533-7928.

S. Martino and H. Rue. Implementing approximate Bayesian inference using integrated nested Laplace approximation: a manual for the INLA program. Technical report, Department of Mathematical Sciences, NTNU, Norway, 2009.

T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, MIT, 2001.

T. P. Minka. Divergence measures and message passing. Technical Report MSR-TR-2005-173, Microsoft Research Ltd., Cambridge, UK, December 2005.

K. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, volume 9, pages 467–475, San Francisco, USA, 1999. Morgan Kaufman.

I. Murray, R. P. Adams, and D. J.C. MacKay. Elliptical slice sampling. In Y. W. Teh and M. Titterington, editors, *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 541–548. 2010.

M. Opper and C. Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

M. Opper and O. Winther. Gaussian processes for classification: Mean-field algorithms. *Neural Computation*, 12(11):2655–2684, 2000.

M. Opper, U. Paquet, and O. Winther. Improving on expectation propagation. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1241–1248. MIT, Cambridge, MA, US, 2009.

H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications*, volume 104 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, London, UK, 2005.

H. Rue, S. Martino, and N. Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society (Series B)*, 71(2):319–392, 2009.

M. W. Seeger. Bayesian inference and optimal design for the sparse linear model. *Journal of Machine Learning Research*, 9:759–813, 2008. ISSN 1533-7928.

K. Takahashi, J. Fagan, and M.-S. Chin. Formation of a sparse impedance matrix and its application to short circuit study. In *Proceedings of the 8th PICA Conference*, 1973.

L. Tierney and J. B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

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.

M. van Gerven, B. Cseke, F. de Lange, and T. Heskes. Efficient Bayesian multivariate fMRI analysis using a sparsifying spatio-temporal prior. *Neuroimage*, 50(1):150–161, March 2010.

O. Zoeter and T. Heskes. Gaussian quadrature based expectation propagation. In Z. Ghahramani and R. Cowell, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 445–452. Society for Artificial Intelligence and Statistics, 2005.

Journal of Machine Learning Research 12 (2011) 455-490 Submitted 5/10; Revised 1/11; Published 2/11

Posterior Sparsity in Unsupervised Dependency Parsing

Jennifer Gillenwater JENGI@CIS.UPENN.EDU
Kuzman Ganchev KUZMAN@CIS.UPENN.EDU
João Graça GRACA@CIS.UPENN.EDU
Department of Computer and Information Science
University of Pennsylvania
Levine 302, 3330 Walnut St
Philadelphia, PA 19104, USA

Fernando Pereira PEREIRA@GOOGLE.COM
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA

Ben Taskar TASKAR@CIS.UPENN.EDU
Department of Computer and Information Science
University of Pennsylvania
Levine 302, 3330 Walnut St
Philadelphia, PA 19104, USA

Editor: Mark Johnson

Abstract

A strong inductive bias is essential in unsupervised grammar induction. In this paper, we explore a particular sparsity bias in dependency grammars that encourages a small number of unique dependency types. We use part-of-speech (POS) tags to group dependencies by parent-child types and investigate sparsity-inducing penalties on the posterior distributions of parent-child POS tag pairs in the posterior regularization (PR) framework of Graça et al. (2007). In experiments with 12 different languages, we achieve significant gains in directed attachment accuracy over the standard expectation maximization (EM) baseline, with an average accuracy improvement of 6.5%, outperforming EM by at least 1% for 9 out of 12 languages. Furthermore, the new method outperforms models based on standard Bayesian sparsity-inducing parameter priors with an average improvement of 5% and positive gains of at least 1% for 9 out of 12 languages. On English text in particular, we show that our approach improves performance over other state-of-the-art techniques.

1. Introduction

We investigate unsupervised learning methods for dependency parsing models that impose sparsity biases on the types of dependencies. We assume a corpus annotated with part-of-speech (POS) tags, where the task is to induce a dependency model from the tag sequences for corpus sentences. In this setting, the *type* of a dependency is defined as a simple pair: tag of the dependent (also known as the child), and tag of the head (also known as the parent) for that dependent. Given that POS tags are typically designed to convey information about grammatical relations, it is reasonable to expect that only some of the possible dependency types would be realized for any given language. For instance, it is ungrammatical for nouns to dominate verbs, adjectives to dominate adverbs, and

©2011 Jennifer Gillenwater, Kuzman Ganchev, João Graça, Fernando Pereira and Ben Taskar.

determiners to dominate almost any part of speech. In other words, the realized dependency types should be a sparse subset of all the possible types.

Previous work in unsupervised grammar induction has mostly focused on achieving sparsity through priors on model parameters. For instance, Liang et al. (2007), Finkel et al. (2007) and Johnson et al. (2007) experimented with hierarchical Dirichlet process priors, and Headden III et al. (2009) proposed a (non-hierarchical) Dirichlet prior. Such priors on parameters encourage a standard generative dependency parsing model (see Section 2) to limit the number of dependent types for each head type. Although not focused on sparsity, several other studies use soft parameter sharing to constrain the capacity of the model and hence couple different types of dependencies. To this end, Cohen et al. (2008) and Cohen and Smith (2009) investigated a (shared) logistic normal prior, and Headden III et al. (2009) used a backoff scheme.

Our experiments (Section 6) show that the more effective sparsity pattern is one that limits the total number of unique head-dependent tag pairs. Unlike sparsity-inducing parameter priors, this kind of sparsity bias does not induce competition between dependent types for each head type. Our experiments validate that this translates into accuracy improvements. In all except one of the 60 model settings we try for English, we observe higher accuracy than with the *best* setting for a parameter prior baseline. In our multi-lingual experiments, we similarly observe an average absolute accuracy gain of 5%.

As we show in Section 4, we can achieve the desired bias with a sparsity constraint on model posteriors, using the posterior regularization (PR) framework (Graça et al., 2007; Ganchev et al., 2010). Specifically, to implement PR we augment the maximum likelihood objective of the generative dependency model with a term that penalizes distributions over head-dependent pairs that are too permissive. We consider two choices for the form of the penalty, and show experimentally that the following penalty works especially well: the model pays for the first time it selects a word with tag c as a dependent of a head with tag p; after that, choosing a the same head tag p for any other occurrence of c is free. While Ravi et al. (2010) also attempt a direct minimization of tag pairs for a supertagging application, they do so with a two-stage integer program that is applied after likelihood maximization is complete.

The remainder of this paper is organized as follows. Section 2 reviews the generative model for dependency parsing. Section 3 illustrates why the expectation-maximization learning method is insufficient and motivates sparse posteriors. Section 4 describes learning with PR constraints and how to encode posterior sparsity under the PR framework. Section 5 summarizes previous approaches that we compare to in our experiments, focusing in particular on attempts to induce sparsity via a parameter prior. Section 6 describes the results of dependency parsing experiments across 12 languages and against recent published state-of-the-art results for the English language. Section 7 analyzes these results, explaining why PR manages to learn where other methods fail, and Section 8 concludes. The model and all the code required to reproduce the experiments are available online at `code.google.com/p/pr-toolkit`, version 2010.11.

2. Parsing Model

The models we consider are based on the dependency model with valence (DMV) of Klein and Manning (2004). We also investigate extensions to the DMV borrowed from McClosky (2008) and Headden III et al. (2009). These extensions are not crucial to our experimental success with posterior regularization, but we choose to explore them for better comparison with previous work.

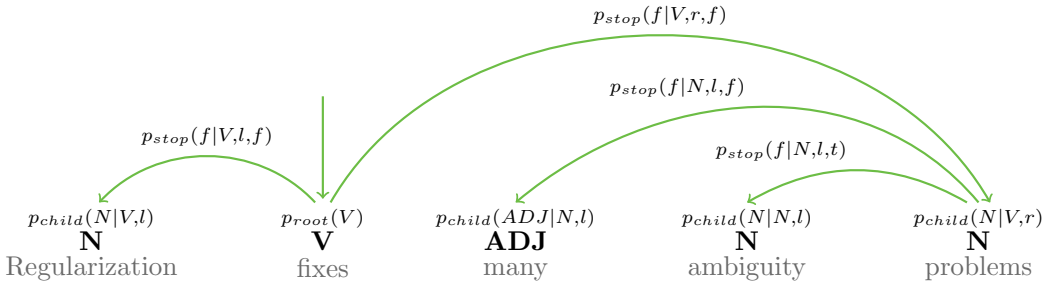


Figure 1: Example of a dependency tree with DMV probabilities. Right-dependents of a head are denoted by r, left-dependents by l. The letters t and f denote 'true' and 'false.' For example, in $p_{stop}(f \mid V,r,f)$ the f to the left of the conditioning bar indicates that the model has decided *not* to stop, and the other f indicates V does *not* yet have any right dependents. Note that the $p_{stop}(t \mid \ldots)$ are omitted in this diagram.

As will be discussed in the experiments section, both for the basic and for the extended models accuracy can be increased by applying posterior regularization. In this section we briefly describe the basic DMV model. Description of the extended models is deferred until the experiments section.

The DMV model specifies the following generative process. For a sentence consisting of POS tags $\mathbf{x}$, the root head POS $r(\mathbf{x})$ is generated first with probability $p_{root}(r(\mathbf{x}))$. For example, in Figure 1 this corresponds to generating the V with probability $p_{root}(V)$.

After generating the root, the model next generates dependents of the root. First, it generates right dependents. It decides whether to produce a right dependent conditioned on the identity of the root and the fact that it currently has no other right dependents. In our example, this decision is represented by the probability $p_{stop}(f \mid V,r,f)$. If it decides to generate a right dependent, it generates a particular dependent POS by conditioning on the fact that the head POS is $r(\mathbf{x})$ and that the directionality is to the right. In our example, this corresponds to the probability $p_{child}(N \mid V,r)$. The model then returns to the choice of whether or not to stop generating right dependents, this time conditioned on the fact that it already has at least one right dependent. In our example, this corresponds to the probability $p_{stop}(t \mid V,r,t)$, which indicates that the model is done generating right dependents of V.

After stopping the generation of right dependents, the model generates left dependents using the mirror image of the right-dependent process. Once the root has generated all of its dependents, the dependents generate their own dependents in the same manner.

We follow the convention that the model generates dependents starting with the rightmost one, moving inward (leftward) until all right dependents are added, then it generates the leftmost left dependent and moves inward (rightward) from there. This is exemplified in Figure 1, where the leftmost dependent of the final N is generated before the other left dependent. This convention has no effect on the final probability of a parse tree under the basic DMV. However, as we will note in the experiments section, it does affect dependency tree probabilities in the extended model.

3. Learning with EM

The baseline for evaluating our sparse learning methods is the expectation maximization (EM) algorithm (Dempster et al., 1977). Before the empirical comparison in Section 6, in we introduce here some notation and review the EM algorithm. In what follows, we denote the entire unlabeled corpus by $\mathbf{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^n\}$, and a set of corresponding parses for each corpus sentence by $\mathbf{Y} = \{\mathbf{y}^1, \ldots, \mathbf{y}^n\}$.

The EM algorithm is a popular method for optimizing marginal likelihood:

$$\mathcal{L}(\theta) = \log \sum_{\mathbf{Y}} p_\theta(\mathbf{X}, \mathbf{Y}).$$

We briefly review the interpretation of the EM algorithm given by Neal and Hinton (1998), as this interpretation best elucidates how the posterior regularization method we propose in Section 4 is a natural modification of the basic EM algorithm. Neal and Hinton (1998) view EM as block coordinate ascent on a function that lower-bounds $\mathcal{L}(\theta)$. We form the lower bound, denoted $F(q, \theta)$, by applying Jensen's inequality to $\mathcal{L}(\theta)$:

$$\mathcal{L}(\theta) = \log \sum_{\mathbf{Y}} q(\mathbf{Y}) \frac{p_\theta(\mathbf{X}, \mathbf{Y})}{q(\mathbf{Y})} \geq \sum_{\mathbf{Y}} q(\mathbf{Y}) \log \frac{p_\theta(\mathbf{X}, \mathbf{Y})}{q(\mathbf{Y})} = F(q, \theta).$$

Splitting up the log terms, we can then rewrite $F(q, \theta)$ as:

$$\begin{aligned} F(q, \theta) &= \sum_{\mathbf{Y}} q(\mathbf{Y}) \log(p_\theta(\mathbf{X}) p_\theta(\mathbf{Y} \mid \mathbf{X})) - \sum_{\mathbf{Y}} q(\mathbf{Y}) \log q(\mathbf{Y}) \\ &= \mathcal{L}(\theta) - \sum_{\mathbf{Y}} q(\mathbf{Y}) \log \frac{q(\mathbf{Y})}{p_\theta(\mathbf{Y} \mid \mathbf{X})} \\ &= \mathcal{L}(\theta) - \mathbf{KL}(q(\mathbf{Y}) \parallel p_\theta(\mathbf{Y} \mid \mathbf{X})). \end{aligned} \tag{1}$$

Based on this formulation, we can view EM as performing coordinate ascent on $F(q, \theta)$. Starting from an initial parameter estimate θ^0, the algorithm iterates two block coordinate ascent steps until a convergence criterion is attained:

$$\begin{aligned} \mathbf{E}: q^{t+1} &= \arg\max_q F(q, \theta^t) = \arg\min_q \mathbf{KL}(q(\mathbf{Y}) \parallel p_{\theta^t}(\mathbf{Y} \mid \mathbf{X})), \\ \mathbf{M}: \theta^{t+1} &= \arg\max_\theta F(q^{t+1}, \theta) = \arg\max_\theta \mathbf{E}_{q^{t+1}} [\log p_\theta(\mathbf{X}, \mathbf{Y})]. \end{aligned} \tag{2}$$

Note that the E-step just sets $q^{t+1}(\mathbf{Y}) = p_{\theta^t}(\mathbf{Y}|\mathbf{X})$, since it performs an unconstrained minimization of a Kullback-Leibler divergence.

Figure 2 illustrates the large mismatch between an EM-trained DMV model and the empirical statistics of dependency types. We will eventually show that posterior regularization reduces the mismatch much more successfully than approaches based on parameter priors.

4. Learning with Sparse Posteriors

We stated in the introduction that posterior regularization makes gains over baseline methods such as EM by inducing sparsity in the posteriors. Before discussing how to learn a model with sparse posteriors, we wish to further motivate the idea. The main intuition behind our method is that a

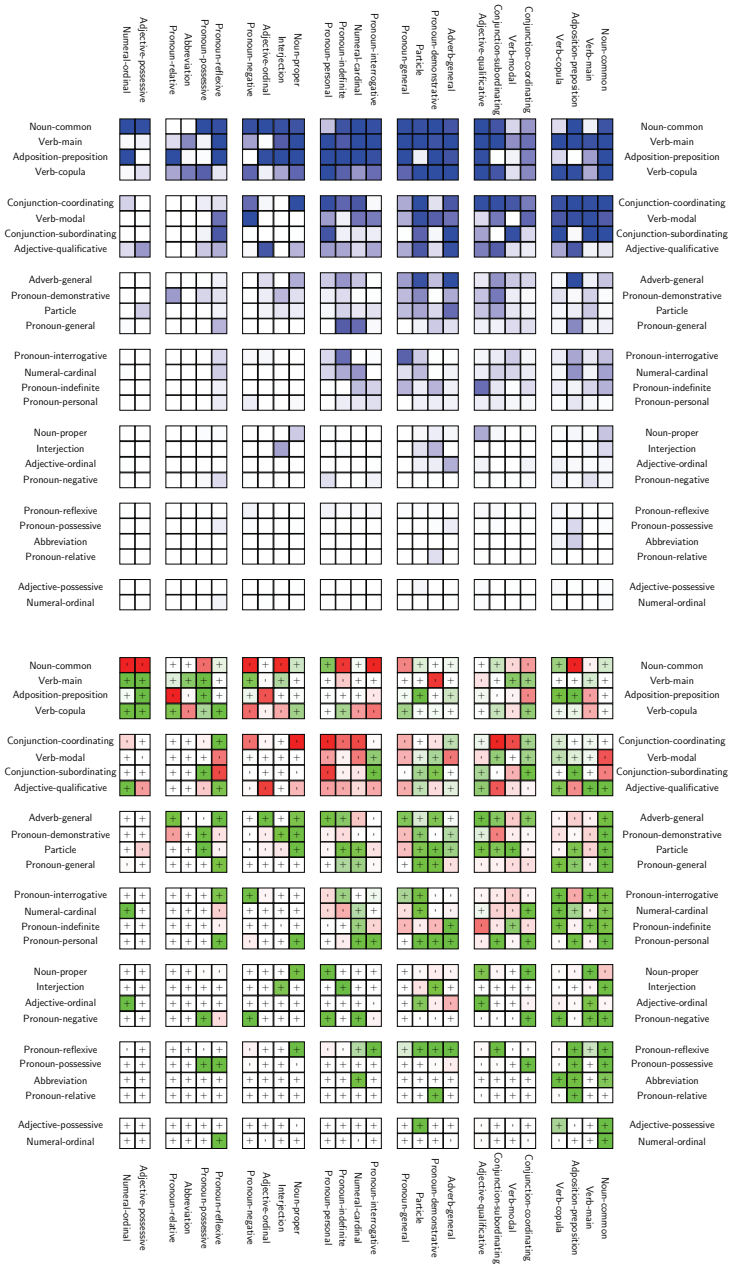


Figure 2: Comparison of posteriors for a maximum likelihood DMV and an EM-trained DMV for Slovene. Each square represents a parent-child pair. Parent tags are listed down, child tags across. Parent tags are sorted top-to-bottom in descending order by the number of unique child tags they take. **Top**: Using maximum likelihood parameter settings (supervised). The saturation of a square with parent p and child c is determined by the max value of the posterior probability of type c having parent p observed in the entire English training corpus (Marcus et al., 1993). More saturated blue indicates higher probability. **Bottom**: Using EM parameter settings. Green ("+") indicates EM posteriors are too high, red ("-") too low. More saturation indicates more deviation. There are significantly more green ("+") squares than red ("-"), indicating EM does not learn a sparse enough model.

useful grammar should only allow a relatively small subset of all possible parent-child relations. If we were asked to parse the tag sequence `DT ADJ N V`, the dependency tree with V as root, N as its child, and the remaining DT and ADJ as N's children is almost forced on us. Yet, if the English grammar allowed all possible parent-child relations, you would have had to consider 30 different (projective) parse trees before selecting the correct one. Knowledge of unlikely relations simplifies parsing for us. Thus, in this work we attempt to limit grammar ambiguity by inducing a grammar that allows only a sparse set of possible dependency relation types.

Empirical evidence that good grammars have sparse coverage of the possible parent-child relations can be seen in Figure 2. The grid corresponding to supervised parameter settings has many white squares, which illustrates that many parent-child relations should have zero posterior. Notice also that while some parent tags can take many different child tags, some parent tags can take just a few child tags, and some tags cannot be parents; the number of allowed child tags spans a wide range. These empirical properties are not captured by previous attempts to achieve model sparsity with hierarchical Bayesian models, which push each *each* parent tag to allow only a few child tags. Instead, the modeling framework should simply favor models with high overall ratio of white squares to blue squares.

The foregoing argument leads us to seek learning methods that will penalize learned distributions $p_\theta(\mathbf{Y}|\mathbf{X})$ that predict a large number of distinct dependency types. In the next section, we discuss different ways of counting dependency types, corresponding to slightly different measures of ambiguity. In Section 4.3, we will explain how to use those measures as mixed-norm penalties on distributions over dependency trees.

We will then discuss how to apply the posterior regularization (PR) framework (Graça et al., 2007; Ganchev et al., 2010) to achieve the desired sparsity in grammar induction. The approach, reviewed in Section 4.2, is closely related to generalized expectation constraints (Mann and McCallum, 2007, 2008; Bellare et al., 2009), and is also indirectly related to a Bayesian view of learning with constraints on posteriors (Liang et al., 2009). The PR framework uses constraints on posterior expectations to help guide parameter estimation. It allows for tractable learning and inference even when the constraints it enforces would be intractable to encode directly as additional model parameters or structure. In particular, PR allows a natural representation of the dependency sparsity constraints based on the ambiguity measures described below. For a more complete analysis of PR and its application to a variety of NLP tasks, we refer the reader to Ganchev et al. (2010).

4.1 Measures of Ambiguity

We now describe precisely how to count dependency types, which will allow us to specify different kinds of dependency sparsity. For each child tag c, let i range over some arbitrary enumeration of all occurrences of c in the corpus, and let p be another tag. The indicator $\phi_{cpi}(\mathbf{X}, \mathbf{Y})$ has value 1 if p is the tag of the parent of the ith occurrence of c, and value 0 otherwise. The number of unique dependency types is then given by:

$$\sum_{cp} \max_i \phi_{cpi}(\mathbf{X}, \mathbf{Y}), \tag{3}$$

where we sum over child-parent types cp, computing the maximum (logical or) over possible occurrences of $c \leftarrow p$ dependencies. Note that there is an asymmetry in this way of counting types: occurrences of the child type c are enumerated with i, but all occurrences of the parent type p are or-ed in ϕ_{cpi}, that is, ϕ_{cpi} is 1 if *any* occurrence of tag p is the parent of the ith occurrence of tag c. See the top sentence in Figure 4 for an example of this; there the noun child in the POS sequence N

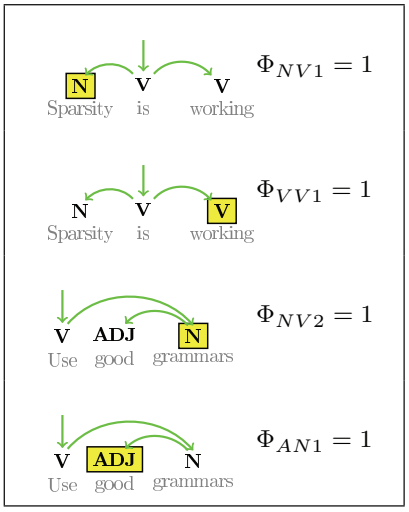


Figure 3: The ℓ_1/ℓ_∞ ambiguity measure for a toy example with gold parse trees. Let $\mathbf{\Phi}_{cpi} = \mathbf{E}_q[\phi_{cpi}]$. For simplicity we ignore the root $\rightarrow c$ edges here, though in our experiments we incorporate their probabilities also. **Left**: Two gold parse trees with two (non-root) children each. Edges in the trees have probability 1, and all other edges probability 0. **Right**: Computation of the grammar ambiguity measure, which is 3 in this case. The same result can also be obtained using ϕ_{cpij} instead.

V V is considered, and the probabilities of each of its possible parents are summed into one factor, $\mathbf{\Phi}_{NV1}$, since the parents are both of the same type (V). We use PR-AS, asymmetric PR, to refer to PR training with constraints based on this ambiguity measure.

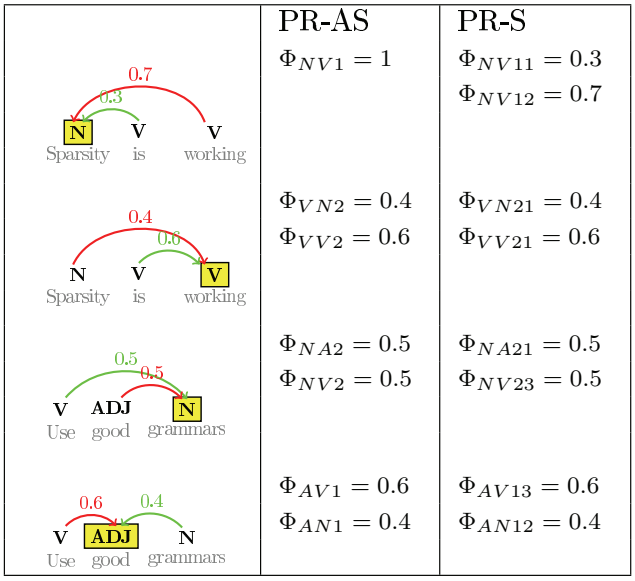
Instead of counting pairs of a child token and a parent type, we could instead have counted pairs of a child token and a parent token by letting p range over all *tokens* rather than *types*. In that case, each potential dependency would correspond to a different indicator ϕ_{cpij}, and the penalty would be symmetric with respect to parents and children. We use PR-S, symmetric PR, to refer to PR training with constraints based on this measure. The number of unique dependency types in this case is given by:

$$\sum_{cp} \max_{i,j} \phi_{cpij}(\mathbf{X}, \mathbf{Y}).$$

On actual dependency trees, where each child has a unique parent, PR-AS and PR-S always yield the same value. However, the values may be different when working with distributions over edge types instead, as exemplified in Figure 4. Both PR-AS and PR-S perform very well. One approach is not clearly better than the other when compared across the twelve languages, so we report results for both versions in the results section.

In addition to PR-AS and PR-S, there is in fact a third way of counting—another asymmetric method. For PR-AS all parent tokens are collapsed, but we could also consider the case where all child tokens are collapsed. Then the number of unique dependency types would be:

$$\sum_{cp} \max_{j} \phi_{cpj}(\mathbf{X}, \mathbf{Y}).$$



	PR-AS	PR-S
	$\Phi_{NV1} = 1$	$\Phi_{NV11} = 0.3$ $\Phi_{NV12} = 0.7$
	$\Phi_{VN2} = 0.4$ $\Phi_{VV2} = 0.6$	$\Phi_{VN21} = 0.4$ $\Phi_{VV21} = 0.6$
	$\Phi_{NA2} = 0.5$ $\Phi_{NV2} = 0.5$	$\Phi_{NA21} = 0.5$ $\Phi_{NV23} = 0.5$
	$\Phi_{AV1} = 0.6$ $\Phi_{AN1} = 0.4$	$\Phi_{AV13} = 0.6$ $\Phi_{AN12} = 0.4$

PR-Asymmetric

$$\xi_{\mathbf{NV}} = \max(\Phi_{NV1}, \Phi_{NV2}) = 1$$
$$\xi_{VN} = \max(\Phi_{VN2}) = 0.4$$
$$\xi_{\mathbf{VV}} = \max(\Phi_{VV2}) = 0.6$$
$$\xi_{NA} = \max(\Phi_{NA2}) = 0.5$$
$$\xi_{AV} = \max(\Phi_{AV1}) = 0.6$$
$$\xi_{\mathbf{AN}} = \max(\Phi_{AN1}) = 0.4$$
$$\sum_{cp} \xi_{cp} = \mathbf{3.5}$$

PR-Symmetric

$$\xi_{\mathbf{NV}} = \max(\Phi_{NV11}, \Phi_{NV12}, \Phi_{NV23}) = 0.7$$
$$\xi_{VN} = \max(\Phi_{VN21}) = 0.4$$
$$\xi_{\mathbf{VV}} = \max(\Phi_{VV21}) = 0.6$$
$$\xi_{NA} = \max(\Phi_{NA21}) = 0.5$$
$$\xi_{AV} = \max(\Phi_{AV13}) = 0.6$$
$$\xi_{\mathbf{AN}} = \max(\Phi_{AN12}) = 0.4$$
$$\sum_{cp} \xi_{cp} = \mathbf{3.2}$$

Figure 4: The ℓ_1/ℓ_∞ ambiguity measure for a toy example using edge posteriors. Let $\Phi_{cpi} = \mathbf{E}_q[\phi_{cpi}]$, and similarly $\Phi_{cpij} = \mathbf{E}_q[\phi_{cpij}]$. For simplicity we ignore the root $\rightarrow c$ edges here, though in our experiments we incorporate their probabilities also. The two POS tag sequences considered are the same as in 3; we also consider the same four children here for easy comparison. In this unsupervised setting, instead of gold trees we have an example posterior distribution over parents for each child. We illustrate computation of the grammar ambiguity measure for both PR-AS (left), and PR-S(right). Since real grammars tend to have few edge types, it should make sense that the ℓ_1/ℓ_∞ of the set of supervised trees in 3 was smaller.

This type of counting leads however to some unintuitive results. For instance, consider a parse tree consisting of a verb with two noun children. There, $\phi_{NV1} = 2$. This does not correspond to a count of unique parent-child pairs, so it does not serve our ultimate goal as well as PR-AS or PR-S. Hence, we do not experiment with this ambiguity measure in this work.

4.2 Posterior Regularization

Having defined several ambiguity measures, we now step back and describe the general PR framework. After this overview, we will show how to apply this general framework to penalize with respect to the specific ambiguity measures we defined. In general, PR can be seen as a penalty on

the standard marginal log-likelihood objective, which we define first as:

$$\textbf{Likelihood objective:}\ \mathcal{L}(\theta) = \log p_\theta(\mathbf{X}) + \log p(\theta) = \sum_{\mathbf{x}\in\mathbf{X}} [\log \sum_{\mathbf{y}} p_\theta(\mathbf{x},\mathbf{y})] + \log p(\theta), \tag{4}$$

where θ represents the model parameters, $p(\theta)$ is a (optional) prior probability on the parameters, and the sum is over the unlabeled sample data. Recall that we use $\mathbf{x}$ to denote a single sentence's POS tags, and $\mathbf{y}$ to denote a single hidden parse tree.

Here we present the penalty version of PR; Ganchev et al. (2010) describe a constraint-set version of PR and give more details. In PR, the desired bias is specified with a penalty on expectations of features ϕ. For any distribution q over latent variables, we can define a penalty as the β-norm of the feature expectations:

$$\left|\left|\mathbf{E}_q[\phi(\mathbf{X},\mathbf{Y})]\right|\right|_\beta,$$

where $\mathbf{Y}$ represents an assignment of parse trees for all sentences in the corpus $\mathbf{X}$. For computational tractability, rather than penalizing the model's posteriors directly, we use an auxiliary distribution, and penalize the marginal log-likelihood of a model by the KL-divergence and penalty term with respect to q. For a fixed set of model parameters θ the PR penalty term we will use is given by:

$$\textbf{Penalty term:}\ \min_q \mathbf{KL}(q(\mathbf{Y}) \parallel p_\theta(\mathbf{Y}|\mathbf{X}))\ +\ \sigma\left|\left|\mathbf{E}_q[\phi(\mathbf{X},\mathbf{Y})]\right|\right|_\beta, \tag{5}$$

where σ is the strength of the regularization. As we will see, using an auxiliary distribution q will make the final objective easier to optimize. Ganchev et al. (2010) describe how to compute this penalty term in general, but we will defer that explanation to Section 4.3 when we describe our particular penalty term. The PR framework seeks to maximize:

$$\textbf{PR objective:}\ J(\theta) = \mathcal{L}(\theta) - \min_q \left[\mathbf{KL}(q(\mathbf{Y}) \parallel p_\theta(\mathbf{Y}|\mathbf{X}))\ +\ \sigma\left|\left|\mathbf{E}_q[\phi(\mathbf{X},\mathbf{Y})]\right|\right|_\beta\right]. \tag{6}$$

The objective in Equation 6 can be optimized by a variant of the EM algorithm (Dempster et al., 1977) used to optimize the objective in Equation 4.

4.3 ℓ_1/ℓ_∞ Regularization

The previous section gave the penalty version of the PR objective in the general case. We will now show how the ambiguity measures we want to incorporate fit into this framework. Specifically, notice that we can view Equation 3 as a mixed-norm penalty on the features ϕ_{cpi} so that the generic β from Equation 5 becomes ℓ_1/ℓ_∞. More precisely, we will penalize the following quantity: the sum (ℓ_1 norm) over c of the maximum (ℓ_∞ norm) over occurrences of c of the posterior probability of selecting a parent with tag p for that child. To compute the value of the PR objective and also to optimize it, we need to compute the projection:

$$\arg\min_q \mathbf{KL}(q(\mathbf{Y})||p_\theta(\mathbf{Y}|\mathbf{X})) + \sigma \sum_{cp} \max_i \mathbf{E}_q[\phi_{cpi}(\mathbf{X},\mathbf{Y})],$$

which can equivalently be written as:

$$\textbf{Projection}: \min_{q,\xi}\ \mathbf{KL}(q(\mathbf{Y}) \parallel p_\theta(\mathbf{Y}|\mathbf{X})) + \sigma \sum_{cp} \xi_{cp} \quad \text{s.t.}\quad \xi_{cp} \geq \mathbf{E}_q[\phi_{cpi}(\mathbf{X},\mathbf{Y})]\ \ \forall c,p,i, \tag{7}$$

where σ is the strength of the regularization, and ξ_{cp} corresponds to the maximum expectation of ϕ_{cpi} over all c and p. Note that the projection problem is convex in q and can be solved efficiently in the dual (just as for the maximum entropy/log linear model fitting). The formulation of Equation 7 makes the derivation of the dual easier (see Ganchev et al., 2010 for a derivation of the dual in the general case). The dual of the projection problem is a fairly simple convex optimization problem with simplex constraints (scaled by σ):

$$\textbf{Projection dual}: \min_{\lambda \geq 0} \; \log \left(\sum_{\mathbf{Y}} p_\theta(\mathbf{Y}|\mathbf{X}) \exp(-\boldsymbol{\lambda} \cdot \phi(\mathbf{X}, \mathbf{Y})) \right) \quad \text{s.t.} \quad \sum_i \lambda_{cpi} \leq \sigma,$$

where ϕ is the vector of feature values ϕ_{cpi} for assignment $\mathbf{Y}$ of parse trees to the entire corpus $\mathbf{X}$, and $\boldsymbol{\lambda}$ is the vector of dual parameters λ_{cpi}. The optimal primal solution is related to the dual solution by the equation $q(\mathbf{Y}) \propto p_\theta(\mathbf{Y}|\mathbf{X}) \exp(-\boldsymbol{\lambda} \cdot \phi(\mathbf{X}, \mathbf{Y}))$. We solve the dual via projected gradient, as described by Bertsekas (1995). Note that projection onto the simplex constraints can be done very efficiently as described in Bertsekas (1995).

When σ is zero, the projection is an identity mapping and the algorithm reduces to EM. For intermediate values of σ, the constraints work to decrease the confidence of the highest probability parent tags for each child instance. For parent tags that are supported by many high-probability instances, this pressure is distributed among many instances and has little effect. For parent tags that are supported by few high-probability instances however, the probability of these instances is more severely reduced, which can (after several iterations of the algorithm) effectively eliminate that parent tag as a possibility for the given child tag.

4.4 Optimization Algorithms

The optimization algorithm for the PR objective uses a minorization-maximization procedure akin to EM. Recall that we defined the PR objective (Equation 6) as:

$$J(\theta) = \mathcal{L}(\theta) - \min_q \left[\mathbf{KL}(q(\mathbf{Y}) \parallel p_\theta(\mathbf{Y}|\mathbf{X})) + \sigma \left|\left| \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] \right|\right|_\beta \right].$$

If we further define:

$$F'(q, \theta) = \mathcal{L}(\theta) - \left[\mathbf{KL}(q(\mathbf{Y}) \parallel p_\theta(\mathbf{Y}|\mathbf{X})) + \sigma \left|\left| \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] \right|\right|_\beta \right],$$

then we can express the PR objective in a form very similar to that of the previously introduced lower bound on EM (Equation 1):

$$J(\theta) = \max_q F'(q, \theta).$$

This objective can then be optimized by modifying the E-step of EM to include the β-norm penalty:

$$\mathbf{E}' : q^{t+1} = \arg\max_q F'(q, \theta^t) = \arg\min_q \mathbf{KL}(q(\mathbf{Y}) \parallel p_{\theta^t}(\mathbf{Y}|\mathbf{X})) + \sigma \left|\left| \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] \right|\right|_\beta . \quad (8)$$

The projected posteriors $q^{t+1}(\mathbf{Y})$ are then used to compute sufficient statistics and update the model's parameters in the M-step, which remains unchanged, as in Equation 2. This scheme is illustrated in Figure 5. The following proposition is adapted from Ganchev et al. (2010), who provide a version for hard constraints.

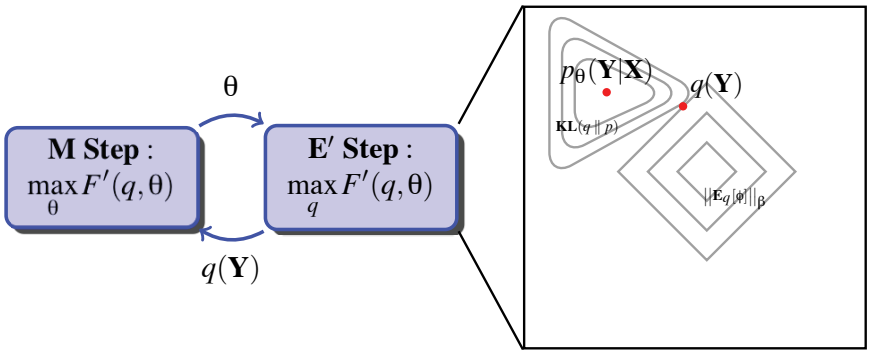


Figure 5: Modified EM for maximizing the PR objective $J(\theta)$ via block-coordinate ascent on lower-bound $F'(q,\theta)$. $\mathbf{E}'$-step minimizes $\mathbf{KL}(q(\mathbf{Y})||p_\theta(\mathbf{Y}|\mathbf{X})) + \sigma\left|\left|\mathbf{E}_q[\phi(\mathbf{X},\mathbf{Y})]\right|\right|_\beta$.

Proposition 4.1 *For the modified EM algorithm illustrated in Figure 5, which iterates the* $\mathbf{E}'$*-step (Equation 8) with the normal* **M***-step (Equation 2), monotonically increases the PR objective:* $J(\theta^{t+1}) \geq J(\theta^t)$.

Proof The proof is analogous to the proof of monotonic increase of the standard EM objective. Essentially:

$$J(\theta^{t+1}) = F'(q^{t+2},\theta^{t+1}) \geq F'(q^{t+1},\theta^{t+1}) \geq F'(q^{t+1},\theta^t) = J(\theta^t).$$

The $\mathbf{E}'$-step sets $q^{t+1} = \arg\max_q F'(q,\theta^t)$, hence $J(\theta^t) = F'(q^{t+1},\theta^t)$. The M-step sets $\theta^{t+1} = \arg\max_\theta F'(q^{t+1},\theta)$, hence $F'(q^{t+1},\theta^{t+1}) \geq F'(q^{t+1},\theta^t)$. Finally, $J(\theta^{t+1}) = \max_q F'(q,\theta^{t+1}) \geq F'(q^{t+1},\theta^{t+1})$. ■

As for standard EM, to prove that coordinate ascent on $F'(q,\theta)$ converges to stationary points of $J(\theta)$, we need to make additional assumptions on the regularity of the likelihood function and boundedness of the parameter space as in Tseng (2004). This analysis can be easily extended to our setting, but is beyond the scope of the current paper.

We note that optimizing the PR objective does take substantially longer than optimizing likelihood by itself. When optimizing likelihood, we can get the optimal posteriors for an E-step using just one call to the inside-outside algorithm for each sentence. For PR though, the function we are optimizing in the $\mathbf{E}'$-step is a **KL** plus a penalty term, so to find its minimum we have to follow the negative gradient. Each step along the negative gradient requires a call to the inside-outside algorithm—several calls if the initial step size we try does not satisfy the Wolfe conditions. Thus, it might be better to use an optimization schedule where $\mathbf{E}'$-step would not be fully optimized in earlier iterations, perhaps taking just a single step along the negative gradient. Then, in later $\mathbf{E}'$-steps, we could increase the precision of the optimization by taking more gradient descent steps (if they are required to get close to the minimum). Fortunately, in practice we found that, at least for the experiments in this paper, the optimization did not take so long that such a schedule was necessary.

5. Prior Learning Approaches and Model Extensions

We will compare PR to simple EM and to the methods of several previous studies in Section 6. Before that, we review the theory behind the previous work.

5.1 Bayesian Learning

The main learning method we will compare with experimentally is Bayesian learning with a sparsity-inducing prior. We will also compare our accuracy to that achieved by several methods that use other priors. This latter comparison will be less direct though, as these priors tend to encode linguistic information at a finer-grained level.

Recent advances in Bayesian inference methods have been applied to DMV grammar induction with varying levels of success. These approaches have focused on injecting linguistic knowledge into the DMV by using a Dirichlet prior to sparsify parameters (Cohen et al., 2008; Headden III et al., 2009), or using logistic normal priors to tie parameters (Cohen et al., 2008; Cohen and Smith, 2009). In the following subsections, we will review those methods; experimental comparisons are given in Section 6.

5.1.1 SPARSITY-INDUCING PRIORS

Dirichlet priors have been often used in DMV learning. More precisely, the prior distribution of the parameters of the DMV represented as a probabilistic context-free grammar (PCFG) is specified as a product of Dirichlets: $p(\theta) = \prod_{A \in V_N} D(\theta_A; \alpha_A)$ where the underlying CFG is $G = (V_N, V_T, R, S)$ with V_N, V_T, and R a set of non-terminals, terminals, and rules, respectively, and S a start symbol. (See Smith, 2006 for a detailed encoding of the DMV as a PCFG.) Each Dirichlet in this prior has the form:

$$D(\theta_A; \alpha_A) = \frac{1}{Z} \prod_{\beta : A \to \beta \in R} \theta_A(\beta)^{\alpha_{A \to \beta} - 1},$$

where Z is a normalization term and the αs are hyperparameters.

The true posterior over the parameters, $p(\theta|\mathbf{X}) \propto \sum_{\mathbf{Y}} p(\mathbf{Y}, \mathbf{X}|\theta) p(\theta)$, is generally multi-modal and intractable to compute. The typical variational approximation is to define an approximate factored posterior over both parameters and latent variables, $q(\mathbf{Y}, \theta) = q(\mathbf{Y}) q(\theta)$, and use mean-field updates to minimize $\mathbf{KL}(q(\mathbf{Y})q(\theta) || p(\mathbf{Y}, \theta|\mathbf{X}))$. As shown by Kurihara and Sato (2004), this can be done efficiently with the product of Dirichlets type of prior. Assuming the hyperparameters of the prior are fixed, the coordinate descent algorithm for updating $q(\mathbf{Y}), q(\theta)$ is similar to EM. In the E-like-step, inference for $\mathbf{Y}$ is performed using the approximate mean parameters $\theta = \mathbf{E}_q[\theta]$. The M-like-step is a slight modification to the standard EM M-step, both shown below:

$$\textbf{EM M-step} : \theta_A^{t+1}(\beta) \propto \mathbf{E}_{q^{t+1}}[\#_{A \to \beta}(\mathbf{Y})],$$
$$\textbf{Dirichlet M-like-step} : \theta_A^{t+1}(\beta) \propto \exp(\psi(\mathbf{E}_{q^{t+1}}[\#_{A \to \beta}(\mathbf{Y})] + \alpha_{A \to \beta})),$$

where ψ is the digamma function. As Figure 6 illustrates, $\exp(\psi(x))$ is upper bounded by $y = x$. That is, it slightly discounts the value of x, though by no more than 0.5, as $y = x - 0.5$ lower bounds it. Thus, $\exp(\psi(x + \alpha))$ is similar to adding $\alpha - 0.5$ to x. For any $\alpha < 0.5$, this encourages parameter sparsity in the **Dirichlet M**-like-step, since small θ will get squashed to zero by the digamma.

This Dirichlet prior method is applied in several previous studies. Cohen et al. (2008) use this method for dependency parsing with the DMV and achieve improvements over basic EM. They set

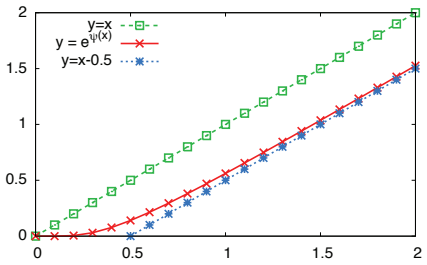


Figure 6: The digamma function.

Step	Learning Method	Formula
E-like	Standard EM	$q^{t+1} = \arg\min_q \mathbf{KL}(q(\mathbf{Y}) \parallel p_{\theta^t}(\mathbf{Y}\|\mathbf{X}))$
	Dirichlet Prior	Same as standard EM, but with θ^t replacing θ^t
	PR	$q^{t+1} = \arg\min_q \mathbf{KL}(q(\mathbf{Y}) \parallel p_{\theta^t}(\mathbf{Y}\|\mathbf{X})) + \sigma \left\|\left\|\mathbf{E}_q[\phi(\mathbf{X},\mathbf{Y})]\right\|\right\|_\beta$
M-like	Standard EM	$\theta^{t+1} \propto \mathbf{E}_{q^{t+1}}[\log p_\theta(\mathbf{X},\mathbf{Y})]$
	Dirichlet Prior	$\theta^{t+1} \propto \exp(\psi(\mathbf{E}_{q^{t+1}}[\log p_\theta(\mathbf{X},\mathbf{Y})] + \alpha))$
	PR	Same as standard EM

Table 1: E-like and M-like steps for the three main learning methods we compare in this work. The main differences are that PR changes the standard E-step to add a penalty term, while a Dirichlet prior changes the standard M-step to add pseudo-counts.

all hyperparameters to 0.25, resulting in a sparsifying prior (this is the method referred to as VB-Dirichlet in their work). In this paper we will refer to our own implementation of this method as the "sparsifying Dirichlet prior" (SDP) method. We will show experiments applying it to both the DMV and the E-DMV. In particular we will show that while it achieves parameter sparsity, this is not the optimal sparsity to aim for in dependency parsing. Intuitively, sparsity of $p_{child}(c \mid p,d)$ means requiring that each parent tag has few unique child tags. But as the supervised grid in Figure 2 illustrates, some parents should be allowed many different types of children. For example, VBZ, VBD, VBP, VB, IN, NN, etc. all should be able to have non-zero $p_{child}(c \mid p,d)$ for many c. We will show that posterior regularization is one way to achieve a better type of sparsity.

Headden III et al. (2009) also use a Dirichlet prior to train both the DMV and the E-DMV. However, they set all hyperparameters to 1, so their prior is not aimed at sparsifying. It nevertheless produces different results than standard EM because it sets parameters according to the mean of the posterior $q(\theta)$ instead of the mode. We will refer to this (non-sparsifying) Dirichlet prior method as DP in the remainder of this paper. We have now covered the two learning methods we will directly compare to, EM and Dirichlet priors, so we summarize their respective E-like and M-like steps along with those of PR in Table 1 for ease of comparison.

5.1.2 PARAMETER-TYING PRIORS

In addition to Dirichlet priors, other types of priors have been used, namely logistic normal priors (LN) (Cohen et al., 2008) and shared logistic normal priors (SLN) (Cohen and Smith, 2009). While the SDP aims to induce parameter sparsity, LN and SLN aim to tie parameters together, but all of the methods have the same goal of favoring more concise grammars. By tying parameters for different tags, the grammar is not really as ambiguous as the full range of possible parameter settings would suggest.

The LN prior has the form $p(\theta) = \prod_{A \in V_N} \mathcal{N}(\mu_A, \Sigma_A)$, where μ_A is a mean vector and Σ_A is a covariance matrix for a normal distribution over the PCFG rules with lefthand side A. The Σ_A allow rules with identical lefthand sides to co-vary, effectively tying these parameters. For example, LN can tie the parameters $p_{child}(c_1 \mid p, d)$ and $p_{child}(c_2 \mid p, d)$. The SLN prior extends the capabilities of the LN prior by allowing any arbitrary parameters to be tied. In this case, parameters such as $p_{child}(c \mid p_1, d)$ and $p_{child}(c \mid p_2, d)$ can be tied even though they correspond to PCGF rules with different lefthand sides. We compare in the experimental section against some results from using LN and SLN and show that our posterior regularization method produces higher accuracy results.

5.2 Other Learning Approaches

Several additional training alternatives have been proposed besides Bayesian methods. In particular, we will briefly describe here four such methods: contrastive estimation (CE), skewed deterministic annealing (SDA), structural annealing (SA), and direct model minimization through an integer program. We present an empirical comparison to the first three of these methods in Section 6 and show we can often achieve superior performance with posterior regularization. The fourth method has not yet been applied to the dependency parsing task we evaluate on in this work, so we defer direct comparison.

The first approach, contrastive estimation (CE), has been used to train log-linear models on unlabeled data (Smith and Eisner, 2005b,a). The basic idea is to maximize the following:

$$\log \prod_i \frac{\sum_{\mathbf{y} \in \mathbf{Y}} \exp(\theta \cdot f(\mathbf{x}^{(i)}, \mathbf{y}))}{\sum_{(\mathbf{x}, \mathbf{y}) \in N(\mathbf{x}^{(i)}) \times Y} \exp(\theta \cdot f(\mathbf{x}, \mathbf{y}))}, \tag{9}$$

where f is some vector of feature functions, and $N(\mathbf{x}^{(i)})$ is a set of $\mathbf{x}$ that are in the "neighborhood" of $\mathbf{x}^{(i)}$. The intuition behind this method is that if a person chose to produce $\mathbf{x}^{(i)}$ out of all the possible $\mathbf{x}$ in $N(\mathbf{x}^{(i)})$, then we want to learn a model that assigns higher value to $\mathbf{x}^{(i)}$ (the numerator in Equation 9) than to these other $\mathbf{x}$. Restricting to a neighborhood is necessary for tractability, and the choice of neighborhood can encode linguistic knowledge. For example, for dependency parsing Smith and Eisner (2005a) formed neighborhoods by deleting any one word from $\mathbf{x}^{(i)}$, or transposing any two words.

Two other non-Bayesian approaches of note are skewed deterministic annealing (SDA) and structural annealing (SA) (Smith and Eisner, 2006). SDA biases towards shorter dependency links as in the K&M initializer, and flattens the likelihood function to alleviate the difficulty of escaping local maxima. Alternatively, SA biases strongly toward short dependency links in early iterations, then relaxes this constraint over time.

A final related learning approach is that of Ravi et al. (2010). This work attempts to directly minimize the number of tag bigrams for a supertagging task starting from the ending point of EM,

then applying first one simple integer program, then a second more complex integer program. This method is similar to ours in that instead of using a prior, it attempts a direct minimization of tag pairs. One natural way to adapt it to dependency parsing would be to have an integer program that minimizes the number of parent-child tag pairs subject to the constraint that every sentence can still be assigned a complete parse tree. We do not compare to this proposed adaptation directly, but suspect that it would produce somewhat similar results to our PR method. One difference would be that while PR is very tightly integrated with EM, trading off between EM and the integer program would not be as straightforward as tuning a single hyperparameter.

5.3 Model Extensions

Before discussing experimental results, we detour to describe the extensions to the basic DMV that we experimented with. We implemented three model extensions, borrowed from McClosky (2008) and Headden III et al. (2009). The first extension relates to the stop probabilities, and the second two relate to dependent probabilities. With our experiments on these extended models, we aim to show that PR also achieves significant gains over other methods in a more complex model space.

5.3.1 EXTENDING STOP PROBABILITIES

The first extension conditions whether to stop generating dependents in a given direction on a larger set of previous decisions. Specifically, the probability of stopping in a particular direction depends not only on whether there are any dependents in that direction already, but also on how many. In the example of Figure 1, this corresponds to changing $p_{stop}(f \mid V, r, f)$ to $p_{stop}(f \mid V, r, 0)$ and similarly for all the other stop probabilities. The 0 in this case indicates that V has no other right dependents when it decides whether to continue generating right dependents.

In later sections of this paper, when we talk about a model with maximum stop valency S, this means we distinguish the cases of $0, 1, \ldots, S-2$, and $\geq S-1$ dependents in a given direction. The basic DMV has maximum stop valency 2 because it distinguishes between having zero dependents and at least one dependent in a given direction. A model with maximum stop valency of 3 would distinguish between having 0, 1, or at least 2 dependents in a particular direction. In this case, when a head generates more dependents in a particular direction after its second dependent, the stopping distribution it draws from will always be the same—for head p and direction d this will be $p_{stop}(\cdot \mid p, d, 2)$.

5.3.2 EXTENDING DEPENDENT PROBABILITIES

The second model extension we implement is analogous to the first, but applies to dependent tag probabilities instead of stop probabilities. That is, we expand the set of variables the model conditions on when selecting a particular dependent tag. Again, what condition on is how many other dependents were already generated in the same direction. For the example in Figure 1, this means $p_{child}(N \mid V, r)$ becomes $p_{child}(N \mid V, r, 0)$ and similarly for all other p_{child}. In later sections of this paper, when we talk about a model with maximum child valency C, this means we distinguish between having $0, 1, \ldots, C-2$, and $\geq C-1$ dependents in a particular direction. The basic DMV has maximum child valency 1 because it does not make these distinctions.

This extension to the child probabilities dramatically increases model complexity. Specifically, the number of parameters grows as $O(CT^2)$. Thus, the third and final model extension we implement

is to add a backoff for the child probabilities that does not condition on the identity of the parent POS (see Equation 10).

With this model extension, the order in which dependents are generated becomes relevant to the probability of an overall parse tree. We choose to follow the standard inwards generation order. In cases where the identity of the rightmost and leftmost dependents have a greater influence on the true stop probability than the inner dependents, this ordering will work to the model's advantage. We do not investigate in this work which languages this holds true for, though changing this ordering might be one additional way to increase parsing accuracy for some languages.

5.3.3 COMPLETE MODEL

Formally, under the extended DMV the probability of a sentence with POS tags $\mathbf{x}$ and dependency tree $\mathbf{y}$ is given by:

$$\begin{aligned} p_\theta(\mathbf{x},\mathbf{y}) = & p_{root}(r(\mathbf{x}))\times \\ & \prod_{y\in\mathbf{y}} p_{stop}(false \mid y_p, y_d, y_{v_s}) p_{child}(y_c \mid y_p, y_d, y_{v_c})\times \\ & \prod_{x\in\mathbf{x}} p_{stop}(true \mid x, left, x_{v_l})\, p_{stop}(true \mid x, right, x_{v_r}), \end{aligned}$$

where $r(\mathbf{x})$ is the root tag of the dependency tree, y is the dependency of y_c on head y_p in direction y_d, and y_{v_c}, y_{v_s}, x_{v_r}, and x_{v_l} indicate valency. To formally define these last four variables, first let V_c denote the model's maximum child valency and let V_s denote maximum stop valency. Further, let a_{cpd} to be the number of y_p's dependents that are further in direction y_d than y_c, and a_{xl} (a_{xr}) be the total number of dependents of parent x to the left (right). Then we can formally express the valency variables as:

$$\begin{aligned} y_{v_c} &= \min(V_c, a_{cpd}), & y_{v_s} &= \min(V_s, a_{cpd}), \\ x_{v_l} &= \min(V_s, a_{xl}), & x_{v_r} &= \min(V_s, a_{xr}). \end{aligned}$$

In the third model extension, the backoff for the child probability to a probability not dependent on parent POS, $p_{child}(y_c \mid y_d, y_{v_c})$, can formally be expressed by:

$$\lambda p_{child}(y_c \mid y_p, y_d, y_{v_c}) + (1-\lambda) p_{child}(y_c \mid y_d, y_{v_c}), \tag{10}$$

for $\lambda \in [0,1]$. In Headden III et al. (2009) λ is a learned model parameter. In our experiments, we do not try to tune λ, but rather fix it at $1/3$. This is a crude approximation to the value used by Headden III et al. (2009). The way Headden III et al. (2009) choose the weighting $(1-\lambda)$ for the backoff is through a Dirichlet prior. To capture the intuition that events seen fewer times should be more strongly smoothed, this prior has hyperparameter value K for the standard child probability and value $2K$ for the backoff probability, where K is the number of PCFG rules with a particular nonterminal on the left-hand side. This ensures that the backoff probability is only ignored when enough examples of the full child probability have been seen. The prior favors the backoff 2 to 1, which is why in our approximation of this scheme we use weight $\lambda = 1/3$.

6. Experiments

In this section we present positive experimental results validating the PR method. In Section 6.2 we detail experiments with different regularization strengths σ on English and analyze the correlation

	Bg	Cz	De	Dk	En	Es	Jp	Nl	Pt	Se	Si	Tr
tags	11	58	51	24	34	17	74	162	19	31	26	28
sentences	5K	24K	13K	2K	5K	0.4K	12K	7K	2K	3K	0.5K	3K
word types	11K	40K	20K	6K	10K	3K	2K	11K	7K	8K	3K	10K
word tokens	27K	139K	77K	11K	37K	2K	43K	43K	14K	23K	3K	18K

Table 2: Training corpus statistics for sentences with lengths ≤ 10, after stripping punctuation. Bg stands for Bulgarian, Cz for Czech, De for German, Dk for Danish, En for English, Es for Spanish, Jp for Japanese, Nl for Dutch, Pt for Portuguese, Se for Swedish, Sl for Slovene, and Tr for Turkish.

between accuracy and the PR learning curves. The maximum accuracy we achieve is 64.5% using an E-DMV with PR-S and $\sigma = 160$. This is significantly above the best result of the SDP baseline, which is only 53.6%. In Section 6.3 we present a summary of related work, attempting to categorize the many dimensions along which researchers have explored modifications to the most basic EM DMV setup. While direct comparison of accuracy numbers from all related work is difficult, we present evidence that combining PR with a few of those modifications (for example random pool initialization) would result in the best accuracy yet achieved, especially for longer sentences. In Section 6.4 we apply PR to 11 additional languages, using English to select the regularization strength. Our multi-lingual results show that the PR method is indeed very broadly applicable. Averaging over all languages, there seem to only be minor differences in accuracy between PR-S and PR-AS, and both produce approximately equally sparse grammars. Under the DMV, PR-AS beats the SDP baseline for 10 out of 12 languages, Danish (Dk) and Swedish (Se) being the exceptions.

We conclude this overview of the experiments with two key points that we feel show PR to be a very useful and robust method for improving unsupervised dependency parsing:

- All except one of the 60 PR settings we try for English result in higher accuracy than the *best* SDP setting.
- In our multi-lingual experiments PR makes an average absolute accuracy gain of 5% over SDP for the DMV model.

6.1 Corpora

We evaluated our models on 12 languages—the English Penn Treebank (Marcus et al., 1993) and 11 languages from the CoNLL X shared task: Bulgarian [Bg] (Simov et al., 2002), Czech [Cz] (Bohomovà et al., 2001), German [De] (Brants et al., 2002), Danish [Dk] (Kromann et al., 2003), Spanish [Es] (Civit and Martí, 2004), Japanese [Jp] (Kawata and Bartels, 2000), Dutch [Nl] (Van der Beek et al., 2002), Portuguese [Pt] (Afonso et al., 2002), Swedish [Se] (Nilsson and Hall, 2005), Slovene [Si] (Džeroski et al., 2006), and Turkish [Tr] (Oflazer et al., 2003). For English we trained on sections 2-21 of the Penn Treebank and tested on section 23. For the other languages, our training and test sets were exactly those used in CoNLL X shared task. Following Smith and Eisner (2006), we stripped punctuation from the sentences and kept only those sentences of length ≤ 10. Table 2 shows the size of the different training corpora after that filtering.

6.2 Results on English

We start with a comparison between EM and the two sparsity-inducing methods, PR and the sparsifying Dirichlet prior (SDP), on the English corpus. For all models we train for 100 iterations. Following Klein and Manning (2004), we use a "harmonic initializer", which we will refer on this paper as K&M. This initialization uses the posteriors of a "pseudo" E-step as initial parameters: posterior root probabilities are uniform $p_{root}(r(\mathbf{x})) = \frac{1}{|\mathbf{x}|}$ and head-dependent probabilities are inversely proportional to the string distance between head and dependent, $p_{child}(y_c \mid y_p, y_d, y_{v_c}) \propto \frac{1}{|y_p - y_c|}$, normalized to form a proper probability distribution. This initialization biases the parameters to prefer local attachments.

At the end of training, we smooth the resulting models by adding e^{-10} to each learned parameter, merely to remove the chance of zero probabilities for unseen events. (We did not bother to tune this value at all as it makes very little difference for final parses.) We score models by the attachment accuracy—the fraction of words assigned the correct parent—of their Viterbi (best) parses. We compare the performance of all training procedures both on the original DMV model as well as on the extended model E-DMV.

In Graça et al. (2010), the authors found that for PR, projecting at decoding consistently improved results on the task of word alignment. Consequently, they always compute the projected distribution q and decode using q rather than the model distribution. In this work, we found that projecting at decode time produced worse results. Thus, the following results do not use projection at decode time.

Following Cohen et al. (2008) we search for the best sparsifying parameter α for SDP training. See Table 5 in Appendix A for more details on the search for α. We find as Cohen et al. (2008) did that 0.25 is optimal for the DMV. SDP only achieves accuracy 46.4 in this setting, and even in its best E-DMV setting (V_s-V_c= 4-4, $\alpha = 0.1$), it only reaches accuracy 53.6. These values are far below most of the PR accuracies we will now discuss.

A comparison between EM and PR for both DMV and E-DMV are shown in Table 3. PR always performs better than EM. We performed a grid search over regularization strength (80 to 180 with a step of 20), for both the PR-S (symmetric constraint) and PR-AS (asymmetric constraint) formulations. A first observation based on Table 3 is that PR-S generally performs better than the PR-AS. Furthermore, PR-S seems less sensitive to the particular regularization strength. Comparing PR-S to EM, PR-S is always better, independent of the particular σ, with improvements ranging from 8% to 16%. The PR-AS constraints are also always better than EM for each model configuration and for all different parameter configurations. Note that the optimal parameter σ depends on the particular model configuration (V_s-V_c).

6.2.1 INSTABILITY WITH RESPECT TO σ

We can give a little more insight as to why we see some instability in the results with respect to the regularization strength. Figure 7 shows the accuracies on the English corpus broken down by POS tag category. The plot shows that sharp changes in overall accuracy are in fact caused by even sharper changes in the attachment accuracies of the tag categories. This should not be surprising, given that whether using EM or PR, the objective has many local maxima with deep valleys between them. The problem continues to be very underspecified, and without knowing the "true" sparsity pattern of a language, we can ultimately only achieve limited parsing accuracy.

Model	EM	PR						
	DMV							
		σ	80	100	120	140	160	180
2-1	45.8	PR-S	60.5	60.9	**62.0**	61.4	61.4	61.6
		PR-AS	53.8	54.3	55.3	54.3	54.6	54.6
V_s-V_c	E-DMV							
2-1	45.1	PR-S	60.7	59.9	61.3	61.6	62.1	60.2
		PR-AS	51.6	54.5	55.0	**62.4**	54.7	54.5
2-2	54.4	PR-S	**62.4**	57.1	57.8	57.6	57.1	58.8
		PR-AS	56.0	56.2	56.6	57.0	57.2	59.0
3-3	55.3	PR-S	59.3	60.8	60.0	62.2	**64.5**	64.1
		PR-AS	59.3	60.0	60.3	60.7	55.8	57.9
4-4	55.1	PR-S	59.4	61.2	61.6	63.9	**64.3**	63.6
		PR-AS	59.5	59.5	61.4	57.7	58.2	58.2

Table 3: Directed attachment accuracy results on the test corpus. Bold represents the best parameter setting for the DMV model and for each of the E-DMV models. The first column contains the V_s-V_c used. Columns represent different σ for both constraints PR-S on the left and PR-AS on the right.

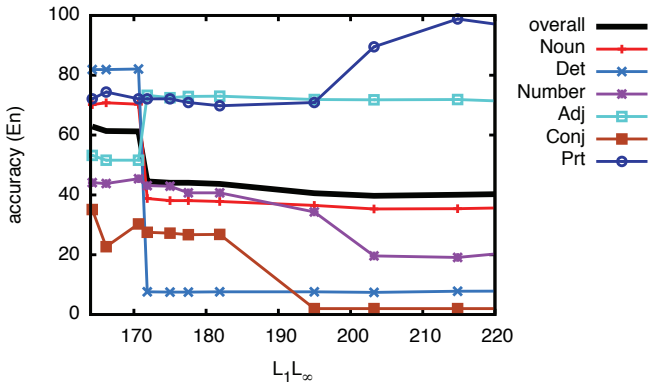


Figure 7: The accuracy overall and for different POS tag types in the English corpus as a function of ℓ_1/ℓ_∞ as we vary the constraint strength. EM has ℓ_1/ℓ_∞ of 431.17.

6.2.2 LEARNING CURVES

The top half of Figure 8 shows how accuracy and the various objective values change on a held-out development corpus for the DMV. (In all experiments, we held out the last 100 sentences of each training corpus for development; the numbers in Table 2 correspond to this reduced training set size. As we will discuss below they were unfortunately not reliable for picking hyperparameters.) First considering EM, we see that its accuracy is very stable after 20 iterations; its maximum value is at 80 iterations, but this is only marginally different from the value at 20 iterations. Its corresponding negative dev log likelihoood hits a minimum around 15 iterations, which correlates fairly well with

accuracy, but then negative dev log likelihood steadily increases after this. So, while dev likelihood would select a reasonable stopping point in this case, it can hardly be said to generally correlate well with accuracy. Next, considering SDP, we see its accuracy is mostly stagnant after 25 iterations, yet its negative dev log likelihood continues to steadily decrease long past iteration 25. Thus, the value of the objective on the dev set for SDP does not provide a way to select a good stopping point, nor does it correlate particularly well with accuracy. Finally, considering PR, we see slightly noisier accuracy curves that take a little longer to reach their maximums: around iteration 30 for PR-S and iteration 40 for PR-AS. The PR dev objective value curves matches the behavior of the accuracy curve fairly well and would select a good iteration for stopping. In summary, for the DMV, dev likelihood would not be a bad proxy for selecting stopping points for EM and PR.

However, the correlation is not as good when using the extended models, whose learning curves are shown in the bottom half of Figure 8. For example, both PR-S and PR-AS experience large jumps in accuracy that are not reflected in the likelihood curves. Thus, in the remainder of this work we do not attempt to select a stopping point based on dev likelihood, but rather simply run all experiments for 100 iterations.

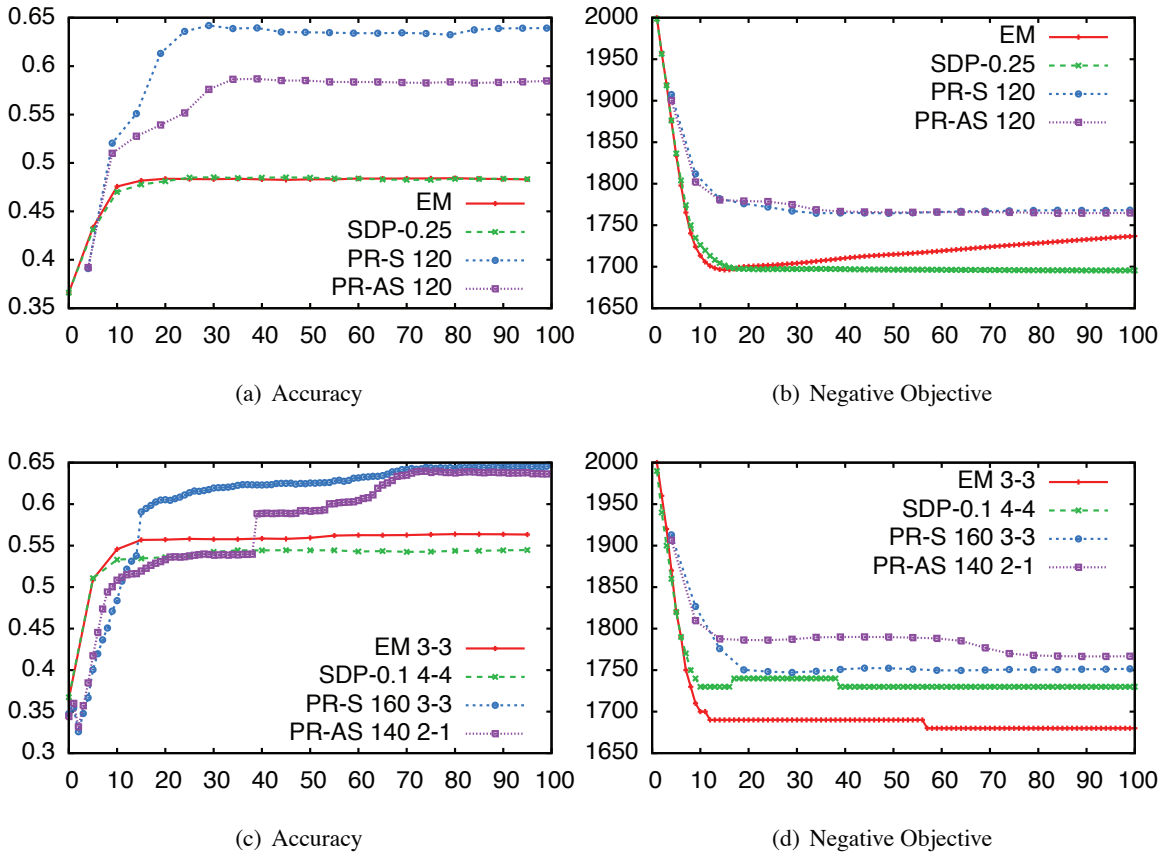
We also tried selecting a stopping point based on constituent contexts, motivated by Reichart and Rappoport (2009). Our hypothesis was that entropy of the distribution over contexts for each constituent should be small when parsing accuracy was high. However, comparing entropy of the gold trees to entropy of the trees produced by EM, this was only true for about half of the languages we tested on, and not strongly so for most of these. Also we note that we found no correlation between the PR objective on the development set and the best setting for the PR constraint strength, which does make it hard to pick this strength parameter in an unsupervised setting.

6.3 Comparison with Previous Work

Most results from previous work are not directly comparable due to differences in initialization, decoding method, or the incorporation of some degree of supervision. For this reason, we present the majority of the comparisons in Appendix B, where we also note implementation differences that we were able to determine. Here, we highlight the most salient accuracy numbers for the methods we mentioned in Section 5.

The best result reported thus far without additional lexical or multilingual information is that of Headden III et al. (2009). With a non-sparsifying Dirichlet prior and a learned (as opposed to constant) λ, they report an accuracy of 65.0 ($\pm$5.7)% for an an E-DMV of complexity $V_s = 2, V_c = 2$. (The $\pm$5.7 is a result of their use of a random pools initialization strategy.) We are able to achieve 64.5% accuracy with PR. We hypothesize that if PR were tested with random pools initialization and a learned λ, it would be able to make even further gains in accuracy. As noted in Appendix B, the learning of the smoothing parameter performed by Headden III et al. (2009) probably increases accuracy by about 5.5%. Similarly, Table 6 shows that random pools initialization tends to perform much better than the deterministic K&M initialization we use.

Other learning methods such as those discussed in Section 5 achieve slightly lower accuracies. We note that it is difficult however to make a complete comparison to them, as they operate only on the DMV model, not on any extended versions. Further, there are differences in the decoding method used. For example, the maximum accuracy achieved using shared logistic normal (SLN) priors with is 61.3% (Cohen and Smith, 2009). This is on the DMV model, where PR's maximum accuracy is a comparable 62%. But the SLN work uses MBR decoding and states its performance



(a) Accuracy (b) Negative Objective

(c) Accuracy (d) Negative Objective

Figure 8: Directed accuracy and the objective values on held-out development data as a function of the training iteration for the DMV (top) and E-DMV (bottom) with the best parameter settings.

is better than that of the Viterbi that we use. So, comparisons should be taken with a grain of salt. Comparing to contrastive estimation and annealing methods, accuracies are further below those of PR. With the DMV model and K&M initialization: CE is 48.7%, SDA is 46.7%, and SA is 51.5%. For a more extensive comparison to experimental results from related work, see Appendix B.

6.4 Results on Other Languages

A grammar induction algorithm is more interesting if it works on a variety of languages. Otherwise, the algorithm might just encode a lot of language-specific information. In this section, we compare several models and learning methods on twelve different languages to test their generalization capabilities. We do not want to assume that a user would have parsed corpora in each language, so we do not include a supervised search over model parameters for all languages as part of the evaluation process. Consequently, we use the following setup: for each model, basic DMV and the four E-DMV complexities we experimented with in the previous sections, pick the best configuration found for English according to its accuracy on the ≤ 10 test set, and use it across the other eleven

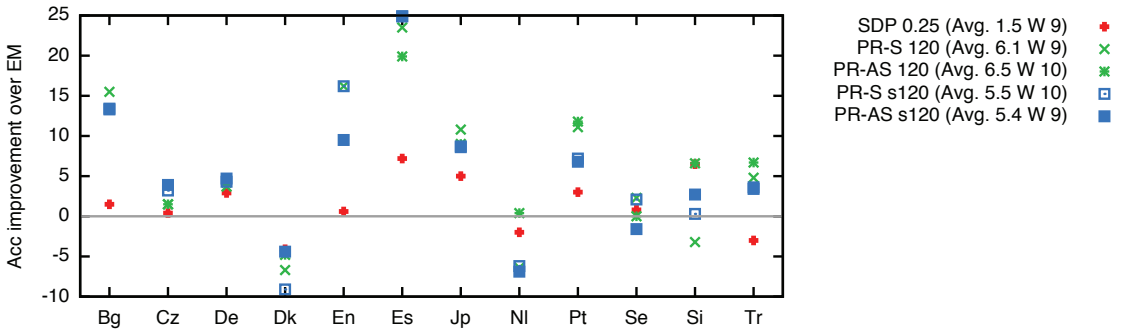


Figure 9: Difference in accuracy between the sparsity inducing training methods and EM training for the DMV model across the 12 languages. Avg: Average improvement over EM. W: Number of languages better than EM.

languages. This might not select the ideal parameters for any particular language, but provides a more realistic test setting: a user has available a labeled corpus in one language, and would like to induce grammars for other languages of interest.

For the PR approach, since the ideal strength is related to corpus size, we try two different approaches. The first is to use exactly the same strength with other languages as used for English. The second approach is to scale the strength by the number of tokens in each corpus. In this case, the strength, σ_x, for a particular language was found by the following formula: $\sigma_x = \sigma_{en} * |tokens_{en}|/|tokens_x|$, where σ_{en} is the best strength for English, $|tokens_{en}|$ is the number of tokens of the English corpus, and $|tokens_x|$ is the number of tokens in language x. This scaling is an approximation that attempts to require a similar amount of sparsity for each language.

For a table of exact accuracy numbers, we refer the reader to Table 7 in Appendix C. In this section we provide some figures illustrating the most salient aspects of the results from this table. Figure 9 illustrates the differences between the EM training and the different sparsity inducing training methods for the DMV. The zero line in Figure 9 corresponds to performance equal to EM. We see that the sparsifying methods tend to improve over EM most of the time. The average improvements are shown in the key of Figure 9. Figure 10 shows a similar comparison of the PR methods with respect to a SDP learning baseline. We see in Figure 10 that PR is better than SDP for most languages. Figure 11 compares the differences of each training method against EM training using the E-DMV model with the best setting found for English. Both PR-S and PR-AS perform better than EM in most cases. The average improvement is even bigger for PR-S than under the DMV, but PR-AS does not make such large gains. This is probably due to the selection of a simpler model for PR-AS (V_s-V_c= 2-1). While this simpler model performed better than the more complex ones for English, this does not generalize to all languages.

Figure 12 compares the different sparsity approaches. On the left we compare PR-S versus PR-AS without scaling on the DMV. PR-AS beats PR-S in 6 out of 12 cases and the two methods tie in one case (Czech). Over all 12 languages, the average difference between PR-AS and PR-S is only 3.2% on the DMV. We note that the difference is bigger for the E-DMV models, but this is possibly due to the selection of a simpler model (V_s-V_c = 2-1) for PR-AS. On the right side of the same figure, we compare PR-AS without scaling versus PR-AS with scaling. The unscaled

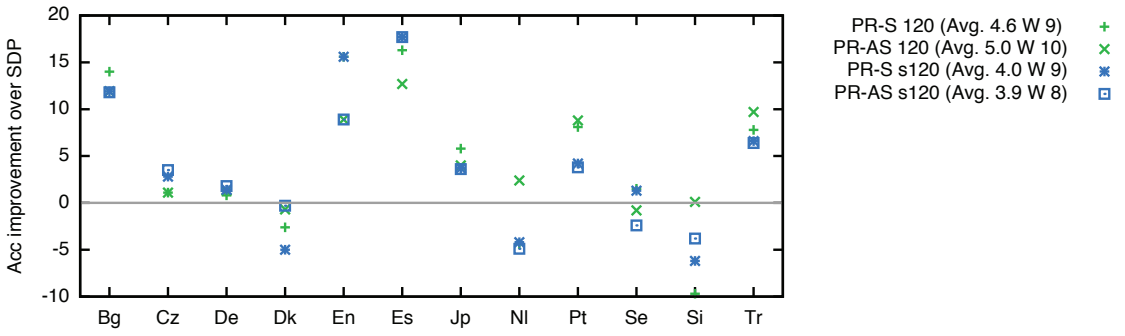


Figure 10: Difference in accuracy between PR training with the different constraints and SDP for the DMV model across the 12 languages. Avg: Average improvement over SDP. W: Number of languages better than SDP.

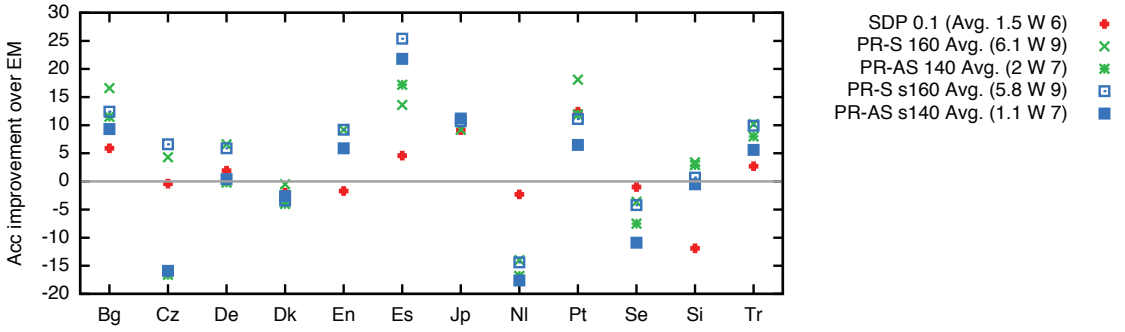
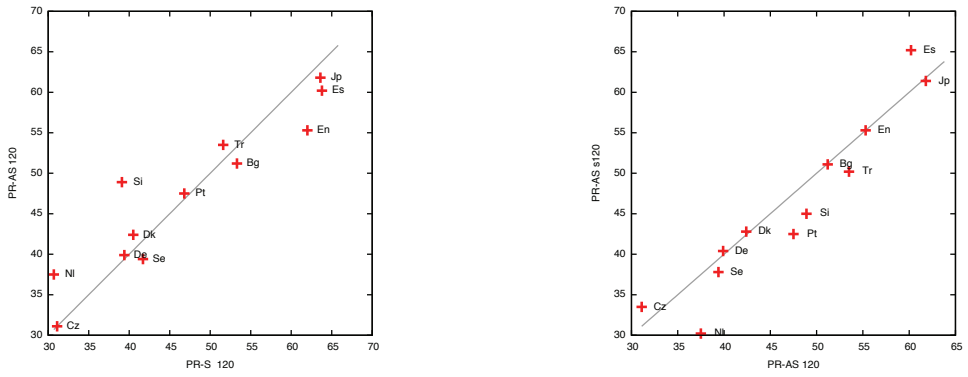


Figure 11: Difference in accuracy between the sparsity inducing training methods and EM training for the E-DMV model with the different training method across the 12 languages. Avg: Average improvement over EM. W: Number of languages better than EM.

version tends to perform better. In general, scaling that increases the constraint strength seems to be advantageous, the exception being for Dutch (Nl). Increased strength tends to correlate with increased runtime though, so there is a tradeoff to be made there.

Figure 13 compares the sparsity achieved by EM, SDP, and the PR methods on the DMV. We can see that the PR methods indeed achieve much greater sparsity than EM, and that SDP is only slightly more sparse than EM. If we also compared to supervised model initialization, most of the PR instances would have greater sparsity than the supervised, and EM and SDP would be much less sparse than the supervised. So, it seems that over-sparsifying is allowing us to achieve better accuracy than under-sparsifying. Although also not shown in the plot, we observe similar sparsity patterns on the test data as well.



(a) PR-S versus PR-AS

(b) No scaling versus scaling

Figure 12: Comparing the different sparsity constraints for the DMV model over twelve different languages. Left: PR-S vs PR-AS. Right: PR-AS without scaling vs PR-AS with scaling.

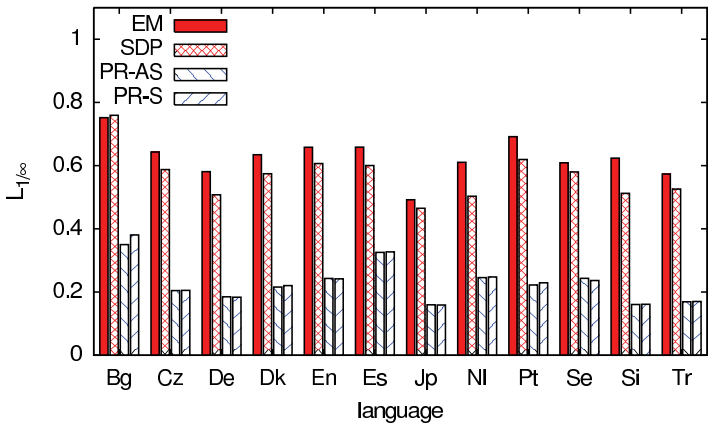


Figure 13: Comparing DMV grammar ambiguities on the training data by computing the average number of parent tags per child tag (ℓ_1/ℓ_∞ divided by number of child tags) and normalizing it by the theoretical maximum for each language. Grammar ambiguities from left to right within each group of bars are those resulting from: EM, SDP with $\alpha = 0.25$, PR-S with $\sigma = 120$, and PR-AS with $\sigma = 120$. Higher values imply less sparsity.

7. Analysis

Our accuracy numbers validate that PR is useful. In this section we attempt to analyze how and why it is useful, to validate our original claim that sparsity in parent-child types is the phenomenon we are capturing.

One common EM error that PR fixes in many languages is the directionality of the noun-determiner relation. Figure 14 shows an example of a Spanish sentence where PR significantly outperforms standard EM because of this fixed relation. As is evidenced in this case, EM frequently assigns a determiner as the parent of a noun, instead of the reverse. PR tends not to make this er-

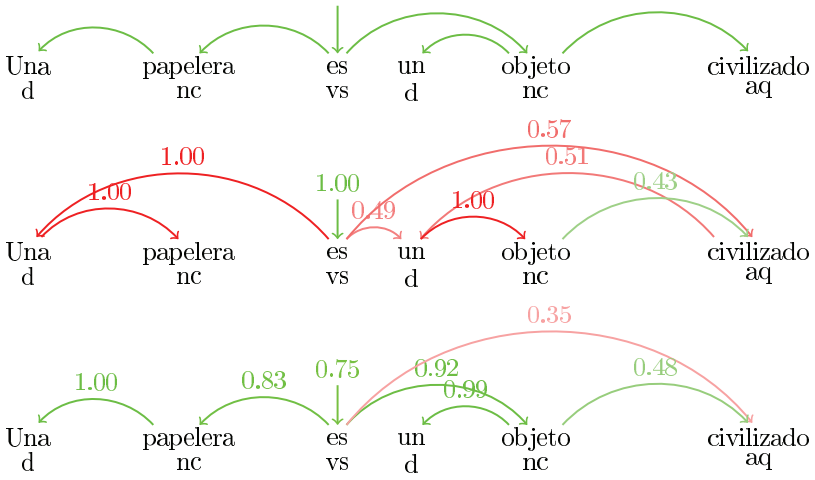


Figure 14: Posterior edge probabilities for an example sentence from the Spanish test corpus. **Top** is Gold, **middle** is EM, and **bottom** is PR.

ror. One explanation for this improvement is that it is a result of the fact that nouns can sometimes appear without determiners. For example, consider the sentence "Lleva tiempo entenderlos" (translation: "It takes time to understand (them)") with tags "main-verb common-noun main-verb". In this situation EM must assign the noun to a parent that is not a determiner. In contrast, when PR sees that sometimes nouns can appear without determiners but that the opposite situation does not occur, it shifts the model parameters to make nouns the parent of determiners instead of the reverse, since then it does not have to pay the cost of assigning a parent with a new tag to cover each noun that does not come with a determiner.

Table 4 contrasts the most frequent types of errors EM, SDP, and PR make on several test sets where PR does well. The "acc" column is accuracy and the "errs" column is the absolute number of errors of the key type. Accuracy for the key "parent POS truth/guess → child POS" is computed as a function of the true relation. So, if the key is $p_t/p_g \rightarrow c$, then accuracy is:

$$\text{acc} = \frac{\#\text{ of } p_t \rightarrow c \text{ in Viterbi parses}}{\#\text{ of } p_t \rightarrow c \text{ in gold parses}}.$$

In the following subsections we provide some analysis of the results from Table 4.

7.1 English Corrections

Considering English first, there are several notable differences between EM and PR errors. Similar to the example for Spanish, the direction of the noun-determiner relation is corrected by PR. This is reflected by the VB/DT → NN key, the NN/VBZ → DT key, the NN/IN → DT key, the IN/DT → NN key, the NN/VBD → DT key, the NN/VBP → DT key, and the NN/VB → DT key, which for EM and SDP have accuracy 0. PR corrects these errors.

A second correction PR makes is reflected in the VB/TO → VB key. One explanation for the reason PR is able to correctly identify VBs as the parents of other VBs instead of mistakenly making TO the parent of VBs is that "VB CC VB" is a frequently occurring sequence. For example, "build

	EM			SDP			PR		
	key	**acc**	**errs**	**key**	**acc**	**errs**	**key**	**acc**	**errs**
es	sp/d → nc	0.0	7	sp/d → nc	0.0	7	vm/<root> → vm	0.0	5
	nc/sp → d	0.0	6	nc/sp → d	0.0	6	<root>/vm → vm	0.0	4
	vm/d → nc	0.0	5	vm/<root> → vm	0.0	6	<root>/vm → vs	0.0	3
	vs/d → nc	0.0	4	nc/vm → d	0.0	6	rg/vm → rg	0.0	2
	vm/<root> → vm	0.0	4	vm/d → nc	0.0	5	aq/aq → cc	0.0	2
	nc/vm → d	0.0	4	<root>/vm → vm	0.0	4	nc/cc → aq	0.0	2
	aq/<root> → cc	0.0	3	vs/d → nc	0.0	4	vs/<root> → vm	0.0	2
	<root>/vm → vm	0.0	3	vm/p → rn	0.0	3	aq/nc → aq	0.0	2
	vm/p → rn	0.0	3	nc/vs → d	0.0	3	vm/vm → sp	75.0	2
	nc/vs → d	0.0	3	nc/<root> → d	0.0	3	vs/vm → cs	0.0	2
	vm/nc → sp	0.0	3	vm/nc → sp	0.0	3	vm/nc → sp	0.0	2
	vm/cs → vs	0.0	2	<root>/rg → vm	0.0	2	aq/cc → aq	0.0	1
	vm/d → p	0.0	2	nc/p → d	0.0	2	nc/vs → aq	0.0	1
	nc/aq → d	0.0	2	<root>/d → nc	0.0	2	<root>/aq → nc	0.0	1
	<root>/vm → vs	0.0	2	aq/cc → aq	0.0	2	vm/vm → cc	50.0	1
bg	<root>/R → V	0.0	65	N/V → R	0.0	53	N/V → R	0.0	56
	N/<root> → R	0.0	37	V/R → N	0.0	47	V/R → N	0.0	46
	V/<root> → R	0.0	29	<root>/C → V	0.0	26	T/V → V	0.0	26
	V/R → R	0.0	24	V/R → R	0.0	25	V/R → R	0.0	25
	N/M → N	0.0	20	T/V → V	0.0	23	V/V → T	42.4	19
	V/V → T	40.6	19	N/M → N	0.0	20	N/N → N	73.4	17
	<root>/C → V	0.0	18	V/V → T	42.4	19	V/V → N	84.8	14
	V/<root> → C	0.0	17	V/<root> → C	0.0	17	V/V → C	30.0	14
	T/V → N	0.0	17	N/<root> → C	0.0	15	T/V → N	0.0	13
	N/<root> → C	0.0	16	R/N → N	0.0	14	<root>/V → T	0.0	11
	V/R → N	0.0	16	T/V → N	0.0	13	N/V → V	0.0	10
	<root>/T → V	0.0	15	V/N → N	0.0	11	T/V → P	0.0	10
	N/V → R	0.0	15	N/R → N	0.0	10	N/N → M	66.7	10
	T/<root> → V	0.0	12	V/V → N	87.3	10	V/N → N	0.0	10
	R/N → N	0.0	12	N/V → V	0.0	10	<root>/V → V	0.0	9
pt	n/prp → art	0.0	39	n/prp → art	0.0	37	prp/v-fin → n	0.0	32
	v/art → n	0.0	31	v/art → n	0.0	32	n/prp → art	0.0	27
	prp/art → n	0.0	24	prp/art → n	0.0	27	v/n → prp	0.0	22
	n/v-fin → prp	0.0	18	n/v-fin → art	0.0	21	n/n → prp	0.0	20
	n/v-fin → art	0.0	17	v/v-fin → prp	72.5	11	v/prp → n	0.0	18
	v/pron-det → n	0.0	12	n/v-fin → prp	0.0	10	prp/v-fin → prop	0.0	11
	v/v-fin → prp	69.4	11	prop/prp → art	0.0	8	prp/prp → n	0.0	11
	v/prp → v	0.0	11	v/v-fin → adv	68.0	8	v/v-fin → adv	64.0	9
	prp/pron-det → n	0.0	10	prp/art → prop	0.0	7	prop/prp → art	0.0	8
	v/prp → prp	0.0	9	v/prp → v	0.0	7	v/v-fin → n	81.0	8
	prop/prp → art	0.0	8	v/prp → n	0.0	7	v/prop → prp	0.0	8
	n/v-fin → pron	0.0	8	<root>/conj-c → v	0.0	5	n/prop → prp	0.0	8
	n/prp → pron	0.0	8	v/<root> → v	0.0	5	v/v-fin → prp	58.8	7
	n/<root> → prp	0.0	8	v/art → prop	0.0	5	v/prp → v	0.0	7
	prp/art → prop	0.0	7	n/<root> → prp	0.0	5	<root>/prp → n	0.0	6
en	VB/DT → NN	0.0	129	VB/DT → NN	0.0	133	NN/NNP → NN	54.2	76
	NN/NNP → NN	60.1	65	NN/NNP → NN	54.7	78	IN/NN → NN	0.0	37
	NN/VBZ → DT	0.0	52	NN/IN → DT	0.0	56	MD/<root> → VB	0.0	25
	NN/IN → DT	0.0	47	NN/VBZ → DT	0.0	52	<root>/VB → MD	0.0	25
	IN/DT → NN	0.0	46	IN/DT → NN	0.0	46	IN/NNS → NN	0.0	24
	NN/VBD → DT	0.0	41	NN/VBD → DT	0.0	35	VB/NN → IN	0.0	21
	VB/TO → VB	0.0	19	VB/TO → VB	0.0	19	NN/NN → DT	86.5	21
	NN/VBP → DT	0.0	19	NN/VBP → DT	0.0	18	VB/DT → IN	0.0	20
	<root>/CD → NN	0.0	14	NN/NN → JJ	78.9	16	IN/VBD → NN	0.0	18
	NN/NN → JJ	81.1	14	VB/IN → JJ	0.0	12	NN/NN → JJ	79.2	16
	NN/VB → DT	0.0	14	VB/PRP$ → NN	0.0	12	IN/VBZ → NN	0.0	15
	NN/CD → CD	0.0	13	<root>/CD → NN	0.0	12	IN/VBP → NN	0.0	13
	VB/PRP$ → NN	0.0	12	NN/VB → DT	0.0	12	VB/VB → RB	18.8	13
	VB/DT → RB	0.0	11	NN/<root> → CD	0.0	11	NN/<root> → NN	0.0	11
	VB/<root> → VB	0.0	10	VB/NNS → RB	0.0	11	VB/NNS → NN	0.0	11

Table 4: Top 15 mistakes by parent POS truth/guess → child POS for English and the three languages where PR makes the greatest gains over EM with the E-DMV.

and hold" and "panic and bail" are two instances of the "VB CC VB" pattern from the test corpus. Presented with such scenarios, where there is no TO present to be the parent of VB, PR chooses the first VB as the parent of the second. It maintains this preference for making the first VB a parent of the second when encountered with "VB TO VB" sequences, such as "used to eliminate", because it would have to pay an additional penalty to make TO the parent of the second VB. In this manner, PR corrects the VB/TO → VB key error of EM and SDP.

A third correction PR makes is reflected in the <root>/CD → NN key. This correction is similar to the noun-determiner correction: CD and NN often co-occur, but while CD almost never appears without NN, NN frequently appears without CD. Thus, if PR chose CD as parent of NN, it would have to pay an additional penalty to select another parent for NN in sentences where no CDs exist. Thus, PR is able to recognize that CD is not usually a good parent for NN. Again, EM and SDP have 0 accuracy for this key.

There are a couple of errors common to EM, SDP, and PR. These correspond to the NN/NN → JJ key and the NN/NNP → NN key. These are notoriously difficult relations to get right, especially for an unlexicalized model that also has no notion of the surface lengths of relations. We predict that combining PR with a model such as the lexicalized DMV of Headden III et al. (2009), or applying the structural annealing technique of Smith and Eisner (2006), could greatly reduce these types of errors. These changes could also help reduce some of the other main errors PR makes, such as the ones corresponding to the keys NN/NN → DT and VB/VB → RB.

Even after all these improvements, there would likely persist at least one type of English error that would be hard to fix: the domination of modals by verbs. By convention, modals dominate verbs in English dependency parses. This is a relatively arbitrary choice, as there are linguistically sound arguments to be made for either dominating the other. In fact, in some of the other languages we work with the annotation convention is the reverse of what it is in English. Thus, for now we merely note that the keys MD/<root> → VB and <root>/VB → MD account for a large portion of the English errors with PR.

7.2 Bulgarian Corrections

Moving beyond English, we consider Bulgarian. We might expect qualitatively different results for Bulgarian for two reasons. First, the language is not in the same family as English. Second, the Bulgarian corpus employs far fewer POS tags.

One large correction PR makes with respect to EM and SDP corresponds to the key N/M → N. The tag M stands for "numeral" in the Bulgarian corpus, so this correction is similar to the English correction involving the tag CD. Another substantial correction PR makes with respect to EM and SDP corresponds to the key <root>/C → V. The tag C stands for "conjunction" in the Bulgarian corpus, so this correction means the model is realizing verbs should usually be sentence roots rather than children of conjunctions. Following the same reasoning about PR that we used before, we note that sentences with verbs but no conjunctions are very common, so if PR chose C as the parent of V, it would have to pay a penalty to give V a different parent in such sentences. The same reasoning explains why PR doesn't see the V/<root> → C errors or the N/<root> → C errors that EM and SDP do.

Although PR is able to make great improvements for Bulgarian parsing, it is clearly crippled by the small number of POS tags. EM, SDP, and PR all make substantial errors in deciding which verb to use as the parent of a particle (see key V/V → T), and many of the main remaining errors

for PR are caused by similar symmetries (see keys N/N $\rightarrow$ N, V/V $\rightarrow$ N, V/V $\rightarrow$ C, N/N $\rightarrow$ M, and <root>/V $\rightarrow$ V). As mentioned in the analysis of English, lexicalization or incorporation of a notion of surface length of relations might help alleviate these problems.

Corrections PR makes in the other languages can be analyzed using the same type of reasoning as we have applied to analysis of English and Bulgarian. We thus leave more extensive interpretation of Table 4 to the reader.

8. Conclusion

In this paper we presented a new method for unsupervised learning of dependency parsers. In contrast with previous approaches that impose a sparsity bias on the model parameters using sparsifying Dirichlet distributions, we impose a sparsity bias on the model posteriors. We do so by using the posterior regularization (PR) framework (Graça et al., 2007) with constraints that favor posterior distributions that have a small number of unique parent-child relations. We propose two such constraints: a symmetric constraint similar in spirit to the sparsity constraint applied to part-of-speech (POS) induction by Graça et al. (2009), and an asymmetric version of the same constraint that more directly tries to minimize the number of different parent-child types instead of different parent-child occurrences. On English our approach consistently outperforms the standard EM algorithm and the approach of training in a Bayesian setting where a sparsifying Dirichlet prior is used. Moreover, we perform an extensive comparison with previous published work and show that our learning approach achieves state-of-the-art results. We compare our approach on 11 additional languages, which as far as we know is the most extensive comparison made for a dependency parser. We report significant improvements over the competing learning approaches. The new approach improves over EM by an average of 6.5% and beats EM by at least 1% on 9 out of 12 languages. It also improves over the Bayesian learning approach by an average of 5% with gains of more than 1% for for 9 out of 12 languages.

One significant problem we encountered was picking the different parameters for the model in an unsupervised way, for which we found no good principled solution that worked for all languages. The PR objective on held-out development sets does not seem to be a reliable proxy for the model quality. Similarly, additional unsupervised measures for parse quality, motivated by the work of Reichart and Rappoport (2009) on counting constituent contexts, were unreliable. Even in the absence of a good unsupervised measure of model quality, a better method for transferring the regularization strength parameter from one language to another is also needed. The regularization strength is strongly dependent on the corpus, both on the number of parent-child pairs being constrained as well as on the number of tokens for each parent and child. Our experiments approximated this dependence by scaling the best English regularization strength by the number of tokens in other corpora, but this is not ideal.

With respect to model initialization, the K&M initialization is highly biased to the simple DMV model, and both RandomP initialization and the initialization approaches proposed by Spitkovsky et al. (2010) can significantly boost the performance of the model. It would be worth initializing our models with the techniques proposed by Spitkovsky et al. (2010), since they produce better results, are deterministic, and reduce the number of parameters that need to be tuned. Following the spirit of those approaches approaches, we also suggest that some success might be had by initializing the simple DMV training it, and then using its learned parameters to initialize more complex models (E-DMV models with larger valence values).

Regarding the sparsity constraints, we note that the versions we are using do not take into account some possibly important information, such as the directionality of the edge. Moreover, the same strength is currently used for the root probabilities and for the parent-child probabilities. Also, we could extend the constraints to work directly on word types rather than on POS tags, since there is a lot of information lost by discarding the particular words. For instance, Headden III et al. (2009) achieve significant improvements by conditioning the edge probabilities on the parent word together with the parent POS. Additionally, we could explore other constraints to encourage locality by preferring short dependency edges as suggested by the SA work of Smith (2006).

Finally, we would like in the future to move to fully unsupervised learning of grammar. That is, we would like to use POS tags induced in an unsupervised manner, instead of assuming gold POS tags, and see how robust our method is under these conditions. Recent studies show that the quality of the DMV model degrades significantly when the induced POS tags are used (Headden III et al., 2008). It would be interesting to see if our model is more robust to the quality of the provided tags. Further, it would be even more interesting to see how our method performs if we applied it to aid in the more complex task of joint induction of POS tags and dependency parses.

Acknowledgments

The authors would like to thank the anonymous reviewers for helpful comments. J. Gillenwater was partially supported by NSF-IGERT 0504487. K. Ganchev was supported by ARO MURI SUBTLE W911NF-07-1-0216. J. V. Graça was supported by a fellowship from Fundação para a Ciência e Tecnologia (SFRH/ BD/ 27528/ 2006) and by FCT project CMU-PT/HuMach/0039/2008. B. Taskar was partially supported by DARPA CSSG and ONR Young Investigator Award N000141010746.

Appendix A. Choosing the SDP Hyperparameter

We tried four different values for α: $\{0.01, 0.1, 0.25, 1\}$. (Note that the value 1 actually results in a non-sparsifying prior; this setting is not as good as the sparsifying, as Table 5 shows.)

Table 5 shows the directed accuracy for both the DMV and the E-DMV models trained using EM and SDP. We see in Table 5 that the extended model generally outperforms the DMV, for both EM and SDP. However, we also see that SDP does not always help: for all valences tried for the E-DMV except $(V_s, V_c) = (2,1)$, the EM models perform better. This contrasts with the findings of Headden III et al. (2009), potentially due to the simplified smoothing that we implemented, and a difference in the stopping criterion—we ran our model for 100 iterations, while Headden III et al. (2009) ran until likelihood on a held-out development set converged. Comparing the performance of the training methods, we see that for the DMV model, SDP training performs better and the best hyperparameter setting is 0.25 which is the same best parameter found by Cohen et al. (2008). The performance of our implementation of the SDP is slightly lower than the one reported in that paper, probably due to different stopping criteria during training.

Appendix B. Extended Comparison to Related Results

In this appendix we present a more extensive comparison between the performances of different models described in the literature for unsupervised dependency parsing. Table 6 presents the accuracy values reported in various previous papers and the values for approaches tried in this paper.

		SDP $\alpha =$			
	EM	1	0.25	0.1	0.01
DMV	45.8	42.2	**46.4**	45.2	45.4
2-1	45.1	42.0	46.0	45.9	44.9
2-2	**54.4**	42.0	43.3	52.5	51.5
3-3	55.3	42.8	47.1	53.5	52.1
4-4	55.1	42.9	47.1	53.6	51.7

Table 5: Directed attachment accuracy results on the test corpus (for sentences of lengths ≤ 10, no punctuation). The second column gives EM results, and the other columns are SDP results for different settings of the hyperparameter α. The second row is for the basic DMV model, and the other rows are E-DMV models represented by their valencies (V_s-V_c). Note that the 2-1 model is just the DMV plus smoothing of the child probabilities with $\lambda = 0.33$. Bold represents the best parameter setting both for the DMV model and the E-DMV model.

We would like to stress that the setup is not identical for all experiments. For instance, normally the stopping criteria for training is different. While we train all our models for 100 iterations, most other works use some kind of convergence criteria to stop training. Moreover, there are likely differences regarding other implementation details. The point of this section is mostly to highlight the many different variations of the DMV training and modeling that have been tried in the past. Table 6 is meant as a resource for comparing some of the best accuracies that these methods have achieved. It is hard to draw any sweeping conclusions from these numbers, but we hope that this summary of related work helps future work by suggesting reasonable choices for initialization, model complexity, smoothing, and other modeling decisions.

We start by comparing the effects of different initialization procedures. (See entries 1-6 in Table 6.) Although orthogonal to the learning procedure used, these differences are significant to keep in mind when comparing to previous work. We compare the results on the DMV. First we compare to work by Headden III et al. (2009) using random pools initialization. A random pool consists of a set of B randomly initialized models trained for a small number of iterations. From these B models, the one that assigns highest likelihood to held-out development data is picked and trained until convergence. M such pools are used to create M final models, whose mean accuracy and standard deviation are reported. We will refer to this initialization method as RandomP; it performs significantly better than K&M.

The other initializations compared in Table 6 are from recent work by Spitkovsky et al. (2010). These initialization methods aim to gradually increase the complexity of a model, as measured by the size of the search space, which for the DMV model is exponential in sentence length. The Baby Steps (BS) method starts by training the model on sentences of length 1, then the parameters of this model are used to initialize a training run over sentences of length 2, and so on. The second method, Less is More (LsM), uses information from the BS method to pick a sentence length that includes enough sentences to train a model with good predictive power, but leaves out longer sentences that do not add much information. A hybrid method Leapfrog (LP) combines the models from the two previous approaches. All of these methods also seem to improve over the K&M initialization.

We note that there are some differences in the setup of the various initialization experiments: the model initialized with RandomP described in Headden III et al. (2009) is trained using a Dirichlet prior with a hyperparameter of 1 (non-sparsifying DP), while all the other models are trained using EM. Additionally, the models from Spitkovsky et al. (2010) use a larger amount of data. Nonetheless, it seems likely that if we combined some of these initializations with out PR method, we would see even better performance than with the K&M setup that we use for simplicity in our current experiments.

The next comparison we make is between the smoothing approach described in Headden III et al. (2009) and the simpler implementation done in this work. Again, although the training methods and the initialization differs we see that the smoothing performed by Headden III et al. (2009) probably increases the accuracy of that model by around 5.5% over our implementation of smoothing (compare entry 2 to entry 7 and entry 1 to entry 8).

Entries 9 to 20 compare different training approaches for the basic DMV. Entry 9 corresponds to training the model with SDP with the best hyperparameter setting. Entries 10 and 11 correspond to training with PR under the two types of sparsity constraints. Entries 12 and 13 use the logistic normal prior (Cohen et al., 2008) and we report the results from the paper using Viterbi decoding. Entries 14, 15, 16, and 17 correspond to the different shared logistic normal priors (Cohen and Smith, 2009). These values are for MBR decoding since the authors do not report values for Viterbi decoding. This gives some advantage to these entries, since according to the authors MBR decoding always outperforms Viterbi decoding. Finally, entries 18, 19, and 20 represent the best value for the three learning approaches contrastive estimation (CE), skewed deterministic annealing (SDA), and structural annealing (SA) proposed by Smith (2006). For these entries we report the best values found using supervised selection of training parameters (several values were tried, and the one that produced the highest accuracy on the test data was selected). Out of all of these methods, the models trained using PR with the sparsity inducing constraints achieve the best results, the symmetric prior being the best. The results are similar to the best shared logistic normal prior when tested on sentences of length up to ten, but when tested on longer sentences the PR trained models perform significantly better then all other approaches.

The last block of results, entries 21 to 27, shows how a variety of learning methods compare on E-DMVs. Entries 21 to 24 compare our implementation of the three different learning approaches, EM, SDP, and PR with both types of constraints. Model selection in these cases is supervised, based on accuracy for the ≤ 10 test data. PR significantly outperforms the other two approaches. In particular the PR-S constraints perform the best with an average of 10% improvement over EM and SDP on sentences of lengths ≤ 10, and an even bigger improvement for longer sentences. In entries 25 to 27 we also compare with the original extended model of McClosky (2008) and with the smoothed extended model proposed by Headden III et al. (2009). The best model is the E-DMV with smoothing on the child probability as described by Headden III et al. (2009). It beats the E-DMV trained with PR-S by a small amount. This difference is much smaller than the gains from using the random initialization and the better smoothing distribution. Thus, we believe that training the same model with random initialization, better child probability smoothing, and the PR constraints would in fact produce the best results. We leave this as future work.

Finally we would like to note that Table 6 doesn't report results for the papers that use extra information. Namely, Headden III et al. (2009) reports the best result published so far, 68.8, for the test set with sentences of lengths ≤ 10, when using lexical information. Also, Cohen and Smith (2009) reports accuracies of 62.0, 48.0, and 42.2 for sentences of lengths ≤ 10, sentences of lengths

	Init	Training	Model	Directed			Undirected		
				≤ 10	≤ 20	all	≤ 10	≤ 20	all
Model Initialization									
1	K&M	EM	DMV	45.8	40.2	35.9	63.4	58.0	54.2
2	RandomP	DP	DMV	55.7 ($\pm$8.0)					
3	BS	Ad-Hoc @15	DMV	55.5	44.3	39.2			
4	BS	Ad-Hoc @45	DMV	55.1	44.4	39.4			
5	LsM	Ad-Hoc @15	DMV	56.2	48.2	44.1			
6	LP	Hybrid @45	DMV	**57.1**	**48.7**	**45.0**			
Smoothing effects									
7	RandomP	DP	DMV (λ learned)	**61.2** ($\pm$1.2)					
8	K&M	EM	DMV ($\lambda = 0.33$)	45.1	38.7	34.0	62.7	56.9	52.7
DMV									
9	K&M	SDP 0.25 *	DMV	46.4	40.9	36.5	64.0	58.6	54.8
10	K&M	PR-S 120 *	DMV	**62.0**	**53.8**	**48.9**	**69.8**	**62.4**	**58.2**
11	K&M	PR-AS 120 *	DMV	55.3	49.4	44.4	67.1	60.7	56.4
12	K&M	LN I	DMV	56.6	43.3	37.4			
13	K&M	LN families	DMV	59.3	45.1	39.0			
14	K&M	SLN Tie V	DMV	60.2	46.2	40.0			
15	K&M	SLN Tie N	DMV	60.2	46.7	40.9			
16	K&M	SLN Tie V & N	DMV	61.3	47.4	41.4			
17	K&M	SLN Tie A	DMV	59.9	45.8	40.9			
18	K&M	CE *	DMV	48.7			64.9		
19	K&M	SDA *	DMV	46.7			64.3		
20	K&M	SA *	DMV	51.5			67.9		
E-DMV									
21	K&M	EM	E-DMV(3,3) ($\lambda = 0.33$) *	55.3	46.4	42.6	69.0	61.9	58.3
22	K&M	SDP 0.1 *	E-DMV(4,4) ($\lambda = 0.33$) *	53.6	43.8	39.6	67.5	59.0	54.9
23	K&M	PR-S 160 *	E-DMV(3,3) ($\lambda = 0.33$) *	64.5	**54.6**	**49.5**	69.9	60.9	56.0
24	K&M	PR-AS 140 *	E-DMV(2,1) ($\lambda = 0.33$) *	62.2	53.2	48.5	**70.8**	**61.9**	**57.8**
25	K&M	EM	E-DMV(2,2)	56.5			69.7		
26	RandomP	DP	E-DMV(2,2)	53.3 ($\pm$7.1)					
27	RandomP	DP	E-DMV(2,2) (λ learned)	**65.0** ($\pm$5.7)					

Table 6: Comparison with previous published results. Results for entries 3, 4, 5, and 6 are taken from Spitkovsky et al. (2010), entries 2, 7, 26, and 27 are taken from Headden III et al. (2009), entry 25 is taken from McClosky (2008), entries 12 and 13 are taken from Cohen et al. (2008), entries 14, 15, 16, and 17 are taken from Cohen and Smith (2009) and entries 18, 19, and 20 are taken from Smith (2006). A star (*) in the training column indicates supervised selection of training parameters (PR regularization strength, SDP prior hyperparameter, etc.); a star in the model column indicates supervised selection of model complexity.

	Bg	Cz	De	Dk	En	Es	Jp	Nl	Pt	Se	Si	Tr	Avg
	DMV Model												
EM	37.8	29.6	35.7	**47.2**	45.8	40.3	52.8	37.1	35.7	39.4	42.3	46.8	40.9
SDP 0.25	39.3	30.0	38.6	43.1	46.4	47.5	57.8	35.1	38.7	40.2	48.8	43.8	42.4
PR-S 120	**53.3**	31.1	39.4	40.5	**62.0**	63.8	**63.6**	30.7	46.8	**41.7**	39.1	51.6	47.0
PR-AS 120	51.2	31.1	39.9	42.4	55.3	60.2	61.8	**37.5**	**47.5**	39.4	**48.9**	**53.5**	47.4
PR-S s120	51.2	32.8	40.0	38.1	**62.0**	**65.2**	61.5	30.9	42.9	41.5	42.6	50.4	46.6
PR-AS s120	51.1	**33.5**	**40.4**	42.8	55.3	**65.2**	61.4	30.2	42.5	37.8	45.0	50.2	46.3
	Extended Model												
EM-(3,3)	41.7	48.9	40.1	**46.4**	55.3	44.3	48.5	**47.5**	35.9	**48.6**	47.5	46.2	45.9
SDP-(4,4) 0.1	47.6	48.5	42.0	44.4	53.6	48.9	57.6	45.2	48.3	47.6	35.6	48.9	47.4
PR-S-(3,3) 160	**58.3**	53.2	**46.7**	45.9	**64.5**	57.9	57.7	33.5	**54.0**	45.0	**50.9**	**56.4**	52.0
PR-AS-(2,1) 140	53.2	32.3	39.9	42.4	61.2	61.5	59.6	30.7	47.8	41.1	50.4	54.2	47.9
PR-S-(3,3) s160	54.1	**55.5**	46.0	43.0	**64.5**	**69.7**	59.2	33.1	47.0	44.4	48.2	56.1	51.7
PR-AS-(2,1) s140	51.0	33.0	40.5	43.8	61.2	66.1	**59.7**	29.9	42.4	37.7	47.0	51.8	47.0
	Scaled Strengths												
English $\sigma = 120$	88	451	249	35	120	8	140	138	47	75	10	57	118
English $\sigma = 140$	103	526	290	41	140	9	163	161	55	88	11	67	138
English $\sigma = 160$	118	602	332	47	160	11	187	185	62	100	13	76	158

Table 7: Attachment accuracy results. For each method we tested both the basic DMV and the E-DMV. The parameters used where the best parameters found for English. For the extended model the child-valency and stop-valency used are indicated in parentheses. **EM**: The EM algorithm. **SDP**: Sparsifying Dirichlet prior. **PR-S**: Our method using the symmetric version of the constraints with strength parameter σ. **PR-S-s**: The same method but strength parameter scaled proportional to the number of tokens in the train set for each language. **PR-AS / PR-AS-s**: Our method with the asymmetric constraints, without and with scaling of the strength parameter. σ: The scaled weights for each corpus for the different values of the strength parameter used for English. Bold indicates the best method for each learning and model type.

≤ 20, and all sentences, respectively, when using multilingual information. This result for sentences of length ≤ 10 is equal to our best result, but is inferior to our results on longer sentences. Thus, we think that PR is a very promising technique for use with other data sets, where longer sentences are common.

Appendix C. Multilingual Results in Table Form

Table 7 shows the performance for all models and training procedures for the 12 different languages.

References

S. Afonso, E. Bick, R. Haber, and D. Santos. Floresta Sinta(c)tica: A treebank for Portuguese. In *Proc. LREC*, 2002.

K. Bellare, G. Druck, and A. McCallum. Alternating projections for learning with expectation constraints. In *Proc. UAI*, 2009.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.

A. Bohomovà, J. Hajic, E. Hajicova, and B. Hladka. The Prague dependency treebank: Three-level annotation scenario. In Anne Abeillé, editor, *Treebanks: Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers, 2001.

S. Brants, S. Dipper, S. Hansen, W. Lezius, and G. Smith. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, 2002.

M. Civit and M.A. Martí. Building cast3lb: A Spanish treebank. *Research on Language & Computation*, 2004.

S.B. Cohen and N.A. Smith. The shared logistic normal distribution for grammar induction. In *Proc. NAACL*, 2009.

S.B. Cohen, K. Gimpel, and N.A. Smith. Logistic normal priors for unsupervised probabilistic grammar induction. In *Proc. NIPS*, 2008.

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

S. Džeroski, T. Erjavec, N. Ledinek, P. Pajas, Z. Žabokrtsky, and A. Žele. Towards a Slovene dependency treebank. In *Proc. LREC*, 2006.

J. Finkel, T. Grenager, and C. Manning. The infinite tree. In *Proc. ACL*, 2007.

K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 2010.

J. Graça, K. Ganchev, and B. Taskar. Expectation maximization and posterior constraints. In *Proc. NIPS*, 2007.

J. Graça, K. Ganchev, B. Taskar, and F. Pereira. Posterior sparsity vs parameter sparsity. In *Proc. NIPS*, 2009.

J. Graça, K. Ganchev, and B. Taskar. Learning tractable word alignment models with complex constraints. *Computational Linguistics*, 2010.

W. Headden III, D. McClosky, and E. Charniak. Evaluating unsupervised part-of-speech tagging for grammar induction. In *Proc. CoNLL*, 2008.

W.P. Headden III, M. Johnson, and D. McClosky. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proc. NAACL*, 2009.

M. Johnson, T.L. Griffiths, and S. Goldwater. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *Proc. NIPS*, 2007.

Y. Kawata and J. Bartels. Stylebook for the Japanese treebank in VERBMOBIL. Technical report, Eberhard-Karls-Universitat Tubingen, 2000.

D. Klein and C. Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. ACL*, 2004.

M.T. Kromann, L. Mikkelsen, and S.K. Lynge. Danish dependency treebank. In *Proc. TLT*, 2003.

K. Kurihara and T. Sato. An application of the variational Bayesian approach to probabilistic context-free grammars. In *IJC-NLP Workshop: Beyond Shallow Analyses*, 2004.

P. Liang, S. Petrov, M.I. Jordan, and D. Klein. The infinite PCFG using hierarchical Dirichlet processes. In *Proc. EMNLP*, 2007.

P. Liang, M.I. Jordan, and D. Klein. Learning from measurements in exponential families. In *Proc. ICML*, 2009.

G. Mann and A. McCallum. Simple, robust, scalable semi-supervised learning via expectation regularization. In *Proc. ICML*, 2007.

G. Mann and A. McCallum. Generalized expectation criteria for semi-supervised learning of conditional random fields. In *Proc. ACL*, 2008.

M. Marcus, M. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 1993.

D. McClosky. Modeling valence effects in unsupervised grammar induction. Technical report, CS-09-01, Brown University, 2008.

R. Neal and G. Hinton. A new view of the EM algorithm that justifies incremental, sparse and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1998.

J. Nilsson and J. Hall, J.and Nivre. MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. *NODALIDA Special Session on Treebanks*, 2005.

K. Oflazer, B. Say, D.Z. Hakkani-Tür, and G. Tür. Building a Turkish treebank. *Treebanks: Building and Using Parsed Corpora*, 2003.

S. Ravi, J. Baldridge, and K. Knight. Minimized models and grammar-informed initialization for supertagging with highly ambiguous lexicons. In *Proc. ACL*, 2010.

R. Reichart and A. Rappoport. Automatic selection of high quality parses created by a fully unsupervised parser. In *Proc. CoNLL*, 2009.

K. Simov, P. Osenova, M. Slavcheva, S. Kolkovska, E. Balabanova, D. Doikoff, K. Ivanova, A. Simov, E. Simov, and M. Kouylekov. Building a linguistically interpreted corpus of Bulgarian: the BulTreebank. In *Proc. LREC*, 2002.

N. Smith. *Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text*. PhD thesis, Johns Hopkins University, 2006.

N. Smith and J. Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. IJC-AI Workshop: Grammatical Inference Applications*, 2005a.

N. Smith and J. Eisner. Guiding unsupervised grammar induction using contrastive estimation. In *Proc. IJC-AI Workshop: Grammatical Inference Applications*, 2005b.

N. Smith and J. Eisner. Annealing structural bias in multilingual weighted grammar induction. In *Proc. ACL*, 2006.

V. I. Spitkovsky, H. Alshawi, and D. Jurafsky. From baby steps to leapfrog: How "less is more" in unsupervised dependency parsing. In *Proc. NAACL-HLT*, 2010.

P. Tseng. An analysis of the EM algorithm and entropy-like proximal point methods. *Mathematics of Operations Research*, 29(1):27–44, 2004.

L. Van der Beek, G. Bouma, R. Malouf, and G. Van Noord. The Alpino dependency treebank. *Language and Computers*, 2002.

Journal of Machine Learning Research 12 (2011) 491-523 Submitted 6/10; Published 2/11

Learning Multi-modal Similarity

Brian McFee BMCFEE@CS.UCSD.EDU
Department of Computer Science and Engineering
University of California
San Diego, CA 92093-0404, USA

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

Editor: Tony Jebara

Abstract

In many applications involving multi-media data, the definition of similarity between items is integral to several key tasks, including nearest-neighbor retrieval, classification, and recommendation. Data in such regimes typically exhibits multiple modalities, such as acoustic and visual content of video. Integrating such heterogeneous data to form a holistic similarity space is therefore a key challenge to be overcome in many real-world applications.

We present a novel multiple kernel learning technique for integrating heterogeneous data into a single, unified similarity space. Our algorithm learns an optimal ensemble of kernel transformations which conform to measurements of human perceptual similarity, as expressed by relative comparisons. To cope with the ubiquitous problems of subjectivity and inconsistency in multi-media similarity, we develop graph-based techniques to filter similarity measurements, resulting in a simplified and robust training procedure.

Keywords: multiple kernel learning, metric learning, similarity

1. Introduction

In applications such as content-based recommendation systems, the definition of a proper similarity measure between items is crucial to many tasks, including nearest-neighbor retrieval and classification. In some cases, a natural notion of similarity may emerge from domain knowledge, for example, cosine similarity for bag-of-words models of text. However, in more complex, multi-media domains, there is often no obvious choice of similarity measure. Rather, viewing different aspects of the data may lead to several different, and apparently equally valid notions of similarity. For example, if the corpus consists of musical data, each song or artist may be represented simultaneously by acoustic features (such as rhythm and timbre), semantic features (tags, lyrics), or social features (collaborative filtering, artist reviews and biographies, etc). Although domain knowledge may be incorporated to endow each representation with an intrinsic geometry—and, therefore, a sense of similarity—the different notions of similarity may not be mutually consistent. In such cases, there is generally no obvious way to combine representations to form a unified similarity space which optimally integrates heterogeneous data.

©2011 Brian McFee and Gert Lanckriet.

Without extra information to guide the construction of a similarity measure, the situation seems hopeless. However, if some side-information is available, for example, as provided by human labelers, it can be used to formulate a learning algorithm to optimize the similarity measure.

This idea of using side-information to optimize a similarity function has received a great deal of attention in recent years. Typically, the notion of similarity is captured by a distance metric over a vector space (e.g., Euclidean distance in $\mathbb{R}^d$), and the problem of optimizing similarity reduces to finding a suitable embedding of the data under a specific choice of the distance metric. *Metric learning* methods, as they are known in the machine learning literature, can be informed by various types of side-information, including class labels (Xing et al., 2003; Goldberger et al., 2005; Globerson and Roweis, 2006; Weinberger et al., 2006), or binary *similar*/*dissimilar* pairwise labels (Wagstaff et al., 2001; Shental et al., 2002; Bilenko et al., 2004; Globerson and Roweis, 2007; Davis et al., 2007). Alternatively, multidimensional scaling (MDS) techniques are typically formulated in terms of quantitative (dis)similarity measurements (Torgerson, 1952; Kruskal, 1964; Cox and Cox, 1994; Borg and Groenen, 2005). In these settings, the representation of data is optimized so that distance (typically Euclidean) conforms to side-information. Once a suitable metric has been learned, similarity to new, unseen data can be computed either directly (if the metric takes a certain parametric form, for example, a linear projection matrix), or via out-of-sample extensions (Bengio et al., 2004).

To guide the construction of a similarity space for multi-modal data, we adopt the idea of using similarity measurements, provided by human labelers, as side-information. However, it has to be noted that, especially in heterogeneous, multi-media domains, similarity may itself be a highly subjective concept and vary from one labeler to the next (Ellis et al., 2002). Moreover, a single labeler may not be able to consistently decide if or to what extent two objects are similar, but she may still be able to reliably produce a rank-ordering of similarity over pairs (Kendall and Gibbons, 1990). Thus, rather than rely on quantitative similarity or hard binary labels of pairwise similarity, it is now becoming increasingly common to collect similarity information in the form of triadic or *relative* comparisons (Schultz and Joachims, 2004; Agarwal et al., 2007), in which human labelers answer questions of the form:

"Is x more similar to y or z?"

Although this form of similarity measurement has been observed to be more stable than quantitative similarity (Kendall and Gibbons, 1990), and clearly provides a richer representation than binary pairwise similarities, it is still subject to problems of consistency and inter-labeler agreement. It is therefore imperative that great care be taken to ensure some sense of robustness when working with perceptual similarity measurements.

In the present work, our goal is to develop a framework for integrating multi-modal data so as to optimally conform to perceptual similarity encoded by relative comparisons. In particular, we follow three guiding principles in the development of our framework:

1. The algorithm should be robust against subjectivity and inter-labeler disagreement.
2. The algorithm must be able to integrate multi-modal data in an optimal way, that is, the distances between embedded points should conform to perceptual similarity measurements.
3. It must be possible to compute distances to new, unseen data as it becomes available.

We formulate this problem of heterogeneous feature integration as a learning problem: given a data set, and a collection of relative comparisons between pairs, we learn a representation of

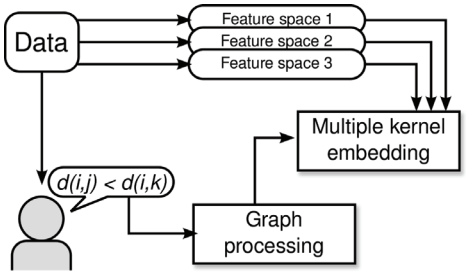


Figure 1: An overview of our proposed framework for multi-modal feature integration. Data is represented in multiple feature spaces (each encoded by a kernel function). Humans supply perceptual similarity measurements in the form of relative pairwise comparisons, which are in turn filtered by graph processing algorithms, and then used as constraints to optimize the multiple kernel embedding.

the data that optimally reproduces the similarity measurements. This type of embedding problem has been previously studied by Agarwal et al. (2007) and Schultz and Joachims (2004). However, Agarwal et al. (2007) provide no out-of-sample extension, and neither support heterogeneous feature integration, nor do they address the problem of noisy similarity measurements.

A common approach to optimally integrate heterogeneous data is based on *multiple kernel learning*, where each kernel encodes a different modality of the data. Heterogeneous feature integration via multiple kernel learning has been addressed by previous authors in a variety of contexts, including classification (Lanckriet et al., 2004; Zien and Ong, 2007; Kloft et al., 2009; Jagarlapudi et al., 2009), regression (Sonnenburg et al., 2006; Bach, 2008; Cortes et al., 2009), and dimensionality reduction (Lin et al., 2009). However, none of these methods specifically address the problem of learning a unified data representation which conforms to perceptual similarity measurements.

1.1 Contributions

Our contributions in this work are two-fold. First, we develop the *partial order embedding* (POE) framework (McFee and Lanckriet, 2009b), which allows us to use graph-theoretic algorithms to filter a collection of subjective similarity measurements for consistency and redundancy. We then formulate a novel multiple kernel learning (MKL) algorithm which learns an ensemble of feature space projections to produce a unified similarity space. Our method is able to produce non-linear embedding functions which generalize to unseen, out-of-sample data. Figure 1 provides a high-level overview of the proposed methods.

The remainder of this paper is structured as follows. In Section 2, we develop a graphical framework for interpreting and manipulating subjective similarity measurements. In Section 3, we derive an embedding algorithm which learns an optimal transformation of a single feature space. In Section 4, we develop a novel multiple-kernel learning formulation for embedding problems, and derive an algorithm to learn an optimal space from heterogeneous data. Section 5 provides experimental results illustrating the effects of graph-processing on noisy similarity data, and the effectiveness of the multiple-kernel embedding algorithm on a music similarity task with human

perception measurements. Finally, we prove hardness of dimensionality reduction in this setting in Section 6, and conclude in Section 7.

1.2 Preliminaries

A *(strict) partial order* is a binary relation R over a set Z ($R \subseteq Z^2$) which satisfies the following properties:[1]

- Irreflexivity: $(a,a) \notin R$,
- Transitivity: $(a,b) \in R \wedge (b,c) \in R \Rightarrow (a,c) \in R$,
- Anti-symmetry: $(a,b) \in R \Rightarrow (b,a) \notin R$.

Every partial order can be equivalently represented as a directed acyclic graph (DAG), where each vertex is an element of Z and an edge is drawn from a to b if $(a,b) \in R$. For any partial order, R may refer to either the set of ordered tuples $\{(a,b)\}$ or the graph (DAG) representation of the partial order; the use will be clear from context.

For a directed graph G, we denote by G^∞ its *transitive closure*, that is, G^∞ contains an edge (i,j) if and only if there exists a path from i to j in G. Similarly, the *transitive reduction* (denoted $G^{\min}$) is the minimal graph with equivalent transitivity to G, that is, the graph with the fewest edges such that $\left(G^{\min}\right)^\infty = G^\infty$.

Let $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ denote the training set of n items. A *Euclidean embedding* is a function $g : \mathcal{X} \to \mathbb{R}^d$ which maps $\mathcal{X}$ into a d-dimensional space equipped with the Euclidean (ℓ_2) metric:

$$\|x-y\|_2 = \sqrt{(x-y)^\mathsf{T}(x-y)}.$$

For any matrix B, let B_i denote its i^{th} column vector. A symmetric matrix $A \in \mathbb{R}^{n \times n}$ has a spectral decomposition $A = V \Lambda V^\mathsf{T}$, where $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a diagonal matrix containing the eigenvalues of A, and V contains the eigenvectors of A. We adopt the convention that eigenvalues (and corresponding eigenvectors) are sorted in descending order. A is *positive semi-definite* (PSD), denoted by $A \succeq 0$, if each eigenvalue is non-negative: $\lambda_i \geq 0,\ i = 1, \ldots, n$. Finally, a PSD matrix A gives rise to the Mahalanobis distance function

$$\|x-y\|_A = \sqrt{(x-y)^\mathsf{T} A (x-y)}.$$

2. A Graphical View of Similarity

Before we can construct an embedding algorithm for multi-modal data, we must first establish the form of side-information that will drive the algorithm, that is, the similarity measurements that will be collected from human labelers. There is an extensive body of work on the topic of constructing a geometric representation of data to fit perceptual similarity measurements. Primarily, this work falls under the umbrella of multi-dimensional scaling (MDS), in which perceptual similarity is modeled by numerical responses corresponding to the perceived "distance" between a pair of items, for

1. The standard definition of a (non-strict) partial order also includes the *reflexive* property: $\forall a, (a,a) \in \mathbb{R}$. For reasons that will become clear in Section 2, we take the *strict* definition here, and omit the reflexive property.

example, on a similarity scale of 1–10. (See Cox and Cox 1994 and Borg and Groenen 2005 for comprehensive overviews of MDS techniques.)

Because "distances" supplied by test subjects may not satisfy metric properties—in particular, they may not correspond to Euclidean distances—alternative *non-metric* MDS (NMDS) techniques have been proposed (Kruskal, 1964). Unlike classical or metric MDS techniques, which seek to preserve quantitative distances, NDMS seeks an embedding in which the rank-ordering of distances is preserved.

Since NMDS only needs the rank-ordering of distances, and not the distances themselves, the task of collecting similarity measurements can be simplified by asking test subjects to order pairs of points by similarity:

"Are i and j more similar than k and ℓ?"

or, as a special case, the "triadic comparison"

"Is i more similar to j or ℓ?"

Based on this kind of *relative comparison* data, the embedding problem can be formulated as follows. Given is a set of objects $\mathcal{X}$, and a set of similarity measurements $\mathcal{C} = \{(i,j,k,\ell)\} \subseteq \mathcal{X}^4$, where a tuple (i,j,k,ℓ) is interpreted as "i and j are more similar than k and ℓ." (This formulation subsumes the triadic comparisons model when $i = k$.) The goal is to find an embedding function $g : \mathcal{X} \to \mathbb{R}^d$ such that

$$\forall (i,j,k,\ell) \in \mathcal{C} : \ \|g(i) - g(j)\|^2 + 1 < \|g(k) - g(\ell)\|^2. \tag{1}$$

The unit margin is forced between the constrained distances for numerical stability.

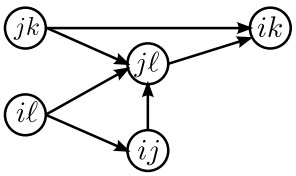
Agarwal et al. (2007) work with this kind of relative comparison data and describe a generalized NMDS algorithm (GNMDS), which formulates the embedding problem as a semi-definite program. Schultz and Joachims (2004) derive a similar algorithm which solves a quadratic program to learn a linear, axis-aligned transformation of data to fit relative comparisons.

Previous work on relative comparison data often treats each measurement $(i,j,k,\ell) \in \mathcal{C}$ as effectively independent (Schultz and Joachims, 2004; Agarwal et al., 2007). However, due to their semantic interpretation as encoding pairwise similarity comparisons, and the fact that a pair (i,j) may participate in several comparisons with other pairs, there may be some *global* structure to $\mathcal{C}$ which these previous methods are unable to exploit.

In Section 2.1, we develop a graphical framework to infer and interpret the global structure exhibited by the constraints of the embedding problem. Graph-theoretic algorithms presented in Section 2.2 then exploit this representation to filter this collection of noisy similarity measurements for consistency and redundancy. The final, reduced set of relative comparison constraints defines a partial order, making for a more robust and efficient embedding problem.

2.1 Similarity Graphs

To gain more insight into the underlying structure of a collection of comparisons $\mathcal{C}$, we can represent $\mathcal{C}$ as a directed graph over $\mathcal{X}^2$. Each vertex in the graph corresponds to a pair $(i,j) \in \mathcal{X}^2$, and an edge from (i,j) to (k,ℓ) corresponds to a similarity measurement (i,j,k,ℓ) (see Figure 2). Interpreting $\mathcal{C}$ as a graph will allow us to infer properties of *global* (graphical) structure of $\mathcal{C}$. In particular, two facts become immediately apparent:



$$\mathcal{C} = \left\{ \begin{array}{ll} (j,k,j,\ell), & (j,k,i,k), \\ (j,\ell,i,k), & (i,\ell,j,\ell), \\ (i,\ell,i,j), & (i,j,j,\ell) \end{array} \right\}$$

Figure 2: The graph representation (left) of a set of relative comparisons (right).

1. If $\mathcal{C}$ contains cycles, then there exists no embedding which can satisfy $\mathcal{C}$.

2. If $\mathcal{C}$ is acyclic, any embedding that satisfies the transitive reduction $\mathcal{C}^{\min}$ also satisfies $\mathcal{C}$.

The first fact implies that no algorithm can produce an embedding which satisfies all measurements if the graph is cyclic. In fact, the converse of this statement is also true: if $\mathcal{C}$ is acyclic, then an embedding exists in which all similarity measurements are preserved (see Appendix A). If $\mathcal{C}$ is cyclic, however, by analyzing the graph, it is possible to identify an "unlearnable" subset of $\mathcal{C}$ which must be violated by any embedding.

Similarly, the second fact exploits the transitive nature of distance comparisons. In the example depicted in Figure 2, any g that satisfies (j,k,j,ℓ) and (j,ℓ,i,k) must also satisfy (j,k,i,k). In effect, the constraint (j,k,i,k) is redundant, and may also be safely omitted from $\mathcal{C}$.

These two observations allude to two desirable properties in $\mathcal{C}$ for embedding methods: *transitivity* and *anti-symmetry*. Together with irreflexivity, these fit the defining characteristics of a *partial order*. Due to subjectivity and inter-labeler disagreement, however, most collections of relative comparisons will not define a partial order. Some graph processing, presented next, based on an approximate maximum acyclic subgraph algorithm, can reduce them to a partial order.

2.2 Graph Simplification

Because a set of similarity measurements $\mathcal{C}$ containing cycles cannot be embedded in any Euclidean space, $\mathcal{C}$ is inherently inconsistent. Cycles in $\mathcal{C}$ therefore constitute a form of *label noise*. As noted by Angelova (2004), label noise can have adverse effects on both model complexity and generalization. This problem can be mitigated by detecting and pruning noisy (confusing) examples, and training on a reduced, but certifiably "clean" set (Angelova et al., 2005; Vezhnevets and Barinova, 2007).

Unlike most settings, where the noise process affects each label independently—for example, random classification noise (Angluin and Laird, 1988)—the graphical structure of interrelated relative comparisons can be exploited to detect and prune inconsistent measurements. By eliminating similarity measurements which cannot be realized by any embedding, the optimization procedure can be carried out more efficiently and reliably on a reduced constraint set.

Ideally, when eliminating edges from the graph, we would like to retain as much information as possible. Unfortunately, this is equivalent to the *maximum acyclic subgraph* problem, which is NP-Complete (Garey and Johnson, 1979). A $1/2$-approximate solution can be achieved by a simple greedy algorithm (Algorithm 1) (Berger and Shor, 1990).

Once a consistent subset of similarity measurements has been produced, it can be simplified further by pruning redundancies. In the graph view of similarity measurements, redundancies can be easily removed by computing the transitive reduction of the graph (Aho et al., 1972).

Algorithm 1 Approximate maximum acyclic subgraph (Aho et al., 1972)

Input: Directed graph $G = (V, E)$
Output: Acyclic graph G'
$E' \leftarrow \emptyset$
for each $(u, v) \in E$ in random order **do**
  if $E' \cup \{(u, v)\}$ is acyclic **then**
    $E' \leftarrow E' \cup \{(u, v)\}$
  end if
end for
$G' \leftarrow (V, E')$

By filtering the constraint set for consistency, we ensure that embedding algorithms are not learning from spurious information. Additionally, pruning the constraint set by transitive reduction focuses embedding algorithms on the most important core set of constraints while reducing overhead due to redundant information.

3. Partial Order Embedding

Now that we have developed a language for expressing similarity between items, we are ready to formulate the embedding problem. In this section, we develop an algorithm that learns a representation of data consistent with a collection of relative similarity measurements, and allows to map unseen data into the learned similarity space after learning. In order to accomplish this, we will assume a feature representation for $\mathcal{X}$. By parameterizing the embedding function g in terms of the feature representation, we will be able to apply g to any point in the feature space, thereby generalizing to data outside of the training set.

3.1 Linear Projection

To start, we assume that the data originally lies in some Euclidean space, that is, $\mathcal{X} \subset \mathbb{R}^D$. There are of course many ways to define an embedding function $g : \mathbb{R}^D \to \mathbb{R}^d$. Here, we will restrict attention to embeddings parameterized by a linear projection matrix M, so that for a vector $x \in \mathbb{R}^D$,

$$g(x) \doteq Mx.$$

Collecting the vector representations of the training set as columns of a matrix $X \in \mathbb{R}^{D \times n}$, the inner product matrix of the embedded points can be characterized as

$$A = X^\mathsf{T} M^\mathsf{T} M X.$$

Now, for a relative comparison (i, j, k, ℓ), we can express the distance constraint (1) between embedded points as follows:

$$(X_i - X_j)^\mathsf{T} M^\mathsf{T} M (X_i - X_j) + 1 \leq (X_k - X_\ell)^\mathsf{T} M^\mathsf{T} M (X_k - X_\ell).$$

These inequalities can then be used to form the constraint set of an optimization problem to solve for M. Because, in general, $\mathcal{C}$ may not be satisfiable by a linear projection of $\mathcal{X}$, we soften the

constraints by introducing a slack variable $\xi_{ijk\ell} \geq 0$ for each constraint, and minimize the empirical hinge loss over constraint violations $1/|\mathcal{C}| \sum_{\mathcal{C}} \xi_{ijk\ell}$. This choice of loss function can be interpreted as a convex approximation to a generalization of the area under an ROC curve (see Appendix C).

To avoid over-fitting, we introduce a regularization term $\mathrm{tr}(M^{\mathsf{T}}M)$, and a trade-off parameter $\beta > 0$ to control the balance between regularization and loss minimization. This leads to a regularized risk minimization objective:

$$\begin{aligned} \min_{M,\xi \geq 0} \quad & \mathrm{tr}(M^{\mathsf{T}}M) + \frac{\beta}{|\mathcal{C}|} \sum_{\mathcal{C}} \xi_{ijk\ell} \\ \text{s.t.} \quad & (X_i - X_j)^{\mathsf{T}} M^{\mathsf{T}} M (X_i - X_j) + 1 \leq (X_k - X_\ell)^{\mathsf{T}} M^{\mathsf{T}} M (X_k - X_\ell) + \xi_{ijk\ell}, \\ & \forall (i,j,k,\ell) \in \mathcal{C}. \end{aligned} \tag{2}$$

After learning M by solving this optimization problem, the embedding can be extended to out-of-sample points x' by applying the projection: $x' \mapsto Mx'$.

Note that the distance constraints in (2) involve differences of quadratic terms, and are therefore not convex. However, since M only appears in the form $M^{\mathsf{T}}M$ in (2), the optimization problem can be expressed in terms of a positive semi-definite matrix $W \doteq M^{\mathsf{T}}M$. This change of variables results in Algorithm 2, a (convex) semi-definite programming (SDP) problem (Boyd and Vandenberghe, 2004), since objective and constraints are linear in W, including the linear matrix inequality $W \succeq 0$. The corresponding inner product matrix is

$$A = X^{\mathsf{T}} W X.$$

Finally, after the optimal W is found, the embedding function $g : \mathbb{R}^D \to \mathbb{R}^D$ can be recovered from the spectral decomposition of W:

$$W = V \Lambda V^{\mathsf{T}} \quad \Rightarrow \quad g(x) = \Lambda^{1/2} V^{\mathsf{T}} x,$$

and a d-dimensional approximation can be recovered by taking the leading d eigenvectors of W.

Algorithm 2 Linear partial order embedding (LPOE)

Input: n objects $\mathcal{X}$,
partial order $\mathcal{C}$,
data matrix $X \in \mathbb{R}^{D \times n}$,
$\beta > 0$
Output: mapping $g : \mathcal{X} \to \mathbb{R}^d$

$$\begin{aligned} \min_{W,\xi} \quad & \mathrm{tr}\,(W) + \frac{\beta}{|\mathcal{C}|} \sum_{\mathcal{C}} \xi_{ijk\ell} \\ d(x_i, x_j) &\doteq (X_i - X_j)^{\mathsf{T}} W (X_i - X_j) \\ d(x_i, x_j) + 1 &\leq d(x_k, x_\ell) + \xi_{ijk\ell} \\ \xi_{ijk\ell} &\geq 0 \qquad\qquad \forall (i,j,k,\ell) \in \mathcal{C} \\ W &\succeq 0 \end{aligned}$$

3.2 Non-linear Projection via Kernels

The formulation in Algorithm 2 can be generalized to support non-linear embeddings by the use of kernels, following the method of Globerson and Roweis (2007): we first map the data into a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ via a feature map ϕ with corresponding kernel function $k(x,y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$; then, the data is mapped to $\mathbb{R}^d$ by a linear projection $M : \mathcal{H} \to \mathbb{R}^d$. The embedding function $g : \mathcal{X} \to \mathbb{R}^d$ is the therefore the composition of the projection M with ϕ:

$$g(x) = M(\phi(x)).$$

Because ϕ may be non-linear, this allows us to learn a non-linear embedding g.

More precisely, we consider M as being comprised of d elements of $\mathcal{H}$, that is, $\{\omega_1, \omega_2, \ldots, \omega_d\} \subseteq \mathcal{H}$. The embedding g can thus be expressed as

$$g(x) = \left(\langle \omega_p, \phi(x) \rangle_{\mathcal{H}} \right)_{p=1}^{d},$$

where $(\cdot)_{p=1}^{d}$ denotes concatenation.

Note that in general, $\mathcal{H}$ may be infinite-dimensional, so directly optimizing M may not be feasible. However, by appropriately regularizing M, we may invoke the generalized representer theorem (Schölkopf et al., 2001). Our choice of regularization is the Hilbert-Schmidt norm of M, which, in this case, reduces to

$$\|M\|_{\mathrm{HS}}^2 = \sum_{p=1}^{d} \langle \omega_p, \omega_p \rangle_{\mathcal{H}}.$$

With this choice of regularization, it follows from the generalized representer theorem that at an optimum, each ω_p must lie in the span of the training data, that is,

$$\omega_p = \sum_{i=1}^{n} N_{pi} \phi(x_i), \quad p = 1, \ldots, d,$$

for some real-valued matrix $N \in \mathbb{R}^{d \times n}$. If Φ is a matrix representation of $\mathcal{X}$ in $\mathcal{H}$ (i.e., $\Phi_i = \phi(x_i)$ for $x_i \in \mathcal{X}$), then the projection operator M can be expressed as

$$M = N\Phi^{\mathsf{T}}. \tag{3}$$

We can now reformulate the embedding problem as an optimization over N rather than M. Using (3), the regularization term can be expressed as

$$\|M\|_{\mathrm{HS}}^2 = \mathrm{tr}(\Phi N^{\mathsf{T}} N \Phi^{\mathsf{T}}) = \mathrm{tr}(N^{\mathsf{T}} N \Phi^{\mathsf{T}} \Phi) = \mathrm{tr}(N^{\mathsf{T}} N K),$$

where K is the kernel matrix over $\mathcal{X}$:

$$K = \Phi^{\mathsf{T}} \Phi, \quad \text{with } K_{ij} = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}} = k(x_i, x_j).$$

To formulate the distance constraints in terms of N, we first express the embedding g in terms of N and the kernel function:

$$g(x) = M(\phi(x)) = N\Phi^{T}(\phi(x)) = N\left(\langle \Phi_i, \phi(x) \rangle_{\mathcal{H}} \right)_{i=1}^{n} = N\left(k(x_i, x) \right)_{i=1}^{n} = NK_x,$$

where K_x is the column vector formed by evaluating the kernel function k at x against the training set. The inner product matrix of embedded points can therefore be expressed as

$$A = KN^{\mathsf{T}}NK,$$

which allows to express the distance constraints in terms of N and the kernel matrix K:

$$(K_i - K_j)^{\mathsf{T}}N^{\mathsf{T}}N(K_i - K_j) + 1 \leq (K_k - K_\ell)^{\mathsf{T}}N^{\mathsf{T}}N(K_k - K_\ell).$$

The embedding problem thus amounts to solving the following optimization problem in N and ξ:

$$\begin{aligned} \min_{N,\xi \geq 0} \quad & \operatorname{tr}(N^{\mathsf{T}}NK) + \frac{\beta}{|\mathcal{C}|}\sum_{\mathcal{C}} \xi_{ijk\ell} \\ \text{s.t.} \quad & (K_i - K_j)^{\mathsf{T}}N^{\mathsf{T}}N(K_i - K_j) + 1 \leq (K_k - K_\ell)^{\mathsf{T}}N^{\mathsf{T}}N(K_k - K_\ell) + \xi_{ijk\ell}, \\ & \forall (i,j,k,\ell) \in \mathcal{C}. \end{aligned} \tag{4}$$

Again, the distance constraints in (4) are non-convex due to the differences of quadratic terms. And, as in the previous section, N only appears in the form of inner products $N^{\mathsf{T}}N$ in (4)—both in the constraints, and in the regularization term—so we can again derive a convex optimization problem by changing variables to $W \doteq N^{\mathsf{T}}N \succeq 0$. The resulting embedding problem is listed as Algorithm 3, again a semi-definite programming problem (SDP), with an objective function and constraints that are linear in W.

After solving for W, the matrix N can be recovered by computing the spectral decomposition $W = V\Lambda V^{\mathsf{T}}$, and defining $N = \Lambda^{1/2}V^{\mathsf{T}}$. The resulting embedding function takes the form:

$$g(x) = \Lambda^{1/2}V^{\mathsf{T}}K_x.$$

As in Schultz and Joachims (2004), this formulation can be interpreted as learning a Mahalanobis distance metric $\Phi W \Phi^{\mathsf{T}}$ over $\mathcal{H}$. More generally, we can view this as a form of kernel learning, where the kernel matrix A is restricted to the set

$$A \in \{KWK \;:\; W \succeq 0\}. \tag{5}$$

3.3 Connection to GNMDS

We conclude this section by drawing a connection between Algorithm 3 and the generalized non-metric MDS (GNMDS) algorithm of Agarwal et al. (2007).

First, we observe that the i-th column, K_i, of the kernel matrix K can be expressed in terms of K and the i^{th} standard basis vector e_i:

$$K_i = Ke_i.$$

From this, it follows that distance computations in Algorithm 3 can be equivalently expressed as

$$\begin{aligned} d(x_i,x_j) &= (K_i - K_j)^{\mathsf{T}}W(K_i - K_j) \\ &= (K(e_i - e_j))^{\mathsf{T}}W(K(e_i - e_j)) \\ &= (e_i - e_j)^{\mathsf{T}}K^{\mathsf{T}}WK(e_i - e_j). \end{aligned} \tag{6}$$

Algorithm 3 Kernel partial order embedding (KPOE)

Input: n objects $\mathcal{X}$,
partial order $\mathcal{C}$,
kernel matrix K,
$\beta > 0$
Output: mapping $g : \mathcal{X} \to \mathbb{R}^n$

$$\begin{aligned}
\min_{W,\xi} \quad & \operatorname{tr}(WK) + \frac{\beta}{|\mathcal{C}|} \sum_{\mathcal{C}} \xi_{ijk\ell} \\
d(x_i, x_j) &\doteq (K_i - K_j)^\mathsf{T} W (K_i - K_j) \\
d(x_i, x_j) + 1 &\leq d(x_k, x_\ell) + \xi_{ijk\ell} \\
\xi_{ijk\ell} &\geq 0 && \forall (i, j, k, \ell) \in \mathcal{C} \\
W &\succeq 0
\end{aligned}$$

If we consider the extremal case where $K = I$, that is, we have no prior feature-based knowledge of similarity between points, then Equation 6 simplifies to

$$d(x_i, x_j) = (e_i - e_j)^\mathsf{T} I W I (e_i - e_j) = W_{ii} + W_{jj} - W_{ij} - W_{ji}.$$

Therefore, in this setting, rather than defining a feature transformation, W directly encodes the inner products between embedded training points. Similarly, the regularization term becomes

$$\operatorname{tr}(WK) = \operatorname{tr}(WI) = \operatorname{tr}(W).$$

Minimizing the regularization term can be interpreted as minimizing a convex upper bound on the rank of W (Boyd and Vandenberghe, 2004), which expresses a preference for low-dimensional embeddings. Thus, by setting $K = I$ in Algorithm 3, we directly recover the GNMDS algorithm.

Note that directly learning inner products between embedded training data points rather than a feature transformation does not allow a meaningful out-of-sample extension, to embed unseen data points. On the other hand, by Equation 5, it is clear that the algorithm optimizes over the entire cone of PSD matrices. Thus, if $\mathcal{C}$ defines a DAG, we could exploit the fact that a partial order over distances always allows an embedding which satisfies all constraints in $\mathcal{C}$ (see Appendix A) to eliminate the slack variables from the program entirely.

4. Multiple Kernel Embedding

In the previous section, we derived an algorithm to learn an optimal projection from a kernel space $\mathcal{H}$ to $\mathbb{R}^d$ such that Euclidean distance between embedded points conforms to perceptual similarity. If, however, the data is heterogeneous in nature, it may not be realistic to assume that a single feature representation can sufficiently capture the inherent structure in the data. For example, if the objects in question are images, it may be natural to encode texture information by one set of features, and color in another, and it is not immediately clear how to reconcile these two disparate sources of information into a single kernel space.

However, by encoding each source of information independently by separate feature spaces $\mathcal{H}^1, \mathcal{H}^2, \ldots$—equivalently, kernel matrices $K^1, K^2, \ldots$—we can formulate a multiple kernel learning algorithm to optimally combine all feature spaces into a single, unified embedding space. In this section, we will derive a novel, projection-based approach to multiple-kernel learning and extend Algorithm 3 to support heterogeneous data in a principled way.

4.1 Unweighted Combination

Let $K^1, K^2, \ldots, K^m$ be a set of kernel matrices, each with a corresponding feature map ϕ^p and RKHS $\mathcal{H}^p$, for $p \in 1, \ldots, m$. One natural way to combine the kernels is to look at the product space, which is formed by concatenating the feature maps:

$$\phi(x_i) = (\phi^1(x_i), \phi^2(x_i), \ldots, \phi^m(x_i)) = (\phi^p(x_i))_{p=1}^m.$$

Inner products can be computed in this space by summing across each feature map:

$$\langle \phi(x_i), \phi(x_j) \rangle = \sum_{p=1}^m \left\langle \phi^p(x_i), \phi^p(x_j) \right\rangle_{\mathcal{H}^p}.$$

resulting in the *sum-kernel*—also known as the *average kernel* or *product space kernel*. The corresponding kernel matrix can be conveniently represented as the unweighted sum of the base kernel matrices:

$$\widehat{K} = \sum_{p=1}^m K^p. \tag{7}$$

Since $\widehat{K}$ is a valid kernel matrix itself, we could use $\widehat{K}$ as input for Algorithm 3. As a result, the algorithm would learn a kernel from the family

$$\begin{aligned}\mathcal{K}_1 &= \left\{ \left(\sum_{p=1}^m K^p \right) W \left(\sum_{p=1}^m K^p \right) \;:\; W \succeq 0 \right\} \\ &= \left\{ \sum_{p,q=1}^m K^p W K^q \;:\; W \succeq 0 \right\}.\end{aligned}$$

4.2 Weighted Combination

Note that $\mathcal{K}_1$ treats each kernel equally; it is therefore impossible to distinguish *good* features (i.e., those which can be transformed to best fit $\mathcal{C}$) from *bad* features, and as a result, the quality of the resulting embedding may be degraded. To combat this phenomenon, it is common to learn a scheme for weighting the kernels in a way which is optimal for a particular task. The most common approach to combining the base kernels is to take a positive-weighted sum

$$\sum_{p=1}^m \mu_p K^p \qquad\qquad (\mu_p \geq 0),$$

where the weights μ_p are learned in conjunction with a predictor (Lanckriet et al., 2004; Sonnenburg et al., 2006; Bach, 2008; Cortes et al., 2009). Equivalently, this can be viewed as learning a feature map

$$\phi(x_i) = \left(\sqrt{\mu_p} \phi^p(x_i) \right)_{p=1}^m,$$

where each base feature map has been scaled by the corresponding weight $\sqrt{\mu_p}$.

Applying this reasoning to learning an embedding that conforms to perceptual similarity, one might consider a two-stage approach to parameterizing the embedding (Figure 3(a)): first construct a weighted kernel combination, and then project from the combined kernel space. Lin et al. (2009) formulate a dimensionality reduction algorithm in this way. In the present setting, this would be achieved by simultaneously optimizing W and μ_p to choose an inner product matrix A from the set

$$\begin{aligned}\mathcal{K}_2 &= \left\{ \left(\sum_{p=1}^{m} \mu_p K^p \right) W \left(\sum_{p=1}^{m} \mu_p K^p \right) \;:\; W \succeq 0, \forall p,\, \mu_p \geq 0 \right\} \\ &= \left\{ \sum_{p,q=1}^{m} \mu_p K^p W \mu_q K^q \;:\; W \succeq 0, \forall p,\, \mu_p \geq 0 \right\}.\end{aligned}$$

The corresponding distance constraints, however, contain differences of terms cubic in the optimization variables W and μ_p:

$$\sum_{p,q} \left(K_i^p - K_j^p \right)^\mathsf{T} \mu_p W \mu_q \left(K_i^q - K_j^q \right) + 1 \leq \sum_{p,q} \left(K_k^p - K_\ell^p \right)^\mathsf{T} \mu_p W \mu_q \left(K_k^q - K_\ell^q \right),$$

and are therefore non-convex and difficult to optimize. Even simplifying the class by removing cross-terms, that is, restricting A to the form

$$\mathcal{K}_3 = \left\{ \sum_{p=1}^{m} \mu_p^2 K^p W K^p \;:\; W \succeq 0, \forall p,\, \mu_p \geq 0 \right\},$$

still leads to a non-convex problem, due to the difference of positive quadratic terms introduced by distance calculations:

$$\sum_{p=1}^{m} \left(K_i^p - K_j^p \right)^\mathsf{T} \mu_p^2 W \left(\mu_p K_i^p - K_j^p \right) + 1 \leq \sum_{p=1}^{m} \left(K_k^p - K_\ell^p \right)^\mathsf{T} \mu_p^2 W \left(\mu_p K_k^p - K_\ell^p \right).$$

However, a more subtle problem with this formulation lies in the assumption that a single weight can characterize the contribution of a kernel to the optimal embedding. In general, different kernels may be more or less informative on different subsets of $\mathcal{X}$ or different regions of the corresponding feature space. Constraining the embedding to a single metric W with a single weight μ_p for each kernel may be too restrictive to take advantage of this phenomenon.

4.3 Concatenated Projection

We now return to the original intuition behind Equation 7. The sum-kernel represents the inner product between points in the space formed by concatenating the base feature maps ϕ^p. The sets $\mathcal{K}_2$ and $\mathcal{K}_3$ characterize projections of the weighted combination space, and turn out to not be amenable to efficient optimization (Figure 3(a)). This can be seen as a consequence of prematurely combining kernels prior to projection.

Rather than projecting the (weighted) concatenation of $\phi^p(\cdot)$, we could alternatively concatenate learned projections $M^p(\phi^p(\cdot))$, as illustrated by Figure 3(b). Intuitively, by defining the embedding as the concatenation of m different projections, we allow the algorithm to learn an ensemble of

projections, each tailored to its corresponding domain space and jointly optimized to produce an optimal space. By contrast, the previously discussed formulations apply essentially the same projection to each (weighted) feature space, and are thus much less flexible than our proposed approach. Mathematically, an embedding function of this form can be expressed as the concatenation

$$g(x) = \left(M^p \left(\phi^p(x)\right)\right)_{p=1}^m.$$

Now, given this characterization of the embedding function, we can adapt Algorithm 3 to optimize over multiple kernels. As in the single-kernel case, we introduce regularization terms for each projection operator M^p

$$\sum_{p=1}^m \|M^p\|_{\mathrm{HS}}^2$$

to the objective function. Again, by invoking the representer theorem for each M^p, it follows that

$$M^p = N^p \left(\Phi^p\right)^{\mathsf{T}},$$

for some matrix N^p, which allows to reformulate the embedding problem as a joint optimization over $N^p,\ p = 1,\ldots,m$ rather than $M^p,\ p = 1,\ldots,m$. Indeed, the regularization terms can be expressed as

$$\sum_{p=1}^m \|M^p\|_{\mathrm{HS}}^2 = \sum_{p=1}^m \mathrm{tr}\left((N^p)^{\mathsf{T}}(N^p)K^p\right). \tag{8}$$

The embedding function can now be rewritten as

$$g(x) = \left(M^p \left(\phi^p(x)\right)\right)_{p=1}^m = \left(N^p K_x^p\right)_{p=1}^m, \tag{9}$$

and the inner products between embedded points take the form:

$$\begin{aligned} A_{ij} = \langle g(x_i), g(x_j)\rangle &= \sum_{p=1}^m \left(N^p K_i^p\right)^{\mathsf{T}} \left(N^p K_j^p\right) \\ &= \sum_{p=1}^m (K_i^p)^{\mathsf{T}}(N^p)^{\mathsf{T}}(N^p)(K_j^p). \end{aligned}$$

Similarly, squared Euclidean distance also decomposes by kernel:

$$\|g(x_i) - g(x_j)\|^2 = \sum_{p=1}^m \left(K_i^p - K_j^p\right)^{\mathsf{T}} (N^p)^{\mathsf{T}}(N^p) \left(K_i^p - K_j^p\right). \tag{10}$$

Finally, since the matrices $N^p,\ p = 1,\ldots,m$ only appear in the form of inner products in (8) and (10), we may instead optimize over PSD matrices $W^p = (N^p)^{\mathsf{T}}(N^p)$. This renders the regularization terms (8) and distances (10) linear in the optimization variables W^p. Extending Algorithm 3 to this parameterization of $g(\cdot)$ therefore results in an SDP, which is listed as Algorithm 4. To solve the SDP, we implemented a gradient descent solver, which is described in Appendix B.

The class of kernels over which Algorithm 4 optimizes can be expressed simply as the set

$$\mathcal{K}_4 = \left\{ \sum_{p=1}^m K^p W^p K^p \ :\ \forall p,\ W^p \succeq 0 \right\}$$

Kernel class	Learned kernel matrix
$\mathcal{K}_1 = \left\{\sum_{p,q} K^p W K^q\right\}$	$\left[K^1 + K^2 + \cdots + K^m\right][W]\left[K^1 + K^2 + \cdots + K^m\right]$
$\mathcal{K}_2 = \left\{\sum_{p,q} \mu_p \mu_q K^p W K^q\right\}$	$\begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^m \end{bmatrix}^\mathsf{T} \begin{bmatrix} \mu_1^2 W & \mu_1\mu_2 W & \cdots & \mu_1\mu_m W \\ \mu_2\mu_1 W & \mu_2^2 W & \cdots & \vdots \\ \vdots & & \ddots & \\ \mu_m\mu_1 W & & & \mu_m^2 W \end{bmatrix} \begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^m \end{bmatrix}$
$\mathcal{K}_3 = \left\{\sum_p \mu_p^2 K^p W K^p\right\}$	$\begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^m \end{bmatrix}^\mathsf{T} \begin{bmatrix} \mu_1^2 W & 0 & \cdots & 0 \\ 0 & \mu_2^2 W & \cdots & \vdots \\ \vdots & & \ddots & \\ 0 & & & \mu_m^2 W \end{bmatrix} \begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^m \end{bmatrix}$
$\mathcal{K}_4 = \left\{\sum_p K^p W^p K^p\right\}$	$\begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^m \end{bmatrix}^\mathsf{T} \begin{bmatrix} W^1 & 0 & \cdots & 0 \\ 0 & W^2 & \cdots & \vdots \\ \vdots & & \ddots & \\ 0 & & & W^m \end{bmatrix} \begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^m \end{bmatrix}$

Table 1: Block-matrix formulations of metric learning for multiple-kernel formulations ($\mathcal{K}_1$–$\mathcal{K}_4$). Each W^p is taken to be positive semi-definite. Note that all sets are equal when there is only one base kernel.

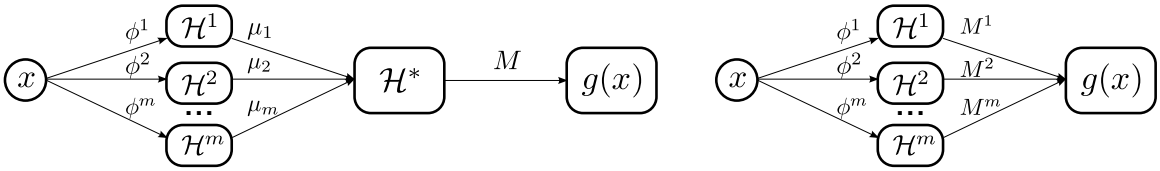
Note that $\mathcal{K}_4$ contains $\mathcal{K}_3$ as a special case when all W^p are positive scalar multiples of each-other. However, $\mathcal{K}_4$ leads to a convex optimization problem, where $\mathcal{K}_3$ does not.

Table 1 lists the block-matrix formulations of each of the kernel combination rules described in this section. It is worth noting that it is certainly valid to first form the unweighted combination kernel $\widehat{K}$ and then use $\mathcal{K}_1$ (Algorithm 3) to learn an optimal projection of the product space. However, as we will demonstrate in Section 5, our proposed multiple-kernel formulation ($\mathcal{K}_4$) outperforms the simple unweighted combination rule in practice.

4.4 Diagonal Learning

The MKPOE optimization is formulated as a semi-definite program over m different $n \times n$ matrices W^p—or, as shown in Table 1, a single $mn \times mn$ PSD matrix with a block-diagonal sparsity structure. Scaling this approach to large data sets can become problematic, as they require optimizing over multiple high-dimensional PSD matrices.

To cope with larger problems, the optimization problem can be refined to constrain each W^p to the set of diagonal matrices. If W^p are all diagonal, positive semi-definiteness is equivalent to non-negativity of the diagonal values (since they are also the eigenvalues of the matrix). This allows the constraints $W^p \succeq 0$ to be replaced by linear constraints $W^p_{ii} \geq 0$, and the resulting optimization problem is a linear program (LP), rather than an SDP. This modification reduces the flexibility of the model, but leads to a much more efficient optimization procedure.



(a) Weighted combination ($\mathcal{K}_2$) (b) Concatenated projection ($\mathcal{K}_4$)

Figure 3: Two variants of multiple-kernel embedding. (a) A data point $x \in \mathcal{X}$ is mapped into m feature spaces via $\phi^1, \phi^2, \ldots, \phi^m$, which are then scaled by $\mu_1, \mu_2, \ldots, \mu_m$ to form a weighted feature space $\mathcal{H}^*$, which is subsequently projected to the embedding space via M. (b) x is first mapped into each kernel's feature space and then its image in each space is directly projected into a Euclidean space via the corresponding projections M^p. The projections are jointly optimized to produce the embedding space.

Algorithm 4 Multiple kernel partial order embedding (MKPOE)

Input: n objects $\mathcal{X}$,
partial order $\mathcal{C}$,
m kernel matrices $K^1, K^2, \ldots, K^m$,
$\beta > 0$
Output: mapping $g : \mathcal{X} \to \mathbb{R}^{mn}$

$$\min_{W^p, \xi} \quad \sum_{p=1}^{m} \operatorname{tr}(W^p K^p) + \frac{\beta}{|\mathcal{C}|} \sum_{\mathcal{C}} \xi_{ijk\ell}$$

$$d(x_i, x_j) \doteq \sum_{p=1}^{m} \left(K_i^p - K_j^p\right)^{\mathsf{T}} W^p \left(K_i^p - K_j^p\right)$$

$$d(x_i, x_j) + 1 \leq d(x_k, x_\ell) + \xi_{ijk\ell}$$

$$\xi_{ijk\ell} \geq 0 \qquad \forall (i, j, k, \ell) \in \mathcal{C}$$

$$W^p \succeq 0 \qquad p = 1, 2, \ldots, m$$

More specifically, our implementation of Algorithm 4 operates by alternating sub-gradient descent on W^p and projection onto the feasible set $W^p \succeq 0$ (see Appendix B for details). For full matrices, this projection is accomplished by computing the spectral decomposition of each W^p, and thresholding the eigenvalues at 0. For diagonal matrices, this projection is accomplished simply by

$$W_{ii}^p \mapsto \max\left\{0, W_{ii}^p\right\},$$

which can be computed in $O(mn)$ time, compared to the $O(mn^3)$ time required to compute m spectral decompositions.

Restricting W^p to be diagonal not only simplifies the problem to linear programming, but carries the added interpretation of weighting the contribution of each (kernel, training point) pair in the construction of the embedding. A large value at W_{ii}^p corresponds to point i being a landmark for the

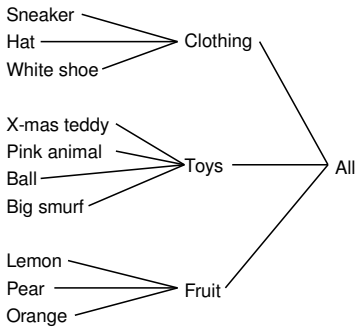


Figure 4: The label taxonomy for the experiment in Section 5.1.

features encoded in K^p. Note that each of the formulations listed in Table 1 has a corresponding diagonal variant, however, as in the full matrix case, only $\mathcal{K}_1$ and $\mathcal{K}_4$ lead to convex optimization problems.

5. Experiments

To evaluate our framework for learning multi-modal similarity, we first test the multiple kernel learning formulation on a simple toy taxonomy data set, and then on a real-world data set of musical perceptual similarity measurements.

5.1 Toy Experiment: Taxonomy Embedding

For our first experiment, we generated a toy data set from the Amsterdam Library of Object Images (ALOI) data set (Geusebroek et al., 2005). ALOI consists of RGB images of 1000 classes of objects against a black background. Each class corresponds to a single object, and examples are provided of the object under varying degrees of out-of-plane rotation.

In our experiment, we first selected 10 object classes, and from each class, sampled 20 examples. We then constructed an artificial taxonomy over the label set, as depicted in Figure 4. Using the taxonomy, we synthesized relative comparisons to span subtrees via their least common ancestor. For example,

$$(\mathit{Lemon}\#1, \mathit{Lemon}\#2, \mathit{Lemon}\#1, \mathit{Pear}\#1),$$
$$(\mathit{Lemon}\#1, \mathit{Pear}\#, 1, \mathit{Lemon}\#1, \mathit{Sneaker}\#1),$$

and so on. These comparisons are consistent and therefore can be represented as a directed acyclic graph. They are generated so as to avoid redundant, transitive edges in the graph.

For features, we generated five kernel matrices. The first is a simple linear kernel over the grayscale intensity values of the images, which, roughly speaking, compares objects by shape. The other four are Gaussian kernels over histograms in the (background-subtracted) red, green, blue, and intensity channels, and these kernels compare objects based on their color or intensity distributions.

We augment this set of kernels with five "noise" kernels, each of which was generated by sampling random points from the unit sphere in $\mathbb{R}^3$ and applying the linear kernel.

The data was partitioned into five 80/20 training and test set splits. To tune β, we further split the training set for 5-fold cross-validation, and swept over $\beta \in \{10^{-2}, 10^{-1}, \ldots, 10^{6}\}$. For

each fold, we learned a diagonally-constrained embedding with Algorithm 4, using the subset of relative comparisons (i, j, k, ℓ) with i, j, k and ℓ restricted to the training set. After learning the embedding, the held out data (validation or test) was mapped into the space, and the accuracy of the embedding was determined by counting the fraction of correctly predicted relative comparisons. In the validation and test sets, comparisons were processed to only include comparisons of the form (i, j, i, k) where i belongs to the validation (or test) set, and j and k belong to the training set.

We repeat this experiment for each base kernel individually (that is, optimizing over $\mathcal{K}_1$ with a single base kernel), as well as the unweighted sum kernel ($\mathcal{K}_1$ with all base kernels), and finally MKPOE ($\mathcal{K}_4$ with all base kernels). The results are averaged over all training/test splits, and collected in Figure 5. For comparison purposes, we include the prediction accuracy achieved by computing distances in each kernel's native space before learning. In each case, the optimized space indeed achieves higher accuracy than the corresponding native space. (Of course, the random noise kernels still predict randomly after optimization.)

As illustrated in Figure 5, taking the unweighted combination of kernels significantly degrades performance (relative to the best kernel) both in the native space (0.718 accuracy versus 0.862 for the linear kernel) and the optimized sum-kernel space (0.861 accuracy for the sum versus 0.951 for the linear kernel), that is, the unweighted sum kernel optimized by Algorithm 3. However, MKPOE ($\mathcal{K}_4$) correctly identifies and omits the random noise kernels by assigning them negligible weight, and achieves higher accuracy (0.984) than any of the single kernels (0.951 for the linear kernel, after learning).

5.2 Musical Artist Similarity

To test our framework on a real data set, we applied the MKPOE algorithm to the task of learning a similarity function between musical artists. The artist similarity problem is motivated by several real-world applications, including recommendation and playlist-generation for online radio. Because artists may be represented by a wide variety of different features (e.g., tags, acoustic features, social data), such applications can benefit greatly from an optimally integrated similarity metric.

The training data is derived from the *aset400* corpus of Ellis et al. (2002), which consists of 412 popular musicians, and 16385 relative comparisons of the form (i, j, i, k). Relative comparisons were acquired from human test subjects through a web survey; subjects were presented with a query artist (i), and asked to choose what they believe to be the most similar artist (j) from a list of 10 candidates. From each single response, 9 relative comparisons are synthesized, indicating that j is more similar to i than the remaining 9 artists (k) which were not chosen.

Our experiments here replicate and extend previous work on this data set (McFee and Lanckriet, 2009a). In the remainder of this section, we will first give an overview of the various types of features used to characterize each artist in Section 5.2.1. We will then discuss the experimental procedure in more detail in Section 5.2.2. The MKL embedding results are presented in Section 5.2.3, and are followed by an experiment detailing the efficacy of our constraint graph processing approach in Section 5.2.4.

5.2.1 FEATURES

We construct five base kernels over the data, incorporating acoustic, semantic, and social views of the artists.

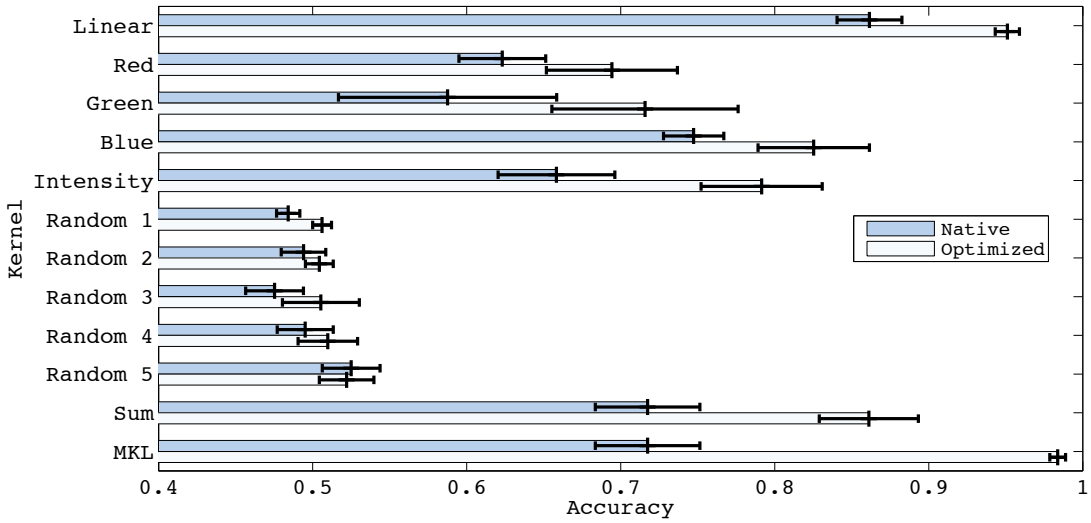


Figure 5: Mean test set accuracy for the experiment of Section 5.1. Error bars correspond to one standard deviation across folds. Accuracy is computed by counting the fraction of correctly predicted relative comparisons in the native space of each base kernel, and then in the optimized space produced by KPOE ($\mathcal{K}_1$ with a single base kernel). The unweighted combination of kernels (*Sum*) significantly degrades performance in both the native and optimized spaces. MKPOE (*MKL*, $\mathcal{K}_4$) correctly rejects the random kernels, and significantly outperforms the unweighted combination and the single best kernel.

- **MFCC**: for each artist, we collected between 1 and 10 songs (mean 4). For each song, we extracted a short clip consisting of 10000 half-overlapping 23ms windows. For each window, we computed the first 13 Mel Frequency Cepstral Coefficients (MFCCs) (Davis and Mermelstein, 1990), as well as their first and second instantaneous derivatives. This results in a sequence of 39-dimensional vectors (delta-MFCCs) for each song. Each artist i was then summarized by a Gaussian mixture model (GMM) p_i over delta-MFCCs extracted from the corresponding songs. Each GMM has 8 components and diagonal covariance matrices. Finally, the kernel between artists i and j is the probability product kernel (Jebara et al., 2004) between their corresponding delta-MFCC distributions p_i, p_j:

$$K_{ij}^{\text{mfcc}} = \int \sqrt{p_i(x)p_j(x)}\,dx.$$

- **Auto-tags (AT)**: Using the MFCC features described above, we applied the automatic tagging algorithm of Turnbull et al. (2008), which for each song yields a multinomial distribution over a set T of 149 musically-relevant tag words (*auto-tags*). Artist-level tag distributions q_i were formed by averaging model parameters (i.e., tag probabilities) across all of the songs of artist i. The kernel between artists i and j for auto-tags is a radial basis function applied to the

χ^2-distance between the multinomial distributions q_i and q_j:

$$K_{ij}^{\text{at}} = \exp\left(-\sigma \sum_{t \in T} \frac{(q_i(t) - q_j(t))^2}{q_i(t) + q_j(t)}\right).$$

In these experiments, we fixed $\sigma = 256$.

- **Social tags (ST)**: For each artist, we collected the top 100 most frequently used tag words from Last.fm,[2] a social music website which allows users to label songs or artists with arbitrary tag words or *social tags*. After stemming and stop-word removal, this results in a vocabulary of 7737 tag words. Each artist is then represented by a bag-of-words vector in $\mathbb{R}^{7737}$, and processed by TF-IDF. The kernel between artists for social tags is the cosine similarity (linear kernel) between TF-IDF vectors.

- **Biography (Bio)**: Last.fm also provides textual descriptions of artists in the form of user-contributed biographies. We collected biographies for each artist in the *aset400* data set, and after stemming and stop-word removal, we arrived at a vocabulary of 16753 biography words. As with social tags, the kernel between artists is the cosine similarity between TF-IDF bag-of-words vectors.

- **Collaborative filtering (CF)**: Celma (2008) collected collaborative filtering data from Last.fm in the form of a bipartite graph over users and artists, where each user is associated with the artists in her listening history. We filtered this data down to include only the aset400 artists, of which all but 5 were found in the collaborative filtering graph. The resulting graph has 336527 users and 407 artists, and is equivalently represented by a binary matrix where each row i corresponds to an artist, and each column j corresponds to a user. The ij entry of this matrix is 1 if we observe a user-artist association, and 0 otherwise. The kernel between artists in this view is the cosine of the angle between corresponding rows in the matrix, which can be interpreted as counting the amount of overlap between the sets of users listening to each artist and normalizing for overall artist popularity. For the 5 artists not found in the graph, we fill in the corresponding rows and columns of the kernel matrix with the identity matrix.

5.2.2 EXPERIMENTAL PROCEDURE

The data was randomly partitioned into ten 90/10 training/test splits. Given the inherent ambiguity in the task, and format of the survey, there is a great deal of conflicting information in the survey responses. To obtain a more accurate and internally consistent set of training comparisons, directly contradictory comparisons (e.g., (i, j, i, k) and (i, k, i, j)) were removed from both the training and test sets. Each training set was further cleaned by finding an acyclic subset of comparisons and taking its transitive reduction, resulting in a minimal partial order. (No further processing was performed on test comparisons.)

After training, test artists were mapped into the learned space (by Equation 9), and accuracy was measured by counting the number of measurements (i, j, i, k) correctly predicted by distance in the learned space, where i belongs to the test set, and j, k belong to the training set.

For each experiment, β is chosen from $\{10^{-2}, 10^{-1}, \ldots, 10^{7}\}$ by holding out 30% of the training constraints for validation. (Validation splits are generated from the unprocessed training set, and the

2. Last.fm can be found at `http://last.fm`.

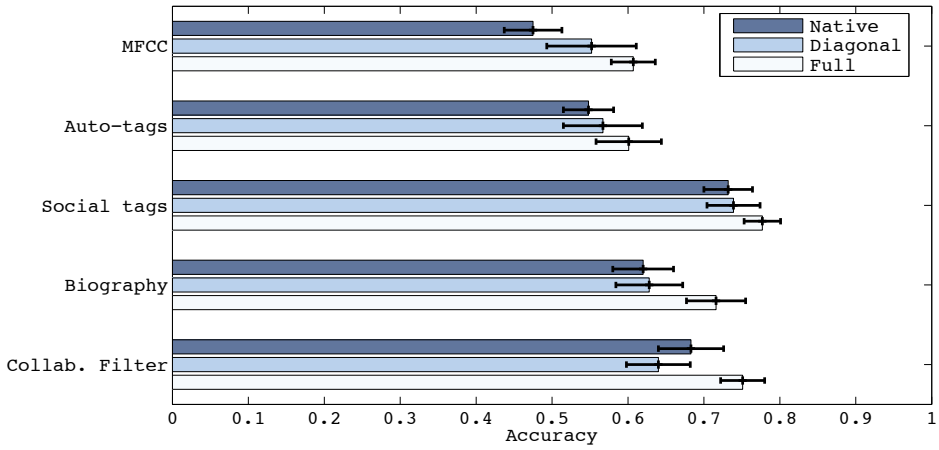


Figure 6: aset400 embedding results for each of the base kernels. Accuracy is computed in each kernel's native feature space, as well as the space produced by applying Algorithm 3 (i.e., optimizing over $\mathcal{K}_1$ with a single kernel) with either the diagonal or full-matrix formulation. Error bars correspond to one standard deviation across training/test splits.

remaining training constraints are processed as described above.) After finding the best-performing β, the embedding is trained on the full (processed) training set.

5.2.3 EMBEDDING RESULTS

For each base kernel, we evaluate the test-set performance in the native space (i.e., by distances calculated directly from the entries of the kernel matrix), and by learned metrics, both diagonal and full (optimizing over $\mathcal{K}_1$ with a single base kernel). Figure 6 illustrates the results.

We then repeated the experiment by examining different groupings of base kernels: acoustic (MFCC and Auto-tags), semantic (Social tags and Bio), social (Collaborative filter), and combinations of the groups. The different sets of kernels were combined by Algorithm 4 (optimizing over $\mathcal{K}_4$). The results are listed in Figure 7.

In all cases, MKPOE improves over the unweighted combination of base kernels. Moreover, many combinations outperform the single best kernel (ST, 0.777 ± 0.02 after optimization), and the algorithm is generally robust in the presence of poorly-performing kernels (MFCC and AT). Note that the poor performance of MFCC and AT kernels may be expected, as they derive from song-level rather than artist-level features, whereas ST provides high-level semantic descriptions which are generally more homogeneous across the songs of an artist, and Bio and CF are directly constructed at the artist level. For comparison purposes, we trained metrics on the sum kernel with $\mathcal{K}_1$ (Algorithm 3), resulting in accuracies of 0.676 ± 0.05 (diagonal) and 0.765 ± 0.03 (full). The proposed approach (Algorithm 4) applied to all kernels results in 0.754 ± 0.03 (diagonal), and 0.795 ± 0.02 (full).

Figure 8 illustrates the weights learned by Algorithm 4 using all five kernels and diagonally-constrained W^p matrices. Note that the learned metrics are both sparse (many 0 weights) and non-uniform across different kernels. In particular, the (lowest-performing) MFCC kernel is eliminated

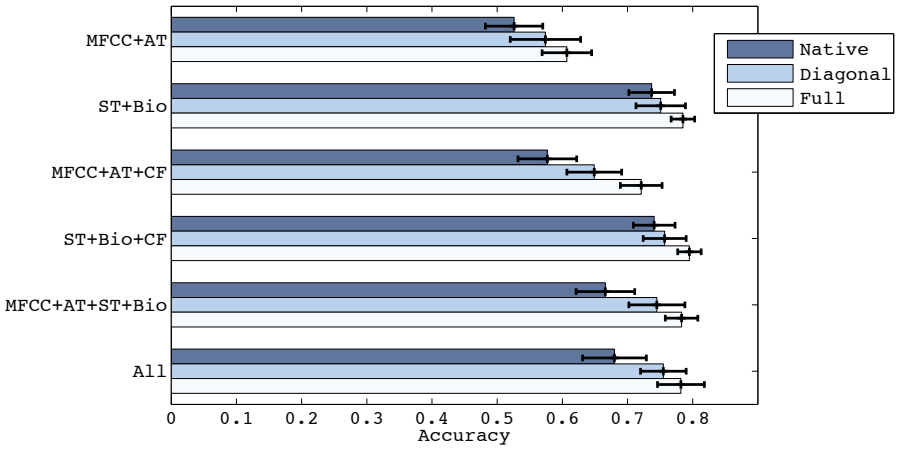


Figure 7: aset400 embedding results with multiple kernel learning: the learned metrics are optimized over $\mathcal{K}_4$ by Algorithm 4. *Native* corresponds to distances calculated according to the unweighted sum of base kernels.

by the algorithm, and the majority of the weight is assigned to the (highest-performing) social tag (ST) kernel.

A t-SNE (van der Maaten and Hinton, 2008) visualization of the space produced by MKPOE is illustrated in Figure 9. The embedding captures a great deal of high-level genre structure: for example, the *classic rock* and *metal* genres lie at the opposite end of the space from *pop* and *hip-hop*.

5.2.4 GRAPH PROCESSING RESULTS

To evaluate the effects of processing the constraint set for consistency and redundancy, we repeat the experiment of the previous section with different levels of processing applied to $\mathcal{C}$. Here, we focus on the Biography kernel, since it exhibits the largest gap in performance between the native and learned spaces.

As a baseline, we first consider the full set of similarity measurements as provided by human judgements, including all inconsistencies. To first deal with what appear to be the most egregious inconsistencies, we prune all directly inconsistent training measurements; that is, whenever (i, j, i, k) and (i, k, i, j) both appear, both are removed.[3] Finally, we consider the fully processed case by finding a maximal consistent subset (partial order) of $\mathcal{C}$ and removing all redundancies. Table 2 lists the number of training constraints retained by each step of processing (averaged over the random splits).

Using each of these variants of the training set, we test the embedding algorithm with both diagonal and full-matrix formulations. The results are presented in Table 2. Each level of graph processing results in a significant reduction in the number of training comparisons (and, therefore,

3. A more sophisticated approach could be used here, for example, majority voting, provided there is sufficient over-sampling of comparisons. The aset400 data lacks sufficient over-sampling for majority voting, so we default to this relatively simple approach.

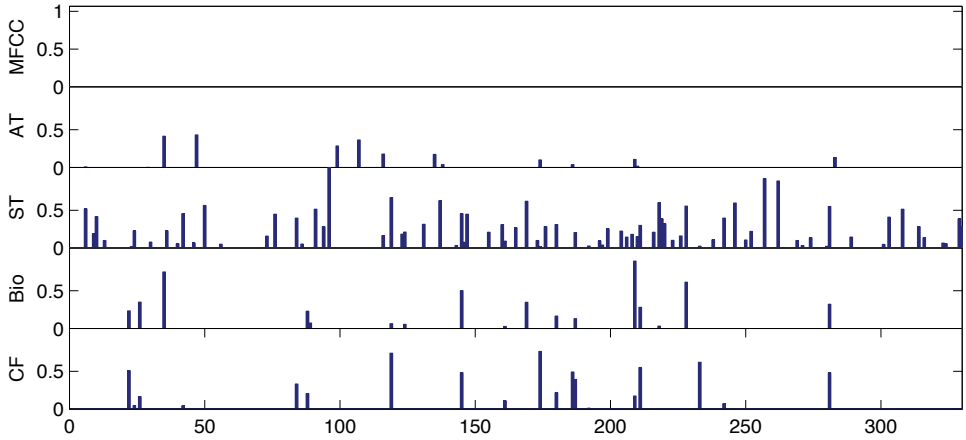


Figure 8: The weighting learned by Algorithm 4 using all five kernels and diagonal W^p. Each bar plot contains the diagonal of the corresponding kernel's learned metric. The horizontal axis corresponds to the index in the training set, and the vertical axis corresponds to the learned weight in each kernel space.

		Accuracy	
$\mathcal{C}$	$\lvert\mathcal{C}\rvert$ (Avg.)	Diagonal	Full
Full	8951.3	0.622±0.05	0.715±0.04
Length-2	6684.5	0.630±0.05	0.714±0.04
Processed	4814.5	0.628±0.05	0.716±0.04

Table 2: aset400 embedding results (Biography kernel) for three possible refinements of the constraint set. *Full* includes all similarity measurements, with no pruning for consistency or redundancy. *Length-2* removes all length-2 cycles (i.e., (i, j, k, ℓ) and (k, ℓ, i, j)). *Processed* finds an approximate maximal consistent subset, and removes redundant constraints.

computational overhead of Algorithm 3), while not degrading the quality of the resulting embedding.

Finally, to test the sensitivity of the algorithm to randomness in the acyclic subgraph routine, we repeated the above experiment ten times, each with a different random maximal acyclic constraint set and the full matrix formulation of the algorithm. As depicted in Figure 10, the randomness in the constraint generation has little impact on the accuracy of the learned metric: the largest standard deviation is 0.007 (split #7).

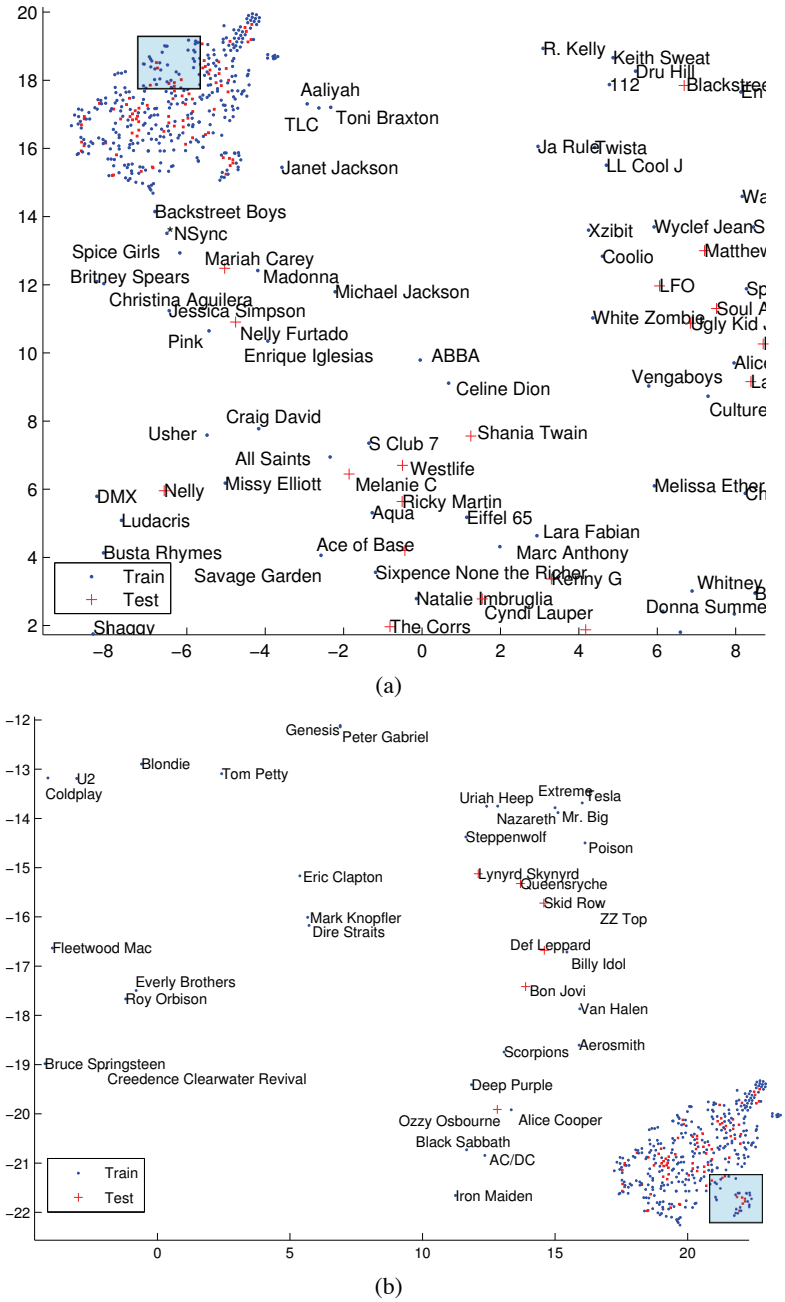


Figure 9: t-SNE visualizations of an embedding of aset400 produced by MKPOE. The embedding is constructed by optimizing over $\mathcal{K}_4$ with all five base kernels. The two clusters shown roughly correspond to (a) pop/hip-hop, and (b) classic rock/metal genres. Out-of-sample points are indicated by a red +.

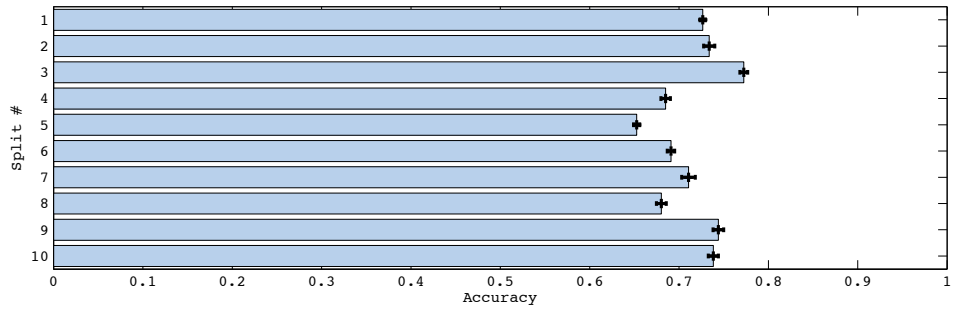


Figure 10: Accuracy of the learned embedding for each training/test split, averaged over ten trials with random maximal acyclic constraint subgraphs. Error bars correspond to one standard deviation.

6. Hardness of Dimensionality Reduction

The algorithms given in Sections 3 and 4 attempt to produce low-dimensional solutions by regularizing W, which can be seen as a convex approximation to the rank of the embedding. In general, because rank constraints are not convex, convex optimization techniques cannot efficiently minimize dimensionality. This does not necessarily imply other techniques could not work. So, it is natural to ask if exact solutions of minimal dimensionality can be found efficiently, particularly in the multidimensional scaling scenario, that is, when $K = I$ (Section 3.3).

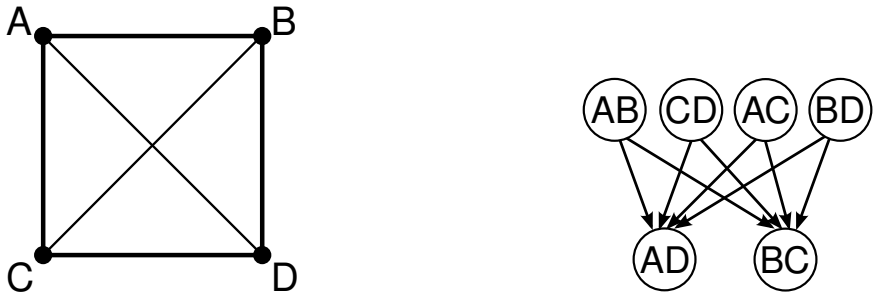
As a special case, one may wonder if any instance $(\mathcal{X}, \mathcal{C})$ can be satisfied in $\mathbb{R}^1$. As Figure 11 demonstrates, not all instances can be realized in one dimension. Even more, we show that it is NP-Complete to decide if a given $\mathcal{C}$ can be satisfied in $\mathbb{R}^1$. Given an embedding, it can be verified in polynomial time whether $\mathcal{C}$ is satisfied or not by simply computing the distances between all pairs and checking each comparison in $\mathcal{C}$, so the decision problem is in NP. It remains to show that the $\mathbb{R}^1$ partial order embedding problem (hereafter referred to as *1-POE*) is NP-Hard. We reduce from the *Betweenness* problem (Opatrny, 1979), which is known to be NP-complete.

Definition 1 (Betweenness) *Given a finite set Z and a collection T of ordered triples (a,b,c) of distinct elements from Z, is there a one-to-one function $f : Z \to \mathbb{R}$ such that for each $(a,b,c) \in T$, either $f(a) < f(b) < f(c)$ or $f(c) < f(b) < f(a)$?*

Theorem 1 *1-POE is NP-Hard.*

Proof Let (Z,T) be an instance of Betweenness. Let $\mathcal{X} = Z$, and for each $(a,b,c) \in T$, introduce constraints (a,b,a,c) and (b,c,a,c) to $\mathcal{C}$. Since Euclidean distance in $\mathbb{R}^1$ is simply line distance, these constraints force $g(b)$ to lie between $g(a)$ and $g(c)$. Therefore, the original instance $(Z,T) \in$ Betweenness if and only if the new instance $(\mathcal{X}, \mathcal{C}) \in$ 1-POE. Since Betweenness is NP-Hard, 1-POE is NP-Hard as well. ■

Since 1-POE can be reduced to the general optimization problem of finding an embedding of minimal dimensionality, we can conclude that dimensionality reduction subject to partial order constraints is also NP-Hard.



(a) (b)

Figure 11: (a) The vertices of a square in $\mathbb{R}^2$. (b) The partial order over distances induced by the square: each side is less than each diagonal. This constraint set cannot be satisfied in $\mathbb{R}^1$.

7. Conclusion

We have demonstrated a novel method for optimally integrating heterogeneous data to conform to measurements of perceptual similarity. By interpreting a collection of relative similarity comparisons as a directed graph over pairs, we are able to apply graph-theoretic techniques to isolate and prune inconsistencies in the training set and reduce computational overhead by eliminating redundant constraints in the optimization procedure.

Our multiple-kernel formulation offers a principled way to integrate multiple feature modalities into a unified similarity space. Our formulation carries the intuitive geometric interpretation of concatenated projections, and results in a semidefinite program. By incorporating diagonal constraints as well, we are able to reduce the computational complexity of the algorithm, and learn a model which is both flexible—only using kernels in the portions of the space where they are informative—and interpretable—each diagonal weight corresponds to the contribution to the optimized space due to a single point within a single feature space. Table 1 provides a unified perspective of multiple kernel learning formulations for embedding problems, but it is clearly not complete. It will be the subject of future work to explore and compare alternative generalizations and restrictions of the formulations presented here.

Acknowledgments

The authors acknowledge support from NSF Grant DMS-MSPA 0625409 and eHarmony, Inc.

Appendix A. Embedding Partial Orders

In this appendix, we prove that any set $\mathcal{X}$ with a partial order over distances $\mathcal{C}$ can be embedded into $\mathbb{R}^n$ while satisfying all distance comparisons.

In the special case where $\mathcal{C}$ is a total ordering over all pairs (i.e., a chain graph), the problem reduces to non-metric multidimensional scaling (Kruskal, 1964), and a constraint-satisfying embedding can always be found by the constant-shift embedding algorithm of Roth et al. (2003). In

Algorithm 5 Naïve total order construction

Input: objects $\mathcal{X}$, partial order $\mathcal{C}$
Output: symmetric dissimilarity matrix $\Delta \in \mathbb{R}^{n \times n}$

for each i in $1 \ldots n$ **do**
  $\Delta_{ii} \leftarrow 0$
end for
for each (k, ℓ) in topological order **do**
  if in-degree$(k, \ell) = 0$ **then**
    $\Delta_{k\ell}, \Delta_{\ell k} \leftarrow 1$
  else
    $\Delta_{k\ell}, \Delta_{\ell k} \leftarrow \max_{(i,j,k,\ell) \in \mathcal{C}} \Delta_{ij} + 1$
  end if
end for

general, $\mathcal{C}$ is not a total order, but a $\mathcal{C}$-respecting embedding can always be produced by reducing the partial order to a (weak) total order by topologically sorting the graph (see Algorithm 5).

Let Δ be the dissimilarity matrix produced by Algorithm 5 on an instance $(\mathcal{X}, \mathcal{C})$. An embedding can be found by first applying classical multidimensional scaling (MDS) (Cox and Cox, 1994) to Δ:

$$A = -\frac{1}{2} H \Delta H,$$

where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ is the $n \times n$ centering matrix, and $\mathbf{1}$ is a vector of 1s. Shifting the spectrum of A yields

$$A - \lambda_n(A) I = \widehat{A} \succeq 0,$$

where $\lambda_n(A)$ is the minimum eigenvalue of A. The embedding g can be found by decomposing $\widehat{A} = V \widehat{\Lambda} V^\mathsf{T}$, so that $g(x_i)$ is the i^{th} column of $\widehat{\Lambda}^{1/2} V^\mathsf{T}$; this is the solution constructed by the constant-shift embedding non-metric MDS algorithm of Roth et al. (2003).

Applying this transformation to A affects distances by

$$\begin{aligned} \|g(x_i) - g(x_j)\|^2 = \widehat{A}_{ii} + \widehat{A}_{jj} - 2\widehat{A}_{ij} &= (A_{ii} - \lambda_n) + (A_{jj} - \lambda_n) - 2A_{ij} \\ &= A_{ii} + A_{jj} - 2A_{ij} - 2\lambda_n. \end{aligned}$$

Since adding a constant $(-2\lambda_n)$ preserves the ordering of distances, the total order (and hence $\mathcal{C}$) is preserved by this transformation. Thus, for any instance $(\mathcal{X}, \mathcal{C})$, an embedding can be found in $\mathbb{R}^{n-1}$.

Appendix B. Solver

Our implementation of Algorithm 4 is based on a simple projected (sub)gradient descent. To simplify exposition, we show the derivation of the single-kernel SDP version of the algorithm (Algorithm 3) with unit margins. (It is straightforward to extend the derivation to the multiple-kernel and LP settings.)

We first observe that a kernel matrix column K_i can be expressed as $K^\mathsf{T} e_i$ where e_i is the i^{th} standard basis vector. We can then denote the distance calculations in terms of Frobenius inner products:

$$\begin{aligned} d(x_i,x_j) &= (K_i - K_j)^\mathsf{T} W (K_i - K_j) \\ &= (e_i - e_j)^\mathsf{T} KWK(e_i - e_j) \\ &= \operatorname{tr}(KWK(e_i - e_j)(e_i - e_j)^\mathsf{T}) = \operatorname{tr}(WKE_{ij}K) \\ &= \left\langle W, KE_{ij}K \right\rangle_F, \end{aligned}$$

where $E_{ij} = (e_i - e_j)(e_i - e_j)^\mathsf{T}$.

A margin constraint (i,j,k,ℓ) can now be expressed as:

$$\begin{aligned} & d(x_i,x_j) + 1 \leq d(x_k,x_\ell) + \xi_{ijk\ell} \\ \Rightarrow \quad & \left\langle W, KE_{ij}K \right\rangle_F + 1 \leq \langle W, KE_{k\ell}K \rangle_F + \xi_{ijk\ell} \\ \Rightarrow \quad & \xi_{ijk\ell} \geq 1 + \left\langle W, K(E_{ij} - E_{k\ell})K \right\rangle_F. \end{aligned}$$

The slack variables $\xi_{ijk\ell}$ can be eliminated from the program by rewriting the objective in terms of the constraints:

$$\min_{W \succeq 0} f(W) \text{ where } f(W) = \operatorname{tr}(WK) + \frac{\beta}{|\mathcal{C}|} \sum_{\mathcal{C}} h\left(1 + \left\langle W, K(E_{ij} - E_{k\ell})K \right\rangle_F\right),$$

where

$$h(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$$

is the hinge loss.

The gradient ∇f has two components: one due to regularization, and one due to the hinge loss. The gradient due to regularization is simply K. The loss term decomposes linearly, and for each $(i,j,k,\ell) \in \mathcal{C}$, a sub-gradient direction can be defined:

$$\frac{\partial}{\partial W} h\left(1 + d(x_i,x_j) - d(x_k,x_\ell)\right) = \begin{cases} 0 & d(x_i,x_j) + 1 \leq d(x_k,x_\ell) \\ K(E_{ij} - E_{k\ell})K & \text{otherwise.} \end{cases} \tag{11}$$

Rather than computing each gradient direction independently, we observe that each violated constraint contributes a matrix of the form $K(E_{ij} - E_{k\ell})K$. By linearity, we can collect all $(E_{ij} - E_{k\ell})$ terms and then pre- and post-multiply by K to obtain a more efficient calculation of ∇f:

$$\frac{\partial}{\partial W} f = K + K \left(\sum_{(i,j,k,\ell) \in \overline{\mathcal{C}}} E_{ij} - E_{k\ell} \right) K,$$

where $\overline{\mathcal{C}}$ is the set of all currently violated constraints.

After each gradient step $W \mapsto W - \alpha \nabla f$, the updated W is projected back onto the set of positive semidefinite matrices by computing its spectral decomposition and thresholding the eigenvalues by $\lambda_i \mapsto \max(0, \lambda_i)$.

To extend this derivation to the multiple-kernel case (Algorithm 4), we can define

$$d(x_i, x_j) \doteq \sum_{p=1}^{m} d^p(x_i, x_j),$$

and exploit linearity to compute each partial derivative $\partial / \partial W^p$ independently.

For the diagonally-constrained case, it suffices to substitute

$$K(E_{ij} - E_{k\ell})K \quad \mapsto \quad \mathrm{diag}(K(E_{ij} - E_{k\ell})K)$$

in Equation 11. After each gradient step in the diagonal case, the PSD constraint on W can be enforced by the projection $W_{ii} \mapsto \max(0, W_{ii})$.

Appendix C. Relationship to AUC

In this appendix, we formalize the connection between partial orders over distances and query-by-example ranking. Recall that Algorithm 2 minimizes the loss $1/|\mathcal{C}| \sum_{\mathcal{C}} \xi_{ijk\ell}$, where each $\xi_{ijk\ell} \geq 0$ is a slack variable associated with a margin constraint

$$d(i, j) + 1 \leq d(k, \ell) + \xi_{ijk\ell}.$$

As noted by Schultz and Joachims (2004), the fraction of relative comparisons satisfied by an embedding g is closely related to the area under the receiver operating characteristic curve (AUC). To make this connection precise, consider the following information retrieval problem. For each point $x_i \in \mathcal{X}$, we are given a partition of $\mathcal{X} \setminus \{i\}$:

$$\mathcal{X}_i^+ = \{x_j \ : \ x_j \in \mathcal{X} \text{ relevant for } x_i\}, \text{ and}$$
$$\mathcal{X}_i^- = \{x_k \ : \ x_k \in \mathcal{X} \text{ irrelevant for } x_i\}.$$

If we embed each $x_i \in \mathcal{X}$ into a Euclidean space, we can then rank the rest of the data $\mathcal{X} \setminus \{x_i\}$ by increasing distance from x_i. Truncating this ranked list at the top τ elements (i.e., closest τ points to x_i) will return a certain fraction of relevant points (true positives), and irrelevant points (false positives). Averaging over all values of τ defines the familiar AUC score, which can be compactly expressed as:

$$\mathrm{AUC}(x_i | g) = \frac{1}{|\mathcal{X}_i^+| \cdot |\mathcal{X}_i^-|} \sum_{(x_j, x_k) \in \mathcal{X}_i^+ \times \mathcal{X}_i^-} \mathbb{1}\left[\|g(x_i) - g(x_j)\| < \|g(x_i) - g(x_k)\|\right].$$

Intuitively, AUC can be interpreted as an average over all pairs $(x_j, x_k) \in \mathcal{X}_i^+ \times \mathcal{X}_i^-$ of the number of times x_i was mapped closer to a relevant point x_j than an irrelevant point x_k. This in turn can be conveniently expressed by a set of relative comparisons for each $x_i \in \mathcal{X}$:

$$\forall (x_j, x_k) \in \mathcal{X}_i^+ \times \mathcal{X}_i^- \ : \ (i, j, i, k).$$

An embedding which satisfies a complete set of constraints of this form will receive an AUC score of 1, since every relevant point must be closer to x_i than every irrelevant point.

Now, returning to the more general setting, we do not assume binary relevance scores or complete observations of relevance for all pairs of points. However, we can define the generalized

AUC score (GAUC) as simply the average number of correctly ordered pairs (equivalently, satisfied constraints) given a set of relative comparisons:

$$\mathrm{GAUC}(g) = \frac{1}{|\mathcal{C}|} \sum_{(i,j,k,\ell)\in\mathcal{C}} \mathbb{1}\left[\|g(x_i) - g(x_j)\| < \|g(x_k) - g(x_\ell)\|\right].$$

Like AUC, GAUC is bounded between 0 and 1, and the two scores coincide exactly in the previously described ranking problem. A corresponding loss function can be defined by reversing the order of the inequality, that is,

$$L_{\mathrm{GAUC}}(g) = \frac{1}{|\mathcal{C}|} \sum_{(i,j,k,\ell)\in\mathcal{C}} \mathbb{1}\left[\|g(x_i) - g(x_j)\| \geq \|g(x_k) - g(x_\ell)\|\right].$$

Note that L_{GAUC} takes the form of a sum over indicators, and can be interpreted as the average 0/1-loss over $\mathcal{C}$. This function is clearly not convex in g, and is therefore difficult to optimize. Algorithms 2, 3 and 4 instead optimize a convex upper bound on L_{GAUC} by replacing indicators by the hinge loss.

As in SVM, this is accomplished by introducing a unit margin and slack variable $\xi_{ijk\ell}$ for each $(i,j,k,\ell) \in \mathcal{C}$, and minimizing $1/|\mathcal{C}| \sum_{\mathcal{C}} \xi_{ijk\ell}$.

References

Sameer Agarwal, Joshua Wills, Lawrence Cayton, Gert Lanckriet, David Kriegman, and Serge Belongie. Generalized non-metric multi-dimensional scaling. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, 2007.

Alfred V. Aho, Michael R. Garey, and Jeffrey D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972. doi: 10.1137/0201008. URL `http://link.aip.org/link/?SMJ/1/131/1`.

Anelia Angelova. Data pruning. Master's thesis, California Institute of Technology, 2004.

Anelia Angelova, Yaser Abu-Mostafa, and Pietro Perona. Pruning training sets for learning of object categories. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1: 494–501, 2005. ISSN 1063-6919. doi: http://doi.ieeecomputersociety.org/10.1109/CVPR.2005.283.

Dana Angluin and Philip Laird. Learning from noisy examples. *Mach. Learn.*, 2(4):343–370, 1988. ISSN 0885-6125. doi: http://dx.doi.org/10.1023/A:1022873112823.

Francis R. Bach. Consistency of the group lasso and multiple kernel learning. *J. Mach. Learn. Res.*, 9:1179–1225, 2008. ISSN 1532-4435.

Yoshua Bengio, Jean-François Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

Bonnie Berger and Peter W. Shor. Approximation alogorithms for the maximum acyclic subgraph problem. In *SODA '90: Proceedings of the First Annual ACM-SIAM Symposium on Discrete algorithms*, pages 236–243, Philadelphia, PA, USA, 1990. Society for Industrial and Applied Mathematics. ISBN 0-89871-251-3.

Mikhail Bilenko, Sugato Basu, and Raymond J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the Twenty-first International Conference on Machine Learning*, pages 81–88, 2004.

Ingwer Borg and Patrick J.F. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer-Verlag, second edition, 2005.

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

Oscar Celma. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Learning non-linear combinations of kernels. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 396–404. 2009.

Trevor F. Cox and Michael A.A. Cox. *Multidimensional Scaling*. Chapman and Hall, 1994.

Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pages 209–216, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: http://doi.acm.org/10.1145/1273496.1273523.

Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. pages 65–74, 1990.

Daniel P.W. Ellis, Brian Whitman, Adam Berenzweig, and Steve Lawrence. The quest for ground truth in musical artist similarity. In *Proeedings of the International Symposium on Music Information Retrieval (ISMIR2002)*, pages 170–177, October 2002.

Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979. ISBN 0716710447.

Jan-Mark Geusebroek, Gertjan J. Burghouts, and Arnold W. M. Smeulders. The Amsterdam library of object images. *Int. J. Comput. Vis.*, 61(1):103–112, 2005.

Amir Globerson and Sam Roweis. Metric learning by collapsing classes. In Yair Weiss, Bernhard Schölkopf, and John Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 451–458, Cambridge, MA, 2006. MIT Press.

Amir Globerson and Sam Roweis. Visualizing pairwise similarity via semidefinite embedding. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, 2007.

Jacob Goldberger, Sam Roweis, Geoffrey Hinton, and Ruslan Salakhutdinov. Neighborhood components analysis. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520, Cambridge, MA, 2005. MIT Press.

Saketha Nath Jagarlapudi, Dinesh G, Raman S, Chiranjib Bhattacharyya, Aharon Ben-Tal, and Ramakrishnan K.R. On the algorithmics and applications of a mixed-norm based kernel learning formulation. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 844–852. 2009.

Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004. ISSN 1533-7928.

Maurice Kendall and Jean Dickinson Gibbons. *Rank correlation methods*. Oxford University Press, 1990.

Marius Kloft, Ulf Brefeld, Soeren Sonnenburg, Pavel Laskov, Klaus-Robert Müller, and Alexander 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. 2009.

Joseph B. Kruskal. Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, 29 (2), June 1964.

Gert R. G. Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I. Jordan. Learning the kernel matrix with semidefinite programming. *J. Mach. Learn. Res.*, 5:27–72, 2004. ISSN 1533-7928.

Yen-Yu Lin, Tyng-Luh Liu, and Chiou-Shann Fuh. Dimensionality reduction for data in multiple feature representations. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 961–968. 2009.

Brian McFee and Gert R. G. Lanckriet. Heterogeneous embedding for subjective artist similarity. In *Tenth International Symposium for Music Information Retrieval (ISMIR2009)*, October 2009a.

Brian McFee and Gert R. G. Lanckriet. Partial order embedding with multiple kernels. In *Proceedings of the 26th International Conference on Machine Learning (ICML'09)*, pages 721–728, June 2009b.

Jaroslav Opatrny. Total ordering problem. *SIAM J. Computing*, (8):111–114, 1979.

Volker Roth, Julian Laub, Joachim M. Buhmann, and Klaus-Robert Müller. Going metric: denoising pairwise data. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 809–816, Cambridge, MA, 2003. MIT Press.

Bernhard Schölkopf, Ralf Herbrich, Alex J. Smola, and Robert Williamson. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory*, pages 416–426, 2001.

Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

Noam Shental, Tomer Hertz, Daphna Weinshall, and Misha Pavel. Adjustment learning and relevant components analysis. In *European Conference on Computer Vision*, 2002.

Sören Sonnenburg, Gunnar Rätsch, Christin Schäfer, and Bernhard Schölkopf. Large scale multiple kernel learning. *J. Mach. Learn. Res.*, 7:1531–1565, 2006. ISSN 1532-4435.

Warren S. Torgerson. Multidimensional scaling: 1. theory and method. *Psychometrika*, 17:401–419, 1952.

Douglas Turnbull, Luke Barrington, David Torres, and Gert Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):467–476, February 2008.

Laurens van der Maaten and Geoffrey Hinton. Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

Alexander Vezhnevets and Olga Barinova. Avoiding boosting overfitting by removing confusing samples. In *ECML 2007*, pages 430–441, 2007.

Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 577–584, 2001.

Kilian Q. Weinberger, John Blitzer, and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. In Yair Weiss, Bernhard Schölkopf, and John Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 451–458, Cambridge, MA, 2006. MIT Press.

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 505–512, Cambridge, MA, 2003. MIT Press.

Alexander Zien and Cheng Soon Ong. Multiclass multiple kernel learning. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pages 1191–1198, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: http://doi.acm.org/10.1145/1273496.1273646.

Journal of Machine Learning Research 12 (2011) 525-564 Submitted 6/09; Revised 5/10; Published 2/11

Minimum Description Length Penalization for Group and Multi-Task Sparse Learning

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

Lyle H. Ungar UNGAR@CIS.UPENN.EDU
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA

Editor: Francis Bach

Abstract

We propose a framework MIC (Multiple Inclusion Criterion) for learning sparse models based on the information theoretic Minimum Description Length (MDL) principle. MIC provides an elegant way of incorporating arbitrary sparsity patterns in the feature space by using two-part MDL coding schemes. We present MIC based models for the problems of grouped feature selection (MIC-GROUP) and multi-task feature selection (MIC-MULTI). MIC-GROUP assumes that the features are divided into groups and induces two level sparsity, selecting a subset of the feature groups, and also selecting features within each selected group. MIC-MULTI applies when there are multiple related tasks that share the same set of potentially predictive features. It also induces two level sparsity, selecting a subset of the features, and then selecting which of the tasks each feature should be added to. Lastly, we propose a model, TRANSFEAT, that can be used to transfer knowledge from a set of previously learned tasks to a new task that is expected to share similar features. All three methods are designed for selecting a small set of predictive features from a large pool of candidate features. We demonstrate the effectiveness of our approach with experimental results on data from genomics and from word sense disambiguation problems.[1]

Keywords: feature selection, minimum description length principle, multi-task learning

1. Introduction

Classical supervised learning algorithms use a set of feature-label pairs to learn mappings from the features to the associated labels. They generally do this by considering each classification task (each possible label) in isolation and learning a model for that task. Learning models independently for different tasks often works well, but when the labeled data is limited and expensive to obtain, an attractive alternative is to build shared models for multiple related tasks (Caruana, 1997; Ando

1. Preliminary versions of this work appeared in Dhillon et al. (2008, 2009); Dhillon and Ungar (2009).

©2011 Paramveer S. Dhillon, Dean Foster and Lyle H. Ungar.

and Zhang, 2005). For example, when one is trying to predict a set of related responses ("tasks"), be they multiple clinical outcomes for patients or growth rates for yeast strains under different conditions, it may be possible to "borrow strength" by sharing information between the models for the different responses. Inductive transfer by building shared models can also be valuable when we have a disproportionate amount of labeled data for "similar" tasks. In such a case, building separate models for each task often gives poor predictive accuracies on tasks which have little data.

As a running example, we consider the problem of disambiguating word senses based on their context. Here, each observation of a word (e.g., a sentence containing the word "fire") is associated with multiple labels corresponding to each of the different possible meanings (e.g., for firing a person, firing a gun, firing off a note, etc.). Given the high-dimensional nature of Word Sense Disambiguation (WSD) data, feature selection is important for both linguistic understanding and for effective prediction (Chen et al., 2006). Also, since the features that are useful for predicting one sense are likely to be useful for predicting the other senses (perhaps with a coefficient of different sign.), we propose to select features that are useful in predicting these multiple responses.

Another closely related problem is grouped feature selection; that is, enforcing sparsity at the level of groups (feature classes) (Yuan and Lin, 2006; Bach et al., 2004; Dhillon et al., 2008). In this problem the group structure is over the features rather than over the tasks. Multi-task learning (described above) can also be thought of as a special case of this "group sparsity" scenario in which a group is defined by fixing a specific feature and ranging over multiple tasks. The block-norm approach to these problems uses a combination of ℓ_1 and ℓ_2 norms as regularization terms and adds each feature into the models of either none or all of the tasks (Obozinski et al., 2009) for the multi-task case and selects either none or all the features from a given group in the case of group sparsity. However, if the regularization constant is chosen using cross-validation, these approaches overestimate the support (Wainwright, 2009) as they select more features than the correct set of sparse features that generated the data. Wainwright (2009), Liu and Zhang (2008) and Nardi and Rinaldo (2008) have showed that certain scalings of the regularization coefficient yields more sparse solutions, which have with high probability the same support as the model generating the data. Even then there are further problems with these methods; in order to obtain very sparse solutions, one has to use a large regularization parameter that leads to suboptimal prediction accuracy because this high penalty not only shrinks irrelevant features to zero, but also shrinks relevant features to zero (Zhang, 2009a). Another alternative is to threshold the obtained coefficients (Lounici, 2008), but this introduces another thresholding parameter which needs to be tuned.

Motivated by the aforementioned reasons and by recent theoretical results on ℓ_0 penalty based regularization (Zhang, 2009a,b), we consider ℓ_0 penalty based formulations in this paper. In particular we propose to solve these two related problems, simultaneous feature selection for a set of multiple related tasks and grouped feature selection for a single task, by using coding schemes inspired by the Minimum Description Length (MDL) principle. We propose a common framework for these problems which we call the Multiple Inclusion Criterion (MIC). We use a "two part" version of MDL (Grünwald, 2005) to define a cost function which is greedily minimized by our methods. Since the greedy feature selection approximates the ℓ_0 penalty, we achieve a high degree of sparsity as is desired for both scientific interpretability and for accurate prediction in domains like Genomics and Word Sense Disambiguation (WSD) which have very high dimensional data. More importantly, our methods achieve *two-level sparsity*. In multi-task learning, each feature is added into models of a (possibly empty) subset of the tasks and in group feature selection, a (possibly empty) subset of the features are selected from each group (feature class).

We also propose a similarly motivated model (TRANSFEAT) (Dhillon and Ungar, 2009) for "intra-domain" adaptation which can be used to transfer knowledge from a set of already learned tasks to a new task which is similar to the aforementioned tasks. As an example, consider the task of predicting whether a word has a given sense when one already has models for predicting senses for synonyms of that word. These models are likely to share many of the same features; that is, a model for disambiguating one sense of "discharge" is likely to use many of the same features as one for disambiguating the sense of "fire" which is its synonym. Unlike MIC where we do simultaneous feature selection, the sharing in this case takes the form of a prior. TRANSFEAT is most beneficial when the word under consideration has considerably less labeled data available than the synonyms of that word (for example) so that building a supervised learning model for that word alone does not yield high predictive accuracy.

The rest of the paper is organized as follows. In the next section, we review relevant previous work. In Section 3, we provide background on feature selection and the MDL principle. Then in Section 4 we develop the general framework used by our models and describe the MIC-MULTI and MIC-GROUP models in detail. In Section 5, we show experimental results on real and synthetic data. In Section 6, we provide some model consistency results for the MIC models. In Section 7, we discuss the TRANSFEAT model and show its effectiveness for intra-domain adaptation on real world data sets. We conclude in Section 8 with a brief summary.

2. Related Work

The main contribution of this paper is to propose a joint framework for the related tasks of simultaneous feature selection for multiple related tasks and grouped feature selection for a single task. We are not aware of any previous work that addresses these two problems together, though Obozinski et al. (2009) do mention that these two problems are related. Nonetheless, there has been much previous work on each of these problems separately.

Jebara (2004) uses maximum-entropy discrimination to select a single subset of features across multiple SVM regression or classification problems that share a common set of potential features. Several other papers work within the framework of regularized regression, taking the penalty term to be an ℓ_1 norm over features of an ℓ_q norm over the coefficients for each feature (an "$\ell_1 - \ell_q$" penalty). Turlach et al. (2005) consider the case $q = \infty$, while Argyriou et al. (2008) and Obozinski et al. (2009) use $q = 2$. Argyriou et al. (2008) show that the general subspace selection problem can be formulated as an optimization problem involving the trace norm. Obozinski et al. (2009) propose BBLASSO, which focuses on the case where the trace norm is not required; they instead use a homotopy-based approach to evaluate the entire regularization path efficiently (Efron et al., 2004). Ando and Zhang (2005) also propose a framework which uses multiple prediction problems to learn an underlying shared structural parameter on the input (feature) space and they penalize the weight vectors by ℓ_2 norm. The idea behind $\ell_1 - \ell_q$ penalties is that when $q > 1$, the cost of making a coefficient nonzero is smaller for features that are shared across more tasks. Indeed, for either $q = 2$ or $q = \infty$, these algorithms tend in practice to yield nonzero coefficients for all of the tasks associated with features that get selected.

The related problem of grouped feature selection for a single task has also been addressed previously by Yuan and Lin (2006), Bach et al. (2004), Meier et al. (2008), Zhao et al. (2008) and is known as "Group Lasso". It is an extension of Lasso (ℓ_1 penalty) to the case of grouped structure in data and it enforces sparsity at the level of groups, that is, an entire group of features is selected. It

penalizes a (ℓ_1/ℓ_2) norm of the feature weights. An alternative formulation of Group Lasso is called Multiple Kernel Learning (MKL) (Bach et al., 2004; Bach, 2008); it penalizes the kernel Hilbert norm instead of the Euclidean norm.

Our approach is different from these methods in that we use ℓ_0 penalty-based greedy feature selection methods which minimize a cost function provided by MDL based coding schemes. MDL-based coding schemes provide much flexibility to incorporate arbitrary sparsity structures in the problem at hand. Recently, Huang et al. (2009) have also used coding schemes similar to the MDL for enforcing arbitrary structured sparsity patterns over the feature space.

3. Background

We assume a setting in which we are given n labeled data samples as $\{(x_i, y_i)_{i=1}^n \in \mathcal{X} \times \mathcal{Y}\}$ where $\mathcal{X} \in \mathbb{R}^p$ (the feature vector lives in a p dimensional space) and our goal is to find the parameter vector $(\mathbf{w} \in \mathbb{R}^p)$ of a statistical model fit to the above data. Alternatively, we can represent the data and the response variables in matrix form as $\mathbf{X}_{n\times p}$ and $\mathbf{Y}_{n\times 1}$, respectively, and the p dimensional weight vector as $\mathbf{w}_{p\times 1}$. Standard linear or logistic regression models of the form $\mathbf{Y} = \mathbf{w}\cdot\mathbf{X}$ (or $P(Y=1|X) = \frac{1}{1+e^{-\mathbf{w}\cdot\mathbf{X}}}$) fail to estimate the weight vector $\mathbf{w}$ in the case in which $p > n$ as they require inversion of a rank deficient matrix. To overcome this problem, *regularized* versions of the linear or logistic regressions are used which penalize some norm of the weight vectors:

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}}\{\|\mathbf{Y} - \mathbf{X}\cdot\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_q^q\},$$

where $\|\mathbf{w}\|_q$ represents the ℓ_q norm of $\mathbf{w}$ and λ is a hyperparameter.

For $q = 2$, the penalized regression is known as *Ridge Regression*, which corresponds to a Bayesian maximum a posteriori estimate for w under a Gaussian prior and shrinks the weight vector but does not enforce sparsity. The ℓ_1 penalty (Lasso) is equivalent to a double exponential prior on w (Tibshirani, 1996) and enforces sparsity by driving some of the weights to zero. As q approaches 0, $\|\mathbf{w}\|_q^q$ approaches the number of non-zero values in w. Hence regularization with ℓ_0 penalty is subset selection: Choosing a small number of the original features to retain in the model. Once a coefficient is in the model, all that counts is the cost of adding it in the first place. The ℓ_0 penalty has a number of advantages, including bounded worst case risk with respect to the ℓ_1 penalty and better control of False Discovery Rate (FDR) (Lin et al., 2008). There are other problems with the ℓ_1 penalty other than being less sparse as mentioned earlier, namely that its sparsity is not explicitly controlled, and in order to obtain very sparse solutions, one has to use a large regularization parameter that leads to suboptimal prediction accuracy because this high penalty not only shrinks irrelevant features to zero, but also shrinks relevant features to zero (Zhang, 2009a). However, one virtue of the ℓ_1 penalty is computational tractability (Efron et al., 2004), in contrast to the ℓ_0 penalty, which requires subset search which is (worst case) NP-Hard (Natarajan, 1995). In practice, approximate greedy algorithms like forward stepwise feature selection yield accurate, highly sparse solutions.

In a regression model, the residual sum of squares is proportional up to an additive constant to the negative log-likelihood of $\mathbf{Y}$ given $\mathbf{X}$ (Bickel and Doksum, 2001). Thus, the ℓ_0 regularization can be rephrased as a penalized likelihood criterion as follows:

$$score = -2\log P(\mathbf{Y}|\hat{w}_q) + F\cdot q,$$

where q is the number of features in the model, $P(\mathbf{Y}|\hat{w}_q)$ is the likelihood of the data given a model containing q features and F is a free parameter that controls the amount of weight placed on the ℓ_0 norm. Various penalties have been proposed for F, including

- $F = 2$, corresponding approximately to the AIC (Akaike Information Criterion) (Akaike, 1973),
- $F = \log n$, giving the BIC (Bayesian Information Criterion) (Schwartz, 1978),
- $F = 2\log p$, giving to RIC (Risk Inflation Criterion—similar to a "Bonferroni correction") (Foster and George, 1994).

As discussed in next subsection, each of these penalties can also be derived by using the Minimum Description Length (MDL) principle under different coding schemes.

3.1 Minimum Description Length (MDL) Principle Preliminaries

MDL (Rissanen, 1978, 1999) is a principle for model selection which treats the best model as the one which maximally compresses a digital representation of the observed data. We can envision a "Sender" who wants to transmit some data to a "Receiver" using as few bits as possible. For an illustrative example of the MDL principle, consider the case of simple linear regression. Assume that both the Sender and Receiver know the $n \times p$ data matrix $\mathbf{X}$, and the Sender wants to convey the values in the $n \times 1$ response matrix $\mathbf{Y}$. The naïve way to do this would be to send the raw values for each of the n observations of $\mathbf{Y}$. However, a more efficient way to send this information would be to describe a regression model $\hat{\mathbf{w}}$ for $\mathbf{Y}$ given $\mathbf{X}$ and then to send the residuals $\mathbf{Y} - \mathbf{X} \cdot \hat{\mathbf{w}}$, which have a much narrower distribution and would require fewer bits to encode.

To minimize description length, then, Sender should choose $\hat{\mathbf{w}}^*$ such that

$$\hat{\mathbf{w}}^* = \underset{\hat{\mathbf{w}}}{\operatorname{argmin}}\{\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}}) + \mathcal{D}(\hat{\mathbf{w}})\}, \tag{1}$$

where the first term is the description length of the residuals about the model, and the second term is the description length of the model itself. In other words, the first term represents the fit of the model to data; as the model fits better this term shrinks. The second term represents the complexity of the model; it grows as the model becomes more complex.

This version of the MDL principle is known as "*Two part* MDL" (Grünwald, 2005); the exact meaning of both these terms is described in the following sections.

In the next sections we show how to choose the two quantities in Equation 1 by incorporating the "structure" (characteristics) of the problem at hand. We then introduce search algorithms to approximately find a sparse set of weights $\hat{\mathbf{w}}$ that minimize Equation 1.

3.1.1 CODING THE DATA: $\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}})$

The Kraft inequality in information theory (Cover and Thomas, 2006) implies that for any probability distribution $\{p_i\}$ over a finite or countable set, there exists a corresponding code with codeword length $\lceil -\lg p_i \rceil$ (The logarithm is base 2). Moreover, these code lengths are optimal in the sense of minimizing the expected code length with respect to $\{p_i\}$. Also, if the Sender and Receiver agree on a model (e.g., linear regression), then they have a probability distribution over the residuals ε, so they will agree to use a code for the residuals with length:

$$\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}}) = -\lg P(\varepsilon|\hat{\mathbf{w}}) = -\lg P(\mathbf{Y}|\hat{\mathbf{w}}), \tag{2}$$

that is, the negative log-likelihood of the data given the model. We dropped the ceiling on $-\lg P(\mathbf{Y}|\hat{\mathbf{w}})$ since we use "idealized" code lengths (Barron et al., 1998).

Consider a forward stepwise-regression setting in which we have already added $q-1$ features to our model (including the intercept term), and we are deciding whether to include an extra q^{th} feature. Let Y_i denote the i^{th} row of $\mathbf{Y}$ and $\hat{w}_q$, a linear regression model with all q features, then,[2]:

$$\begin{aligned}
\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}}) &= -\lg \prod_{i=1}^{n} P(Y_i|\hat{w}_q) \\
&= -\sum_{i=1}^{n} \lg \left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(Y_i - X_i \cdot \hat{w}_q)^2 \right) \right] \\
&= \frac{1}{2\ln 2} \left[n \ln(2\pi\sigma^2) + \frac{(\mathbf{Y} - \mathbf{X}\cdot\hat{w}_q)^2}{\sigma^2} \right],
\end{aligned}$$

σ^2 is unknown in practice, but it can be estimated as:

$$\hat{\sigma}^2 = \frac{(\mathbf{Y} - \mathbf{X}\cdot\hat{w}_{q-1})^2}{n}.$$

Note that this is the ML (Maximum Likelihood) estimate for σ^2 which Sender uses, as ignoring the model-coding cost, maximizing likelihood is equivalent to minimizing description length. Some statisticians, in practice, use the unbiased estimate $\hat{\sigma}^2 = \frac{(\mathbf{Y}-\mathbf{X}\cdot\hat{w}_{q-1})^2}{n-q}$. In the experiments presented in this paper, we estimate $\hat{\sigma}^2$ without the current q^{th} feature in model, in order to prevent overfitting.

We can write the final expression for $\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}})$, incorporating $\hat{\sigma}^2$ as:

$$\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}}) = \frac{n}{2\ln 2} \left[\ln\left(\frac{2\pi \times (\mathbf{Y} - \mathbf{X}\cdot\hat{w}_{q-1})^2}{n} \right) + \left(\frac{\mathbf{Y} - \mathbf{X}\cdot\hat{w}_q}{\mathbf{Y} - \mathbf{X}\cdot\hat{w}_{q-1}} \right)^2 \right].$$

3.1.2 CODING THE MODEL: $\mathcal{D}(\hat{\mathbf{w}})$

Just as $\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}})$ depends on the model for the residuals that Sender and Receiver choose, so their coding scheme for $\hat{\mathbf{w}}$ itself will reflect their prior expectations.[3] When the number of features p is large (say, > 1000), Sender will likely only want to transmit a few of them that are most relevant, and hence the $\hat{\mathbf{w}}$ will contain mostly zeros. So, the first step in coding $\hat{\mathbf{w}}$ could be to say where the non-zero entries are located; if only a few features enter the model, this can be done relatively efficiently by listing the indices of the features in the set $\{1, 2, ..., p\}$. This requires $\lceil \lg p \rceil$ bits or approximately $\lg p$ bits.

The second step is to encode the numerical values of those coefficients. Rissanen (1983) suggested the basic approach for doing this by creating a discrete grid over some possible parameter

2. Note that in the following notation the square of vector, that is, $\mathbf{Z^2}$ means $\mathbf{Z^T Z}$.

3. By the Kraft inequality, we can interpret $2^{-\mathcal{D}(\hat{\mathbf{w}})}$ as a prior over possible models $\mathbf{w}$. In fact, this is done explicitly in the Minimum Message Length (MML) principle which is a Bayesian analogue of MDL, which chooses the model $\hat{\mathbf{w}}$ with maximum $P(\mathbf{w}|\mathbf{Y})$, that is, it chooses the model that minimizes
$-\lg P(\mathbf{w}|\mathbf{Y}) = -\lg P(\mathbf{w}) - \lg P(\mathbf{Y}|\mathbf{w}) + const.$

values, and use a code for integers to specify which grid point is closest. A simple way to approximate the value of a particular coefficient $\hat{w}$ is to encode an integer version of its z-score relative to the null-hypothesis value w_0 (which in our case is 0):

$$\left\langle \frac{\hat{w}-w_0}{SE(\hat{w})} \right\rangle = \left\langle \frac{\hat{w}}{SE(\hat{w})} \right\rangle,$$

where $\langle x \rangle$ means the closest integer to x and SE represents standard error. The z-score can be coded with the idealized universal code for the positive integers of Rissanen (1983), in which the cost to code $i \in 1,2,3,...$ is

$$\lg^* i + b,$$

where $\lg^* i = \lg i + \lg\lg i + \lg\lg\lg i + \ldots$ so long as the terms remain positive, and $b \approx \lg 2.865 \approx 1.516$ is the constant such that

$$\sum_{i=1}^{\infty} 2^{-(\lg^* i + b)} = 1.$$

We require the $\lg^*$ instead of a simple lg because the number of bits Sender uses to convey the integer i will vary, and she needs to tell the Receiver how many bits to expect. The number of bits is itself an integer than can be coded, hence the iteration of logarithms.

In fact, in practice it is unnecessary to allow our integer code to extend to arbitrarily large integers. We are interested in features near the limit of detectability and we expect our z-scores to be roughly in the range ~ 2 to ~ 4, since if they were much higher, the true features would be obvious and would not require sensitive feature selection. We could thus impose some maximum possible z-score Z that we might ever want to encode (say, 1000) and assume that all of our z-scores will fall below it. In this case, the constant c can be reduced to a new value c_Z, now only being large enough that,

$$\sum_{i=1}^{Z} 2^{(-\lg^* i + c_Z)} = 1. \tag{3}$$

In particular $c_{1000} \approx 1.199$. In our implementation in this paper, we avoid computing the actual values of our z-scores and instead assume a constant 2 bits per coefficient. The reason behind choosing 2 bits over using a more conservative penalty like BIC (Bayesian Information Criterion) $(\lg n)$ bits is that using a fewer number of bits allows us to select even those features which provide marginal benefit. This is important since our goal is to build sparse models with better predictive accuracy rather than identifying the correct set of sparse features. We explain later in the section on Model Consistency that using a liberal penalty of 2 bits to code a coefficient allows us to achieve finite *risk-inflation* (Foster and George, 1994).

Combining the cost of the residuals with the cost of the model gives the following formula for the description length as a function of number of features that we include in the model:

$$-\lg P(\mathbf{Y}|\hat{\mathbf{w}}) + q(\lg p + 2), \tag{4}$$

where q is the number of features in the model and p is the total number of candidate features.

The above formula represents the simplest possible coding scenario and we will refer to it later in the paper as "**Baseline Coding Scheme**" when we propose more complex coding schemes for the problems of simultaneous feature selection for a set of multiple related tasks and grouped feature selection for a single task.

4. Multiple Inclusion Criterion (MIC)

MIC is a general framework for ℓ_0 penalty based greedy feature selection which minimizes a cost function provided by the Minimum Description Length (MDL) principle. MIC provides an elegant way of incorporating arbitrary sparsity patterns in the feature space by using MDL coding schemes customized to the problem at hand. In this section, we describe how MIC can be used to provide statistically efficient models for the problems of simultaneous feature selection for multiple related tasks and grouped feature selection for a single task. To do that, we first introduce some more notation and follow up on the MDL introduction in Section 3.1.

For the problem of simultaneous feature selection for a set of related tasks (which is addressed using MIC-MULTI) we assume a set of h regression or classification tasks which can potentially share a set of p features and a total of n labeled training examples. The task is to learn a set of joint ("shared") models for all the h tasks. We represent the feature, response and the weight matrices as $\mathbf{X}_{n\times p}$, $\mathbf{Y}_{n\times h}$ and $\mathbf{w}_{p\times h}$ respectively. Additionally, for simplicity of analysis we assume a linear regression setting of the form[4] $\mathbf{Y} = \mathbf{w}\cdot\mathbf{X} + \varepsilon$ with a Gaussian noise term $\varepsilon_{n\times h}$. Note that the noise on the responses (ε) may be correlated; for instance, if our responses consist of temperature measurements at various locations, taken with the same thermometer, then if our thermometer drifted high at one location, it will have been high at the other location also. Thus, we take the rows of ε to have non-zero covariance:

$$\varepsilon_i \sim \mathcal{N}_h(0, \mathbf{\Sigma}),$$

where ε_i is the i^{th} row of ε and $\mathbf{\Sigma}$ is an arbitrary $h \times h$ covariance matrix.

Similarly, for the related problem of grouped feature selection (which is addressed using MIC-GROUP) also, we have a total of p candidate features which are further divided into K groups (equal or unequal). Again, we assume the availability of a (fixed number) n of labeled training examples. Just as above we can represent the feature, response and weight matrices as $\mathbf{X}_{n\times p}$, $\mathbf{Y}_{n\times 1}$ and $\mathbf{w}_{p\times 1}$ respectively.

Let S represent the total description length (TDL) of the MDL message that is exchanged between the Sender and the Receiver. In the case of MIC-MULTI, S is the combined message length for all h tasks and hence we select features for all the h tasks simultaneously to minimize S and in the case of MIC-GROUP it can either be the combined message length for all the features within a given group (feature class) (MIC-GROUP(I)) or the message length of a given feature (MIC-GROUP-SC). Thus, when we evaluate a feature for addition into the model, we want to maximize the reduction of TDL by adding that feature to our model. More formally, at each iteration we greedily add those features to our model that:

$$\begin{aligned} \Delta S^i &= \Delta S^i_E - \Delta S^i_M, \\ \textit{Best Feature} &= \underset{\hat{i}}{\operatorname{argmax}}\{\Delta S^i\}, \end{aligned}$$

where $\Delta S_E \geq 0$ is the reduction in residual-error coding cost, that is, the first term on right hand side in Equation 1, due to the increase in data likelihood given this new feature and $\Delta S_M > 0$ is the

4. It can be extended to the standard classification setting by replacing the squared loss with a logistic loss, but due to lack of closed form solutions for logistic regression and since correlation between residuals is inconvenient to model in classification settings, we refrain from analyzing them.

increase in model cost to encode the new feature (second term in Equation 1) and i ranges over all the p features.

In the next subsections we describe how we code the S_E and S_M terms (i.e., the residual error and model) for MIC-MULTI and MIC-GROUP in detail.

4.1 MIC-MULTI

As mentioned earlier, MIC-MULTI borrows strength across multiple tasks and hence selects a joint set of features for related tasks (Dhillon et al., 2009).

4.1.1 CODING THE MODEL

MIC-MULTI borrows strength across responses by efficiently specifying the feature-response pairs in the $p \times h$ matrix $\hat{w}$. The naïve approach would be to put each of the ph coefficients in a linear order and specify the index of the desired coefficient using $\lg(mh)$ bits. But we can do better. If we expect nearly all the responses to be correlated with the predictive features, we could give all the responses nonzero coefficients (using $2h$ bits to code each of the h response coefficients) and simply specify the feature that we are talking about by using $\lg p$ bits, as in Section 3.1.2. From now on we will refer to this approach as FULL-MIC-MULTI (*fully dependent MIC*-MULTI) coding scheme, as it assumes that a selected feature will be added in the models of all the tasks, in much the same way as BBLASSO (Obozinski et al., 2009). Another limiting case is the one when we do feature selection for all the tasks independently (the baseline "Independent" Coding Scheme); the coding scheme in that case takes the form given in Equation 4.

However, these assumptions are usually unrealistic; each feature is generally neither correlated with almost all the responses nor with none of the responses, but is rather correlated with a few of them. A more flexible coding scheme would allow us to specify only the subset of the responses to which we want to give nonzero coefficients. For instance, suppose we are considering feature number 2609; and, of the $h = 20$ responses, we think that only $\{3, 7, 14, 17\}$ should have nonzero coefficients with the current feature. Then, we can use $\lg p$ bits to specify our feature (number 2609) once, and then we can list the particular responses that have nonzero coefficients with feature 2609, thereby avoiding paying the cost of $\lg(mh)$ four times to specify each coefficient in isolation.

A standard practice in information theory literature to code a subset of size h is to first specify how many $k \leq h$ elements the subset contains and then which of the $\binom{h}{k}$ possible subsets with k elements we are referring to Cover and Thomas (2006). In particular, we choose to code k using $\lg^* k + c_h$ bits, with c_h as defined in Equation 3. We then need $\lg \binom{h}{k}$ additional bits to specify the particular subset. We refer to this code as *partially dependent* MIC-MULTI or simply PARTIAL-MIC-MULTI.

The total cost (S_M^i) to code the model of a feature for MIC-MULTI is composed of three parts as follows:

$$S_M^i = \ell_H + \ell_I + \ell_\theta,$$

where ℓ_H is the number of bits needed to specify the subset k of the h tasks models in which to include the feature; ℓ_I is the number of bits used to describe which feature is being added and ℓ_θ is the description length of the coefficients of non-zero features.

We have already described the cost for ℓ_H above; it is equal to:

$$\ell_H = \lg^* k + c_h + \lg \binom{h}{k}.$$

For ℓ_0, we use a cost of 2 bits per coefficient, the motivation for which was described earlier in Section 3.1.2. For ℓ_I, which specifies the size of the code for the given feature, we use $\lg p$ bits, which is equivalent to a uniform prior over the features,[5] that is, each feature is equally likely to be selected. This can be accomplished by simply keeping a linear array of features and coding the indices of the features with nonzero coefficients.

Thus, we can represent the total model cost for MIC-MULTI as:

$$S_M = \left(\lg^* k + c_h + \lg \binom{h}{k}\right) + (\lg p) + (2k). \tag{5}$$

4.1.2 CODING THE DATA

Let $\mathbf{E}$ be the residual error ($\mathbf{Y}$ - $\mathbf{X}\cdot\mathbf{w}$) matrix, and as mentioned above, let ε_i, $i = 1,2,\ldots,n$ denote the i^{th} row of the error and let $\mathbf{\Sigma}$ be its $h \times h$ covariance matrix. The model likelihood under the Gaussian assumption[6] can be written as:

$$P(Y_i|\hat{w}_q) = \frac{1}{\sqrt{(2\pi)^h|\Sigma|}} \exp\left(\frac{1}{2}\varepsilon_i^T \mathbf{\Sigma}^{-1}\varepsilon_i\right), \tag{6}$$

$$\begin{aligned} S_E &= -\lg \prod_{i=1}^{n} P(Y_i|\hat{w}_q) \\ &= \frac{1}{2\ln 2}\left[n\ln\left((2\pi)^h|\mathbf{\Sigma}|\right) + \sum_{i=1}^{n}(Y_i - X_i\cdot\hat{w}_q)^T\mathbf{\Sigma}^{-1}(Y_i - X_i\cdot\hat{w}_q)\right] \end{aligned} \tag{7}$$

with subscript i denoting the i^{th} row. Since $\mathbf{\Sigma}$ is in fact unknown, we estimate it using maximum likelihood (ML):

$$\hat{\Sigma}_F = \frac{1}{n}(\mathbf{Y} - \mathbf{X}\cdot\hat{w}_{q-1})^T(\mathbf{Y} - \mathbf{X}\cdot\hat{w}_{q-1}),$$

where the subscript F stands for "full covariance", and we use $\hat{w}_{q-1}$ to get ML estimate, instead of $\hat{w}_q$ to prevent overfitting, as we mentioned in Section 3.1.1.

In practice, we find that estimating all the h^2 entries of the covariance matrix can lead to overfitting. Therefore we use shrunken estimates of the form $\hat{\Sigma}_\lambda = \lambda\hat{\Sigma}_D + (1-\lambda)\hat{\Sigma}_F$ for $(\lambda \in [0,1])$, which tend to work well. Note that we do not need to pay an extra coding cost for estimating $\mathbf{\Sigma}$ as we are using a *prequential* coding scheme; $\mathbf{\Sigma}$ is calculated using information that was already paid for. We describe more technical details about our implementation in the Experiments section.

S_E as described in Equation 7 and S_M as Equation 5 form the quantities $\mathcal{D}(\mathbf{Y}|\hat{\mathbf{w}})$ and $\mathcal{D}(\hat{\mathbf{w}})$ in Equation 1. Thus our objective becomes $\hat{\mathbf{w}}^* = \underset{\hat{\mathbf{w}}}{\operatorname{argmin}}\{S_E + S_M\}$.

4.1.3 COMPARISON OF VARIOUS MIC-MULTI CODING SCHEMES

In this section we discussed three MDL based information-theoretic approaches to multitask feature selection, namely FULL-MIC-MULTI, Baseline "Independent" Coding Scheme and PARTIAL-MIC-MULTI. In general, the negative log-likelihood portion of *Independent* may differ from that of

5. The uniform code gives the worst-case minimax optimal code lengths (Grünwald, 2005).
6. As mentioned earlier, we are considering linear regression for simplicity of analysis and ease of modeling the correlation between residuals.

the other two methods, because *Full* and *Partial* can use a non-diagonal covariance estimate like $\hat{\Sigma}_F$ or $\hat{\Sigma}_\lambda$, while *Independent* only operates on one response at a time, and thus implicitly uses $\hat{\Sigma}_D$. However, since we generally use $\hat{\Sigma}_\lambda$, as mentioned earlier, for *Full* and *Partial*, and since $\hat{\Sigma}_\lambda$ approximates the diagonal $\hat{\Sigma}_D$ (for λ close to 1), the real difference comes from the coding schemes.

The coding costs for these three methods are compared in Table 1 for $p = 2000$ features, $h = 20$ responses, and for various values of k, the number of responses to which we add the current feature under consideration. FULL-MIC-MULTI is only allowed to take $k = 0$ or $k = h$, so it has h nonzero coefficients in all three rows of the table. However, if the extra $h - k$ coefficients correspond to non-predictive features, the extra reduction in residual-coding cost that FULL-MIC-MULTI enjoys over the other methods is likely to be small. As expected, each coding scheme is cheapest in the case for which it was designed; however, the MIC-MULTI methods are never excessively expensive, unlike *Independent* for $k = h$.

k	PARTIAL-MIC-MULTI		FULL-MIC-MULTI		Baseline (Independent)	
1	$\log p + c_h + \log h + 2$	[18.4]	$\log p + 2h$	[51.0]	$\log p + 2$	**[13.0]**
$\frac{h}{4}$	$\log p + \log^* \binom{h}{\frac{h}{4}} + c_h + \log \binom{h}{h/4} + \frac{h}{2}$	**[39.8]**	$\log p + 2h$	[51.0]	$\frac{h}{4} \log p + \frac{h}{2}$	[64.8]
h	$\log p + \log^* h + c_h + 2h$	[59.7]	$\log p + 2h$	**[51.0]**	$h \log p + 2h$	[259.3]

Table 1: Costs in bits for each of the three schemes to code a model with $k = 1$, $k = \frac{h}{4}$, and $k = h$ nonzero coefficients. $p \gg h \gg 1$, $\ell_I = \lg p$, $\ell_{\hat{\theta}} = 2$, and for $h \in \{5, \ldots, 1000\}$, $c_h \approx 1$. Examples of these values for $p = 2{,}000$ and $h = 20$ appear in brackets; the smallest of the costs appears in bold. *Note:* The costs are given per feature.

4.2 MIC-GROUP

MIC-GROUP is the algorithm for grouped feature selection, when features fall into *groups* or *classes* (Dhillon et al., 2008; Yuan and Lin, 2006; Bach et al., 2004). For example, genes can be divided into gene classes based on what pathway they occur in or features of a word can be grouped based on whether they are based on specific neighbouring words, parts of speech, or more global document properties. More generically, starting from any set of features, one can generate new classes of features by using projections such as principle components analysis (PCA) or non-negative matrix factorization (NNMF), transformations such as log or square root, and interactions (products of features) (Dhillon et al., 2010). The problem of grouped feature selection (MIC-GROUP) is very closely related to the problem of simultaneous feature selection for a set of related tasks (MIC-MULTI) as has also been pointed out by Obozinski et al. (2009). The multi-task problem we described earlier can also be thought of as a grouped feature selection scenario in which a group is defined by fixing a specific feature and ranging over multiple tasks. Our MIC based models for these two problems also follow the same intuition; in (MIC-MULTI) the tendency is to add a given feature into models of more and more tasks[7] and similarly in (MIC-GROUP) the tendency is to add more and more features from the same group as the whole rationale behind doing grouped

7. The $\lg \binom{h}{k}$ part of the model cost is only small when k is small or it is very large, that is, $k \approx h$ as $\binom{h}{k} = \binom{h}{h-k}$.

feature selection is based on the fact that some feature groups contain highly predictive features than others.

4.2.1 CODING SCHEMES FOR MIC-GROUP

Since the problem of grouped feature selection is similar to the problem of simultaneous feature selection for a set of related tasks, we can propose a coding scheme which is analogous to the coding scheme for MIC-MULTI. For example, in this case also we can code the data as $P(Y_i|\hat{w}_q) = \frac{1}{\sqrt{(2\pi)^{h_{single}}|\mathbf{\Sigma}|}} \exp\left(\frac{1}{2}\varepsilon_i^T \mathbf{\Sigma}^{-1}\varepsilon_i\right)$ in a similar fashion as Equation 6 where h_{single} is the number of features in a given group (feature class) and we will estimate the covariance matrix, which represents covariance between different features in the same group (feature class), in a similar way as we did for MIC-MULTI, that is, by Maximum Likelihood Estimation. Remember, that in this case S_E term will be the message length for all the features within a given feature class. In a similar fashion, the number of bits to code the model can be represented as $S_M = \left[\lg^* k + c_h + \lg\binom{h_{single}}{k}\right] + \log p + 2k$, which corresponds to Equation 5.[8] The other mechanics of the coding scheme will also be the same as for MIC-MULTI as this time we are trying to find a best subset of size k in a group (feature class) of size h_{single} and so we do a stepwise greedy search as earlier. From now on we refer to this coding scheme as MIC-GROUP (I).

Although this coding scheme works very well in practice, but it turns out that we are not exploiting the full flexibility that MDL based coding offer us. So, we propose a new coding scheme, which is computationally more efficient than MIC-GROUP (I), as it does not require a stepwise search for subset selection, though the predictive accuracy of both these coding schemes is comparable. We call this new computationally efficient coding scheme which uses "Switch Coding" as MIC-GROUP-SC and it is explained in detail below.

Coding the data with MIC-GROUP-SC (MIC-GROUP-Switch Coding): In this new coding scheme S_E is the message length for a single feature and ΔS_E represents the increase in likelihood of the data by adding that feature to the model.

Let $\mathbf{E}$ be the residual error ($\mathbf{Y}$ - $\mathbf{X}\cdot\mathbf{w}$) matrix as earlier, and let ε_i, $i = 1, 2, \ldots, n$ denote the i^{th} row of the error and let σ be variance of the Gaussian noise. The model likelihood can be written as:

$$\begin{aligned} P(Y_i|\hat{w}_q) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{\varepsilon_i^2}{2\sigma^2}), \\ S_E &= -\lg \prod_{i=1}^{n} P(Y_i|\hat{w}_q). \end{aligned} \tag{8}$$

This equation is similar to the corresponding equation for MIC-MULTI except that here we only have a single response (task). In this case also, the variance σ^2 is estimated using the Maximum Likelihood principle.

Coding the model with MIC-GROUP-SC: This is where we differ from MIC-GROUP (I) and we use a coding scheme better suited to the group structure of the features. The intuition behind this coding scheme is that once we have selected (at least) one feature from a given group then it should

8. For simplicity of analysis and ease of comparison with coding schemes for MIC-MULTI we are assuming that all groups are of the same size h_{single}, though in reality the groups may be of unequal size and the same coding scheme still holds.

become easier to select more features from the group. In other words, the cost of adding more features from the same feature class should be low. The total cost of the model is composed of three parts:

$$S_M^i = \ell_C + \ell_I + \ell_\theta,$$

where ℓ_C is the number of bits used to code the index of the group of the evaluated feature, ℓ_I is the number of bits to code the index of the evaluated feature (within that particular group) and ℓ_θ is the cost to code the coefficient of the evaluated feature.

Our coding scheme assumes a uniform prior over all the groups; that is, each group is equally likely to contain beneficial features.[9] So ℓ_C is $\lg K$ where K is the total number of groups (feature classes) in the data. Now, if a feature gets selected from a group (feature class) from which we had previously selected features, then we can save some bits by using "switch" coding and coding ℓ_C using only $1 + \lg Q$ bits where Q is the total number of groups (feature classes) included in the model till that point of time and 1 bit is used to represent that this group (feature class) has previously produced beneficial features. (Think of keeping an indexed list of length Q of the feature classes that have been selected). This is where our method wins over other methods and we do not need to code the same feature class over and over again if it has produced beneficial features in the past. Therefore ℓ_C is (Note that we added 1 bit to $\lg K$ also to ensure that the group whose index starts with 1 is not confused with the "switch".):

$$\ell_C = \begin{cases} 1 + \lg K & \textit{if the feature class is not in the model} \\ 1 + \lg Q & \textit{if the feature class is already in the model.} \end{cases}$$

To code ℓ_I we again assume a uniform prior over all the features within that particular group. This corresponds to $\lg m_i$ bits where m_i is the total number of features in the feature class of which the i^{th} feature is a part of. This is pretty similar to RIC (Risk Inflation Criterion) style (Foster and George, 1994) coding or the widely use Bonferroni penalty. Finally, to code ℓ_θ we use 2 bits per coefficient, the motivation for which was described earlier. Therefore, the model cost per feature can be represented as:

$$S_M^i = (\ell_C) + (\lg m_i) + 2. \tag{9}$$

As mentioned earlier, this coding scheme is computationally cheaper than MIC-GROUP (I) as it does not require a subset search every time a feature is added to the model and it provides comparable predictive accuracy to MIC-GROUP (I). Note that just analogous to MIC-MULTI it is possible to come up with a new coding scheme called FULL MIC-GROUP(I) which just like its MIC-MULTI counterpart would add all the features from a given group (feature class) into the model. The MIC-GROUP schemes presented here are the most general setting and are analogous to PARTIAL MIC-MULTI for the multi response (task) scenario.

Just as with MIC-MULTI, here we optimize the objective $\hat{\mathbf{w}}^* = \underset{\hat{\mathbf{w}}}{\operatorname{argmin}}\{S_E + S_M\}$ with the S_E and S_M terms as described by Equations 8 and 9 respectively.

9. This is actually a pretty good assumption as mentioned earlier. The uniform code gives the worst-case minimax optimal code lengths (Grünwald, 2005) and hence it is reasonable to use it if the data distribution is *completely unknown* or if *no* distribution is assumed.

4.3 Algorithms and Implementation Details

In this subsection we outline the algorithms for MIC-MULTI and MIC-GROUP and also explain some details of the search strategy that we used for efficient subset search in case of MIC-MULTI.

4.3.1 ALGORITHMS

The algorithm for MIC-MULTI is as described in Algorithm 1. We provide algorithm for the most general case, that is, PARTIAL MIC-MULTI as the other two cases, that is, *Full* and *Independent* are the special cases of this scenario.

Algorithm 1 PARTIAL MIC-MULTI

Include the intercept (feature number 1) in all h response models.
$remaining_features = \{2, \ldots, p\}$.
$keep_adding_features =$ true.
while $keep_adding_features$ **do**
  for j in $remaining_features$ **do**
    // Find the best subset of response models to which to add feature j.
    for $k = 1$ to h **do**
      Try including feature j in the best k response models. (We greedily assume that the best k responses are the union of the best $k-1$ responses with the remaining response that, if included, would most increase likelihood.)
      Compute ΔS^k_{jE}, the decrease in data residual cost, and ΔS^k_{jM}, the resulting increase in model-coding cost, relative to not including feature j in any response models.
    end for
    Let k_j be the value of k that maximizes $\Delta S^k_{jE} - \Delta S^k_{jM}$.
    $\Delta S_j := \Delta S^{k_j}_{jE} - \Delta S^{k_j}_{jM}$.
  end for
  Let j^* be the feature j that maximizes ΔS_j, the reduction in TDL for adding feature j.
  if $\Delta S_{j^*} > 0$ **then**
    Add feature j^* to the appropriate k_{j^*} response models.
    $remaining_features = remaining_features - \{j^*\}$.
  else
    $keep_adding_features =$ false.
  end if
end while

The algorithm for MIC-GROUP-SC is described in Algorithm 2. The algorithm makes multiple passes through data and at each iteration adds the best feature to the model. It stops when no feature provides better ΔS than in the previous iteration. Since, it can be the case that it is not worth adding a single feature from a particular group (feature class) but it is still beneficial to add multiple features from that class. So, a clever search strategy that we found helpful with MIC-GROUP-SC was to use a mixed forward-backward greedy stepwise strategy in which one continues the search past the stopping criterion given in the algorithm and then sequentially removes the "worst" features from the now overfit model by making a "Backward" pass. In practice, we found this search strategy helpful. A similar hybrid forward-backward strategy was also used by Zhang (2009a).

Algorithm 2 MIC-GROUP-SC

$flag = True$; // flag for indicating when to stop
$model = \{\}$; // initially no features in model
$prev_max = 0$; // keeps track of the value of ΔS_E in the previous iteration
while {flag == True} **do**
  for {i = 1 to p} **do**
    $Compute\ \Delta S_E^i$; // Increase in likelihood by adding feature 'i' to the model
    $Compute\ \Delta S_M^i$; // Number of extra bits required to code the i^{th} feature
    $\Delta S^i := \Delta S_E^i - \Delta S_M^i$;
  end for
  $i_{max} := argmax_i\{\Delta S^i\}$; //The best feature in the current iteration
  $current_max := max_i\{\Delta S^i\}$; //The best penalized likelihood change in the current iteration
  if $\{current_max > prev_max\}$ **then**
    $model := model \bigcup \{i_{max}\}$; // Add the current feature to model
    $prev_max := current_max$;
  else
    $flag := False$;
  end if
end while

Note that we do not provide algorithm for MIC-GROUP (I) as it is pretty similar to MIC-MULTI with minor notational modifications as mentioned in the previous subsection.

4.3.2 STEPWISE SEARCH METHOD

Since MIC-MULTI requires subset search over the set of possible tasks in which to consider a feature for addition, so a discussion of our greedy search strategy is warranted.

For each feature, we evaluate the change in TDL (Total Description Length) that would result from adding that feature to the model with the optimal number of associated tasks. We add the best feature and then recompute the changes in TDL for the remaining features.[10] This continues until there are no more features that would reduce TDL if added. The number of evaluations of features for possible addition is thus $O(pp_s)$, where p_s is the number of features eventually added.

To select the optimal number k of task models in which to include a given feature, we again use a stepwise-style search. In this case, we evaluate the reduction in TDL that would result from adding the feature to each task, add the feature to the best task, recompute the reduction in TDL for the remaining tasks, and continue.[11] However, unlike a normal stepwise search, we continue this process until we have added the feature to all h task models. The reason for this is two-fold. First, because we want to borrow strength across tasks, we need to avoid overlooking cases where the correlation of a feature with any single task is insufficiently strong to warrant addition, yet the

10. Remember that TDL changes due to the increase in likelihood and the additional model coding cost due to the added feature.
11. A stepwise search that re-evaluates the quality of each task at each iteration is necessary because, if we take the covariance matrix Σ to be non-diagonal, the values of the residuals for one task may affect the likelihood of residuals for other tasks. If we take Σ to be diagonal, as we do in Section 5, then an $O(h)$ search through the tasks without re-evaluation suffices.

correlations with all of the tasks are. Second, the $\log \binom{h}{k}$ term in PARTIAL MIC-MULTI's coding cost does not increase monotonically with k, so even if adding the feature to an intermediate number of tasks does not look promising, adding it to all of them might still be worthwhile. Thus, when evaluating a given feature, we compute the description length of the model $O(h^2)$ times. Since we need to identify the optimal k for each feature evaluation, the entire algorithm requires $O(h^2 p p_s)$ evaluations of TDL.

While not shown explicitly in Algorithm 1, we use two branch-and-bound-style optimizations to cut this cost significantly in practice:

1. Before searching through subsets of responses to find the optimal subset for each feature, we make an $O(p)$ sweep through the features to compute an upper bound on the decrease in TDL that could result from adding that feature as

   $$(\text{decrease in TDL if the feature is added to all } h \text{ response models}) - \log p.$$

   Here, the first term is an upper bound on the benefit of adding the feature to the optimal number of response models (since adding a feature can only make a model fit better), and the second term underestimates the model cost of adding the feature, regardless of how many response models would actually be used. We sort the features in decreasing order by this upper bound, and when we reach features whose upper bounds are less than the best actual decrease in TDL observed so far, we terminate the search early.

2. For the stepwise search over responses, we can bound from above the potential benefit of adding the feature to k response models as

   $$(\text{decrease in TDL if the feature is added to all } h \text{ response models}) - \left(\log^* k + c_k + \log \binom{h}{k} + 2k \right),$$

   where the subtracted term represents the coding cost of including the feature in k response models. We can stop the search early when no higher value of k has an upper bound that exceeds the best reduction in TDL seen so far for any feature's response subset.[12]

5. Experimental Results

In this section we empirically show the usefulness of our MIC based models (MIC [MULTI and GROUP]) on a variety of real world data sets pertaining to Genomics and Computational Linguistics (particularly Word Sense Disambiguation) domains. Besides this we also show results on synthetic data sets to illustrate the cases when our models are most beneficial.

It is important to note that we are interested in maximizing the predictive accuracy in these experiments rather than the totally orthogonal and antagonistic objective of identifying the correct set of sparse features ("sparsity pattern consistency") at the expense of predictive accuracy. We would like to note that much sparser results can be obtained by using our models by using more conservative coding schemes and also by ℓ_1/ℓ_2 penalty (BBLASSO, GROUP LASSO) models by using thresholding techniques like Zhou (2009) and Lounici (2008).

12. We say "no higher value of k" rather than "the next higher value of k" because (10) does not decrease monotonically with k, due to the $\log \binom{h}{k}$ quantity.

5.1 MIC-MULTI

In this section, we first evaluate the MIC-MULTI approach on three synthetic data sets, each of which is designed to match the assumptions of, respectively, the PARTIAL and FULL MIC-MULTI, and Baseline (Independent) coding scheme (Equation 4). We then test the methods on two biological data sets, a Yeast Growth data set (Perlstein et al., 2007), which consists of real-valued growth measurements of multiple strains of yeast under different drug conditions, and a Breast Cancer data set (van 't Veer et al., 2002), which involves predicting prognosis, ER (Estrogen Receptor) status, and three other descriptive variables from gene-expression values for different cell lines.

We compare the three coding schemes of Section 4.1.3 against two other multitask algorithms: ANDOZHANG (Ando and Zhang, 2005) and BBLASSO (Obozinski et al., 2009), as implemented in the Berkeley Transfer Learning Toolkit (Rakhlin, 2007). We did not compare MIC-MULTI with other methods from the toolkit as they all require the data to have additional structure, such as *meta-features* (Lee et al., 2007; Raina et al., 2006), or expect the features to be frequency counts, such as for the Hierarchical Dirichlet Processes algorithm. Also, none of the neglected methods does feature selection.

For ANDOZHANG we use 5-fold CV to find the best value of the parameter that Ando and Zhang (2005) call h (the dimension of the subspace Θ, not to be confused with h as we use it in this paper). We tried values in the range $[1, 100]$ as is done in Ando and Zhang (2005).

MIC-MULTI, as presented in Section 4.1.2, is a regression algorithm, but ANDOZHANG and BBLASSO are both designed for classification. Therefore, we made each of our responses binary 0/1 values before applying MIC-MULTI with a regular regression likelihood term. Once the features were selected, however, we used logistic regression applied to just those features to obtain MIC-MULTI's actual model coefficients.

As noted in Section 4.1.2, MIC-MULTI's negative log-likelihood term can be computed with an arbitrary $h \times h$ covariance matrix $\mathbf{\Sigma}$ among the h tasks. We did not estimate all the h^2 entries of $\mathbf{\Sigma}$ as it lead to overfitting, so we instead took $\mathbf{\Sigma}$ to be diagonal.[13]

5.1.1 EVALUATION ON SYNTHETIC DATA SETS

We created synthetic data according to three separate scenarios—called *Partial*, *Full*, and *Independent*. For each scenario, we generated a matrix of continuous responses as

$$\mathbf{Y}_{n\times h} = \mathbf{X}_{n\times p} \cdot \mathbf{w}_{p\times h} + \varepsilon_{n\times h},$$

where $p = 2000$ features, $h = 20$ responses, and $n = 100$ observations. Then, to produce binary responses, we set to 1 those response values that were greater than or equal to the average value for their column and set to 0 the rest; this produced a roughly 50-50 split between 1's and 0's because of the normality of the data. Each nonzero entry of $\mathbf{w}$ was i.i.d. $\mathcal{N}(0,1)$, and entry of ε was i.i.d. $\mathcal{N}(0,0.1)$, with no covariance among the ε entries for different tasks. Each task had $p^* = 4$ beneficial features, that is, each column of $\mathbf{w}$ had 4 nonzero entries.

The scenarios differed according to the distribution of the beneficial features in $\mathbf{w}$.

13. Informal experiments showed that estimating $\mathbf{\Sigma}$ as a convex combination of the full and diagonal estimates (i.e., $\hat{\Sigma}_\lambda$) also works well but we chose to use diagonal $\mathbf{\Sigma}$ (i.e., $\hat{\Sigma}_D$) due to its simplicity and to show the advantage of using a better coding scheme to code the model as by using diagonal $\mathbf{\Sigma}$ Partial and Independent methods are the same except S_M (i.e., cost of coding the model).

- In the *Partial* scenario, the first feature was shared across all 20 responses, the second was shared across the first 15 responses, the third across the first 10 responses, and the fourth across the first 5 responses. Because each response had four features, those responses $(6-20)$ that did not have all of the first four features had other features randomly distributed among the remaining features $(5, 6, \ldots, 2000)$.
- In the *Full* scenario, each response shared exactly features $1-4$, with none of features $5-2000$ being part of the model.
- In the *Independent* scenario, each response had four random features among candidate features $1, \ldots, 2000$.

For the synthetic data, we report precision and recall to measure the quality of feature selection. This can be done both at a coefficient[14] level (Was each nonzero coefficient in w correctly identified as nonzero, and vice versa?) and at an overall feature level (For features with *any* nonzero coefficients, did we correctly identify them as having nonzero coefficients for any of the tasks, and vice versa?). Note that Full MIC-MULTI and BBLASSO always make entire rows of their estimated w matrices nonzero and so tend to have larger numbers of nonzero coefficients. Table 2 shows the performance of each of the methods on five instances of the Partial, Full, and Independent synthetic data sets. On the *Partial* data set, PARTIAL MIC-MULTI performed the best, closely followed by BASELINE (INDEPENDENT); on the *Full* synthetic data, FULL MIC-MULTI and PARTIAL MIC-MULTI performed equally well; and on the *Independent* synthetic data, the *Baseline* algorithm performed the best closely followed by PARTIAL MIC-MULTI. It is also worth noting that the best-performing methods tended to have the best precision and recall on coefficient selection. The performance trends of the three methods are in consonance with the theory of Section 4.1.3.

The table shows that only in one of the three cases does non-MIC methods compete with MIC methods. BBLASSO on the Full synthetic data shows comparable performance to the MIC methods, but even in that case it has a very low feature precision, since it added many more spurious features than the MIC methods.

5.1.2 EVALUATION ON REAL DATA SETS

This section compares the performance of MIC-MULTI methods with ANDOZHANG and BBLASSO on a Yeast and a Breast Cancer data set. These are typical biological data sets in that only a handful of features are predictive from thousands of potential features. This is precisely the case in which MIC-MULTI outperforms other methods. MIC-MULTI not only gives better accuracy, but does so by choosing fewer features than BBLASSO's $\ell_1 - \ell_2$-based approach.

Yeast Data Set: Our Yeast data set comes from Perlstein et al. (2007). It consists of real-valued growth measurements of 104 strains of yeast ($n = 104$ observations) under 313 drug conditions. In order to make computations faster, we hierarchically clustered these 313 conditions into 20 groups using correlation as the similarity measure. Taking the average of the values in each cluster produced $h = 20$ real-valued responses (tasks), which we then binarized into two categories: values at least as big as the average for that response (set to 1) and values below the average (set to 0).The features consisted of 526 markers (binary values indicating major or minor allele) and 6,189 transcript levels in rich media for a total of $p = 6715$ features.

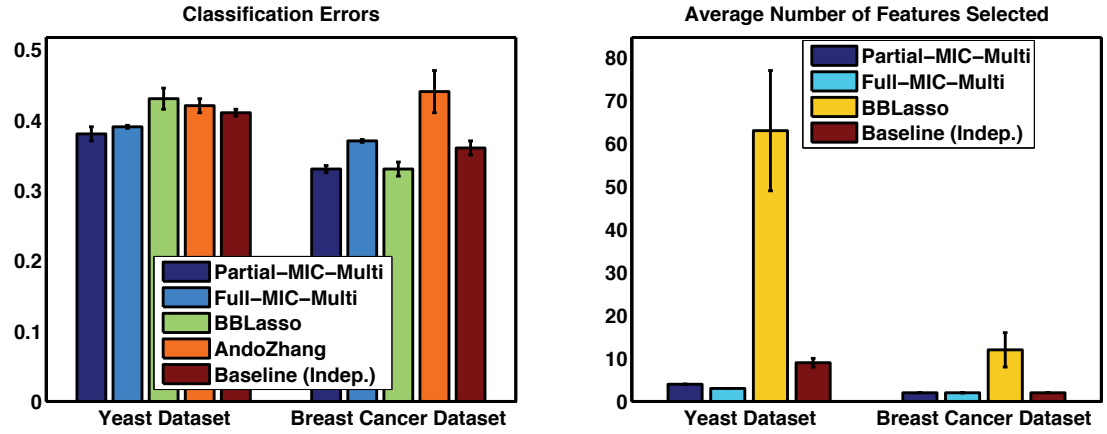
14. A coefficient is defined as the addition of a given feature to a single task. For example if a feature was added to models of 10 tasks, then 1 feature and 10 coefficients were selected.

Method	Test Error $\mu \pm \sigma$	Coefficient Precision/Recall	Feature Precision/Recall
Partial Synthetic Data Set			
TRUE MODEL	0.07 ± 0.00	$1.00 \pm 0.00/1.00 \pm 0.00$	$1.00 \pm 0.00/1.00 \pm 0.00$
PARTIAL MIC-MULTI	$\mathbf{0.10 \pm 0.00}$	$0.84 \pm 0.02/0.77 \pm 0.02$	$0.99 \pm 0.01/0.54 \pm 0.05$
FULL MIC-MULTI	0.17 ± 0.01	$0.26 \pm 0.01/0.71 \pm 0.03$	$0.97 \pm 0.02/0.32 \pm 0.03$
BASELINE (INDEPENDENT)	0.12 ± 0.01	$0.84 \pm 0.02/0.56 \pm 0.02$	$0.72 \pm 0.05/0.62 \pm 0.04$
BBLASSO	0.19 ± 0.01	$0.04 \pm 0.00/0.81 \pm 0.02$	$0.20 \pm 0.03/0.54 \pm 0.01$
ANDOZHANG	0.50 ± 0.02	NA	NA
Full Synthetic Data Set			
TRUE MODEL	0.07 ± 0.00	$1.00 \pm 0.00/1.00 \pm 0.00$	$1.00 \pm 0.00/1.00 \pm 0.00$
PARTIAL MIC-MULTI	$\mathbf{0.08 \pm 0.00}$	$0.98 \pm 0.01/1.00 \pm 0.00$	$0.80 \pm 0.00/1.00 \pm 0.00$
FULL MIC-MULTI	$\mathbf{0.08 \pm 0.00}$	$0.80 \pm 0.00/1.00 \pm 0.00$	$0.80 \pm 0.00/1.00 \pm 0.00$
BASELINE (INDEPENDENT)	0.11 ± 0.01	$0.86 \pm 0.02/0.63 \pm 0.02$	$0.36 \pm 0.06/1.00 \pm 0.00$
BBLASSO	0.09 ± 0.00	$0.33 \pm 0.03/1.00 \pm 0.00$	$0.33 \pm 0.17/1.00 \pm 0.00$
ANDOZHANG	0.45 ± 0.02	NA	NA
Independent Synthetic Data Set			
TRUE MODEL	0.07 ± 0.00	$1.00 \pm 0.00/1.00 \pm 0.00$	$1.00 \pm 0.00/1.00 \pm 0.00$
PARTIAL MIC-MULTI	0.17 ± 0.01	$0.95 \pm 0.01/0.44 \pm 0.02$	$1.00 \pm 0.00/0.44 \pm 0.02$
FULL MIC-MULTI	0.36 ± 0.01	$0.06 \pm 0.01/0.15 \pm 0.02$	$1.00 \pm 0.00/0.14 \pm 0.02$
BASELINE (INDEPENDENT)	$\mathbf{0.13 \pm 0.01}$	$0.84 \pm 0.02/0.58 \pm 0.02$	$0.83 \pm 0.02/0.58 \pm 0.03$
BBLASSO	0.35 ± 0.01	$0.02 \pm 0.00/0.43 \pm 0.02$	$0.30 \pm 0.05/0.42 \pm 0.06$
ANDOZHANG	0.49 ± 0.00	NA	NA

Table 2: Test-set accuracy, precision, and recall of MIC-MULTI and other methods on 5 instances of various synthetic data sets generated as described in Section 5.1.1. Standard errors are reported over each task; that is, with 5 data sets and 20 tasks per data set, the standard errors represent the sample standard deviation of 100 values divided by $\sqrt{100}$. *Note:* ANDOZHANG's NA values are due to the fact that it does not explicitly select features.

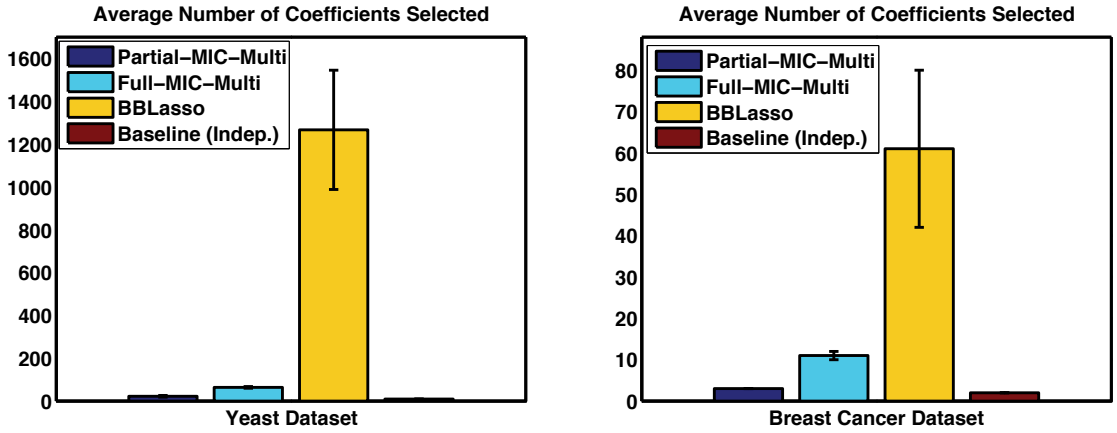
Figure 1 (a) shows classification test errors from 5-fold CV on this data set. As can be seen from the table, PARTIAL MIC-MULTI performs better than BBLASSO or ANDOZHANG. BASELINE and FULL MIC-MULTI perform slightly worse than PARTIAL MIC-MULTI, underscoring the point that it is preferable to use a more general MIC coding scheme compared to FULL MIC-MULTI or BASELINE. The latter methods have strong underlying assumptions, which cannot always correctly capture sharing across tasks.

Breast Cancer Data Set: Our second data set pertains to Breast Cancer, and contains data from five of the seven data sets used in van 't Veer et al. (2002). It contains 1171 observations for 22,268 RMA-normalized gene-expression values. We considered five associated responses (tasks); two were binary—prognosis ("good" or "poor") and ER (Estrogen Receptor) status ("positive" or "negative")—and three were not—age (in years), tumor size (in mm), and grade (1, 2, or 3). We binarized the three non-binary responses into two categories: Response values at least as high as the average, and values below the average. Finally we scaled the data set down to $n = 100$ and



(a) (b)

Figure 1: Accuracy and number of features selected on five folds of CV for the Yeast and Breast Cancer data sets. *Note:* 1). Remember that we are interested in better predictive accuracy and not in identifying the correct set of sparse features; we can get much sparser models if instead our objective is choosing the correct set of sparse features. 2). ANDOZHANG's average number of features selected are not present in the graph as it does not explicitly select features. 3). These are true cross-validation accuracies and no parameters have been tuned on them.



(a) (b)

Figure 2: Number of coefficients selected on five folds of CV for the Yeast and Breast Cancer data sets. *Note:* 1). ANDOZHANG's average number of coefficients selected are not present in the graph as it does not explicitly select features.

$p = 5000$ (the 5000 features with the highest variance), to save computational resources. Figure 1 (a) shows classification test errors from 5-fold CV on this data set. As is clear from the table, PARTIAL MIC-MULTI and BBLASSO are the best methods here. But as was the case with other data sets, BBLASSO puts in more features, which is undesirable in domains (like biology and medicine) where simpler and hence more interpretable model are sought.

The number of features and coefficients selected by all the methods are shown in Figures. 1 (b) and 2 respectively.

5.2 MIC-GROUP

In this section we demonstrate the results of the MIC-GROUP scheme on synthetic and real world data sets. For our experiments we use both the MIC-GROUP (I) and MIC-GROUP-SC (as described in Algorithm 2) methods and compare against BASELINE Feature Selection (which in this case is equivalent to a RIC penalized regression and has a coding scheme similar to Equation 4, Lasso (Tibshirani, 1996), Elastic Nets (Zou and Hastie, 2005) and Group Lasso/ Multiple Kernel Learning (Yuan and Lin, 2006; Jacob et al., 2009; Bach et al., 2004).

For Group Lasso/Multiple Kernel Learning,[15] we used a set of 13 candidate kernels, consisting of 10 Gaussian Kernels (with bandwidths $\sigma = 0.5 - 20$) and 3 polynomial kernels (with degree 1-3) for each feature class as is done by Rakotomamonjy et al. (2008). In the end the kernels which have non zero weights are the ones that correspond to the selected feature classes. Since GL/MKL minimizes a mixed $\ell_1 - \ell_2$ norm so, it zeros out some groups (feature classes). However it is possible to estimate the exact support by thresholding (cross-validated) the estimated weights, as has been done by Zhou (2009) and Lounici (2008), and enforce sparsity within the groups also but as mentioned earlier our main goal is better predictive accuracy and not identifying the correct set of sparse features. The Group Lasso (Yuan and Lin, 2006; Jacob et al., 2009) and Multiple Kernel Learning are equivalent, as has been mentioned in Bach (2008), therefore we used the *SimpleMKL* toolbox (Rakotomamonjy et al., 2008) implementation for our experiments. For Lasso and Elastic Nets we used their standard LARS (Least Angle Regression) implementations (Efron et al., 2004). When running Lasso and Elastic Nets, we pre-screened the data sets and kept only the best $\sim$ 1,000 features (based on their p-values), as otherwise LARS is prohibitively slow. (The authors of the code we used do similar screening, for similar reasons.) For all our experiments on Elastic Nets (Zou and Hastie, 2005) we chose the value of λ_2 (the weight on the ℓ_2 penalty term), as 10^{-6}.

We demonstrate the effectiveness of MIC-GROUP on synthetic data sets and on real data sets pertaining to Word Sense Disambiguation (WSD) (Chen and Palmer, 2005) (ONTONOTES Data Set Hovy et al., 2006) and gene expression data (Mootha et al., 2003).

5.2.1 EVALUATION ON SYNTHETIC DATA SETS

The main hypothesis is that MIC-GROUP methods are beneficial when some groups have multiple predictive features, while others lack them. MIC-GROUP is particularly effective when there are small groups which contain highly predictive features and big groups containing no predictive features.

In order to validate our hypothesis, we test MIC-GROUP on two synthetic data sets. For both the data sets, 1000 features were generated independently from a Normal Distribution $\mathcal{N}(0,1)$, and the response vector of 100 observations Y was computed as the linear combination of a set of 7 beneficial features and Gaussian additive noise $\mathcal{N}(0, 1.7^2)$. The first data set (Set 1) had 4 groups (feature classes) of unequal sizes and 7 beneficial features, all of which lie in a small feature class of size 12. The second synthetic data set (Set 2) was generated so as to reflect the other extreme

15. There is a similar relation between MIC-GROUP and GL/MKL as it is between MIC-MULTI and BBLASSO. Both BBLASSO and GL/MKL are ℓ_1/ℓ_2 penalty based methods and try to solve the same sparsity problem as the corresponding MIC method.

case, in which all the classes are of same size, and had 100 feature classes, each of size 100. Again all 7 beneficial features were in a single feature class.

Method	Avg. Features Selected				10-Fold CV Error	
	Correct		Spurious			
	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2
MIC-GROUP-SC	$\mathbf{6.8 \pm 0.1}$	$\mathbf{5.6 \pm 0.0}$	$\mathbf{0.1 \pm 0.0}$	0.3 ± 0.1	$\mathbf{0.09 \pm 0.02}$	$\mathbf{0.27 \pm 0.01}$
MIC-GROUP (I)	6.7 ± 0.0	5.4 ± 0.1	$\mathbf{0.1 \pm 0.1}$	0.2 ± 0.1	0.11 ± 0.02	0.28 ± 0.02
LASSO	5.2 ± 1.0	4.3 ± 1.2	2.2 ± 1.0	1.8 ± 0.1	0.22 ± 0.03	0.41 ± 0.02
ELASTIC NETS	6.4 ± 0.2	4.9 ± 0.7	3.3 ± 1.1	2.1 ± 1.3	0.20 ± 0.03	0.43 ± 0.02
BASELINE (RIC)	4.4 ± 1.4	3.2 ± 2.2	0.2 ± 0.1	$\mathbf{0.0 \pm 0.0}$	0.27 ± 0.05	0.61 ± 0.04

Table 3: The number of correct and spurious Features Selected and 10 Fold CV Test Errors averaged over 10 runs. Set 1). Unequal class sizes, Set 2). Uniform class sizes.

As can be seen from the results in Table 3, in both cases the MIC-GROUP methods outperform other competing methods.

5.2.2 EVALUATION ON REAL DATA SETS

In order to benchmark the real world performance of our MIC-GROUP, we chose two data sets pertaining to two diverse applications of feature selection methods, namely Computational Linguistics and Gene Expression Analysis. More information regarding the data and the experimental results are given below.

Word Sense Disambiguation (WSD) Data Sets: A WSD data set (ONTONOTES Hovy et al., 2006) consisting of 172 ambiguous verbs and a rich set of contextual features (Chen and Palmer, 2005) was chosen for evaluation. It consists of hundreds of observations of noun-noun collocation, noun-adjective-preposition-verb (syntactic relations in a sentence) and noun-noun combinations (in a sentence or document).

The data set had a total of 172 verbs with $40-45$ feature classes (groups). The number of observations n for the various verbs varied from 100 to 3500 and the number of features p varied from 1000 to 11500.

As with MIC-MULTI we used MIC-GROUP to do feature selection and once we had selected the features we used logistic regression for the final classification problem. The classification test accuracies averaged over all the 172 verbs are shown in Figure 3.

Note that these accuracies are for the binary prediction problem of predicting the most frequent sense. On the entire set of 172 verbs, MIC-GROUP methods are significantly (5 % significance level (Paired t-Test)) better than the competing methods on $160/172$ verbs and have the same accuracy as the best method on 4 occasions. It is also worth noting that MIC-GROUP-SC was ~ 7 times faster than MIC-GROUP (I) as we had hypothesized earlier, as for each selected feature it does a subset search within that feature's group (feature class) to find the optimal number of features to select from the group.

Gene Set Enrichment Analysis (GSEA) Data Sets: The second real data sets that we used for our experiments were gene expression data sets from GSEA (Mootha et al., 2003). There are multiple gene expression data sets and multiple criteria on which the genes can be grouped into classes. For

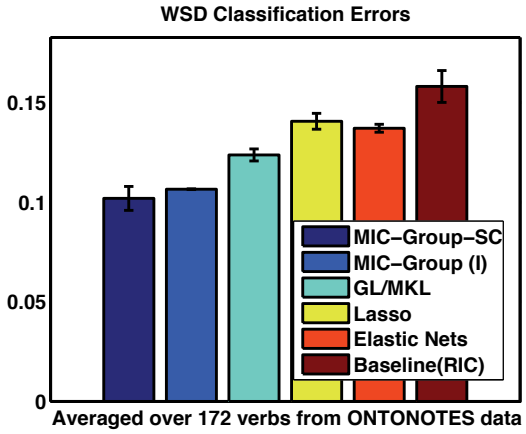


Figure 3: 10 Fold CV classification test accuracies averaged over 172 verbs. *Note:* 1). These are true cross-validation accuracies and no parameters have been tuned on them.

example, different ways of generated gene classes include C1: Positional Gene Sets, C2: Curated Gene Sets, C3: Motif Gene Sets, C4: Computational Gene Sets, C5: GO Gene Sets.

For our experiments, we used gene classes from the C1 and C2 collections. The gene sets in collection C1 consists of genes belonging to the entire human chromosome, divided into each cytogenetic band that has at least one gene. Collection C2 contained gene sets from various sources such as online pathway databases and knowledge of domain experts.

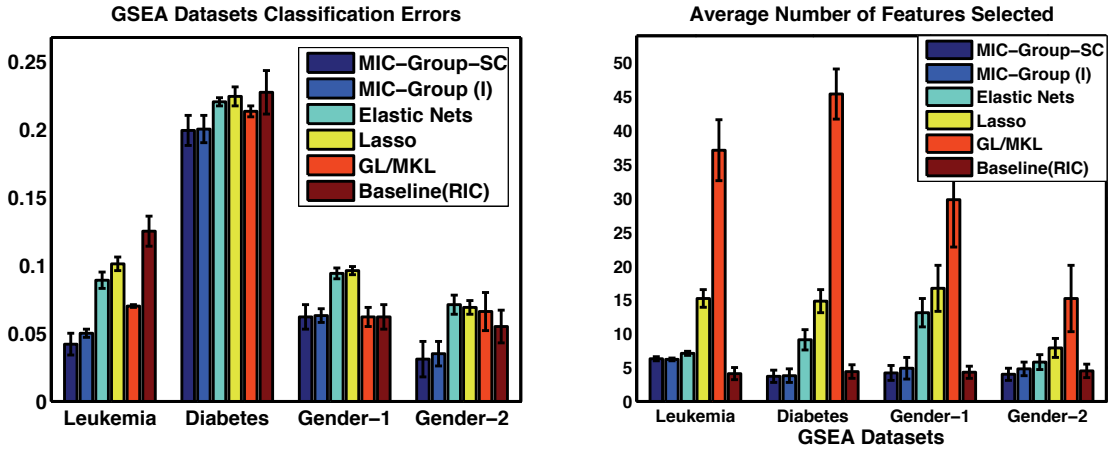
The data sets that we used and their specifications are as shown in Table 4. Though the goal of GSEA is not building classification models but identifying the groups of genes (gene families) which are over-represented when they are filtered by a certain selection procedure; however we are interested in using the transcriptional profiles and the associated group structure for classifying the phenotype, that is, ALL (Acute Lymphoblastic Leukemia) or AML (Acute Myeloid Leukemia) in case of leukemia; DMT (Diabetes Mellitus Type I) or NGT (Normal Glucose Tolerance) in case of diabetes; and determining whether the transcriptional profiles are from a M (Male) or a F (Female) for the gender data sets.

Data Set	# Observations (n)	# Features (p)	# Classes (K)
LEUKEMIA (C1)	48 (24 ALL & 24 AML)	10056	182
GENDER 1 (C1)	32 (17 F & 15 M)	15056	212
DIABETES (C2)	34 (17 NGT & 17 DMT)	15056	318
GENDER 2 (C2)	32 (17 F & 15 M)	15056	318

Table 4: GSEA Data Sets.

The results for these GSEA data sets are as shown in the Figure 4.

For these data sets also MIC-GROUP methods beat the competing methods. Here also MIC-GROUP is significantly (5% significance level, Paired t-test) better than the competing methods. It is interesting to note that MIC-GROUP methods sometimes selected substantially fewer features, but still gave better performance than other methods which goes onto show that adding all or many features from a single group contributes to a redundant signal and efficient feature selection "within" a group (feature class) is warranted.



(a) (b)

Figure 4: 10-fold CV classification test accuracies and the average number of features selected by various methods on the GSEA data sets. *Note:* 1). Remember that we are interested in better predictive accuracy and not in identifying the correct set of sparse features; we can get much sparser models if instead our objective is choosing the correct set of sparse features. 2). These are true cross-validation accuracies and no parameters have been tuned on them.

6. MIC Model Consistency

In this section we show that our MIC methods based on two part MDL and with the model coding costs as described in Section 4 are consistent. By "consistent" we mean that if the data is distributed by one of the probabilistic sources in the set of candidate model classes that our MDL based estimators consider ($\mathcal{M}$), then given enough data, MIC will output the true distribution generating the data. The proof of consistency is similar to the proof of classical two part MDL consistency as given in Barron and Cover (1991) and the recent improvement to that proof by Zhang (2004) by using ideas from KL-complexity. To extend these proofs to the case of MIC, we require the concepts of *Universal Codes*, *KL-Distinguishability*, *Probabilistic Sources* and the *No-Hypercompression Inequality* from information theory. (Refer to Appendix)

We first define some common notation that will be useful throughout this section. Assume we have n data samples (observations) $X_1 \in \mathcal{X}, X_2 \in \mathcal{X}, \ldots, X_n \in \mathcal{X}$ and that they are distributed according to some distribution P_{True}. Further let $P^{(n)}_{MIC}$ be an arbitrary distribution on $\mathcal{X}$ (the distribution estimated by our MIC based model). Also, as shorthand we denote $-\sum_{i=1}^{n} \lg Q(X_i)$ as $-\lg Q(X^n)$ throughout this section and $P^{(n)}$ denotes the marginal distribution on the first n outcomes that is induced by the probabilistic source P.

6.1 Consistency Results

As mentioned earlier, two part MDL has been proved to be consistent in a variety of settings (Barron and Cover, 1991; Zhang, 2004; Grünwald, 2007). Here we provide similar proofs for the case of our MIC based models.

Let $\mathcal{M}$ be a countably infinite set of probabilistic sources and let ℓ_n be some code length function corresponding to a code over elements of $\mathcal{M}$. We assume that the true underlying probabilistic source belongs to the set of sources that our models consider, that is, $P_{True} \in \mathcal{M}$; Zhang (2004) and Barron and Cover (1991) also make this assumption. Also, let P_{MIC} be the probabilistic source corresponding to the two part MDL model selected by MIC and $P^{(n)}_{MIC}$ be the marginal distribution induced by this probabilistic source on the first n outcomes.

Theorem 1 *Let $\mathcal{M}^\delta = \{Q \in \mathcal{M} | KL(P_{True}||Q) \geq \delta\}$. Also, let $\ell_n(P_{True}) < \infty$, then*

$$P_{True}(P^{(n)}_{MIC} \in \mathcal{M}^\delta) \to 0 \text{ as } n \to \infty. \tag{10}$$

The theorem states that the probability that MIC selects a probabilistic source to explain the data that is KL-distinguishable from the true underlying distribution (P_{True}) approaches 0 as the number of observations increase. In other words, with overwhelming probability, P_{True} is KL-indistinguishable from $P^{(n)}_{MIC}$ as n approaches infinity.

Proof

Let $\mathcal{P}_n$ be the distribution corresponding to the code for hypotheses with lengths ℓ_n, such that for all $Q \in \mathcal{M}$, $\mathcal{P}_n(Q) = 2^{-\ell_n(Q)}$. This follows from Kraft's Inequality (Grünwald, 2005). $\mathcal{M}$ is countable, so $\mathcal{M}^\delta$ must also be countable; therefore we can order the elements in $\mathcal{M}^\delta$ according to increasing description length $\ell_n(Q)$ (decreasing $\mathcal{P}_n(Q)$) as $Q_1, Q_2, \ldots$. Fix some θ ($0 < \theta < 1$) and define $\mathcal{M}^\delta_{1:N(\theta)}$ as the subset of $\mathcal{M}^\delta$ consisting of the first N distributions in $\mathcal{M}^\delta$, where N is the smallest number such that $\sum_{j=1}^{N} \mathcal{P}_n(Q_j) \geq \theta$. Now, we define $\mathcal{M}^\delta_{N(\theta)+1:\infty} = \mathcal{M}^\delta \backslash \mathcal{M}^\delta_{1:N(\theta)}$, that is, $\mathcal{M}^\delta_{N(\theta)+1:\infty} = \{Q_{N+1}, Q_{N+2}, \ldots\}$. It can be easily seen that,

$$\mathcal{P}_n(\mathcal{M}^\delta_{N(\theta)+1:\infty}) = \sum_{j=N+1}^{\infty} \mathcal{P}_n(Q_j) \leq 1 - \theta. \tag{11}$$

Now, for any $\mathcal{M}' \subseteq \mathcal{M}^\delta$,

$$\begin{aligned} &P_{True}\left[P^{(n)}_{MIC} \in \mathcal{M}'\right] = \\ &\quad P_{True}\left[\, for\ some\ Z \in \mathcal{M}' : \ell_n(P_{True}) + \ell_n(X^n|P_{True}) \geq \ell_n(Z) + \ell_n(X^n|Z)\right] \\ &\quad \leq \sum_{Z \in \mathcal{M}'} P_{True}\left[\ell_n(P_{True}) + \ell_n(X^n|P_{True}) \geq \ell_n(Z) + \ell_n(X^n|Z)\right]. \end{aligned} \tag{12}$$

The above inequality is obtained by applying the Union Bound.

Now, by re-arranging Equation 12 and noting that the error term in two-part coding can be replaced by a log term as in Equation 2,

$$g_n(Z) = P_{True}\left[-\lg(P_{True}(X^n)) \geq -\lg(Z(X^n)) + \ell_n(Z) - \ell_n(P_{True})\right]. \tag{13}$$

The $\ell_n(Z)$ term corresponds to the number of bits required to code the model and the $-\lg Z(X^n)$ term corresponds to the data likelihood term in the two part MDL coding scheme.

From Equations 12, 13 and 10, it follows that:

$$\begin{aligned} P_{True}\left[P^{(n)}_{MIC} \in \mathcal{M}^\delta\right] &= P_{True}\left[P^{(n)}_{MIC} \in \mathcal{M}^\delta_{1:N(\theta)}\right] + P_{True}\left[P^{(n)}_{MIC} \in \mathcal{M}^\delta_{N(\theta)+1:\infty}\right] \\ &\leq \sum_{Z \in \mathcal{M}^\delta_{1:N(\theta)}} g_n(Z) + \sum_{Z \in \mathcal{M}^\delta_{N(\theta)+1:\infty}} g_n(Z). \end{aligned} \tag{14}$$

Since, each $Z \in \mathcal{M}^{\delta}_{1:N(\theta)}$ is KL-Distinguishable from P_{True} and we have $g_n(Z) \leq e^{-n\delta}$ (Stein's Lemma Cover and Thomas, 2006) for all $Z \in \mathcal{M}^{\delta}$. Hence $\sum_{Z \in \mathcal{M}^{\delta}_{1:N(\theta)}} g_n(Z)$ is a sum of a finite number of Z's, each of which is exponentially small, so we have $\lim_{n\to\infty} \sum_{Z \in \mathcal{M}^{\delta}_{1:N(\theta)}} g_n(Z) = 0$.

Now, we consider the second term in Equation 14. First assume that there exists $N'(\theta) \geq N(\theta) + 1$ such that the sources $\mathcal{M}^{\delta}_{N(\theta)+1:N'(\theta)}$ have description lengths (ℓ_n) smaller than the description length of P_{True} and the sources $\mathcal{M}^{\delta}_{N'(\theta)+1:\infty}$ have description lengths greater than $\ell_n(P_{True})$. Also, we have $\ell_n(P_{True}) = O(\log n)$, given all the coding schemes we have designed using MIC models. Now, by Kraft's Inequality it follows that

$$\begin{aligned} & \sum_{Z' \in \mathcal{M}^{\delta}_{N(\theta)+1:N'(\theta)}} 2^{-\ell_n(Z')} \leq 1 \\ \Longrightarrow & \sum_{Z' \in \mathcal{M}^{\delta}_{N(\theta)+1:N'(\theta)}} 2^{-k \log n} \leq 1 \\ \Longrightarrow & \sum_{Z' \in \mathcal{M}^{\delta}_{N(\theta)+1:N'(\theta)}} n^{-k} \leq 1. \end{aligned}$$

Therefore, there are at most a polynomial number n^k of elements in $\mathcal{M}^{\delta}_{N(\theta)+1:\infty}$ with shorter description lengths than P_{True}. Hence, as earlier $\sum_{Z' \in \mathcal{M}^{\delta}_{N(\theta)+1:N'(\theta)}} g_n(Z')$ is a sum of a finite number of (Z)s, each of which is exponentially small, so we have $\lim_{n\to\infty} \sum_{Z' \in \mathcal{M}^{\delta}_{N(\theta)+1:N'(\theta)}} g_n(Z') \; = \; 0$.

Now, we bound the remaining terms by applying the no-hypercompression inequality to each term in $\sum_{Z \in \mathcal{M}^{\delta}_{N'(\theta)+1:\infty}} g_n(Z)$ with $K = \ell_n(Z) - \ell_n(P_{True})$,

$$g_n(Z) = P_{True}\left[-\lg(P_{True}(X^n)) \geq -\lg(Z(X^n)) + \ell_n(Z) - \ell_n(P_{True})\right] \leq 2^{-\ell_n(Z)+\ell_n(P_{True})}. \qquad (15)$$

From Equations 11 and 15, we get

$$\sum_{Z \in \mathcal{M}^{\delta}_{N'(\theta)+1:\infty}} g_n(Z) \leq \sum_{Z \in \mathcal{M}^{\delta}_{N'(\theta)+1:\infty}} 2^{-\ell_n(Z)+\ell_n(P_{True})} \leq (1-\theta) \cdot 2^{\ell_n(P_{True})}.$$

The above holds for every $0 < \theta < 1$, so for every $\varepsilon > 0$ we can choose $\theta = 1 - \varepsilon \cdot 2^{-\ell_n(P_{True})}$ giving $\sum_{Z \in \mathcal{M}^{\delta}_{N'(\theta)+1:\infty}} g_n(Z) \leq \varepsilon$ for all large n. Combining this with Equation 14 we find that for all $\varepsilon > 0$

$$\lim_{n\to\infty} P_{True}[P^{(n)}_{MIC} \in \mathcal{M}^{\delta}] < \varepsilon.$$

■

A corollary of the above theorem is that the MIC coding schemes as described in Section 4 are not an arbitrary procedure. There can be many valid codes, but we can not tweak MDL by using arbitrary codes to give the answers that we would like it to give.

Another important theoretical property that is attractive for sparse learning algorithms is *sparsistency*, which is shorthand for "sparsity pattern consistency". In other words:

$$P\left[supp(\mathbf{w}^{True}) = supp(\mathbf{w}_n^{MIC})\right] \rightarrow 1 \; as \; n \rightarrow \infty,$$

where $supp(\mathbf{w}) = \{\mathbf{w} : \mathbf{w}_j \neq 0\}$, $\mathbf{w}^{True}$ is the true sparse weight vector and $\mathbf{w}^{MIC}$ is the weight vector estimated by MIC based methods. Sparsistency implies that the learning algorithm is consistently able to identify the correct set of sparse features in the asymptotic limit.

Lasso and Group Lasso have been proved to be *sparsistent* under irrepresentable conditions that depend on the sign of the true weight vector ($\mathbf{w}^{True}$) (Zhao and Yu, 2006; Wainwright, 2009; Meinshausen and Bühlmann, 2006; Bach, 2008). Tropp (2004) proved that forward greedy feature selection also selects features consistently when the linear model has a zero-mean stochastic noise; Zhang (2009b) improved this result to include non-zero mean sub-Gaussian stochastic noise. However, due to the complexity of the forward greedy feature selection the sparsistency condition in this case depends only on the feature (design) matrix $\mathbf{X}$, unlike Lasso and Group Lasso.

Since our MIC based methods are based on forward greedy feature selection, that is, they use the MDL principle to provide a cost function which is greedily minimized by a forward search, they should be *sparsistent*. However, for ℓ_0 penalized regression, the *sparsistency* condition also depends on the information theoretic penalty in that the penalty must increase with n (the number of observations) (Wu and Zhou, 2010). For our MIC based methods this penalty is a combination of RIC, AIC (to code the coefficients) and other coding schemes which incorporate the structure of the problem at hand. The penalties for the MIC based methods as presented in this paper do not have the required dependence on n, so they are not sparsistent. However, we could modify our coding schemes slightly by using the BIC penalty ($\lg n$ bits) to code the coefficients instead of AIC to ensure sparsistency of MIC. However, we prefer that our methods are not sparsistent as in that case we achieve competitive performance with the true underlying model, that is, we get finite *risk-inflation* of about $2 \lg p$ (Foster and George, 1994) whereas if we chose sparsistency then MIC would have infinite risk-inflation. Thus, given the choice between *better model-fit* and sparsistency, we chose the former. However, if sparsistency is more important than predictive accuracy, making a small change in the coding schemes would guarantee it.

7. A Model for "Intra Domain" Adaptation: TRANSFEAT

In the previous sections we proposed MIC based methods for the related problems of simultaneous feature selection for a set of multiple related tasks (MIC-MULTI) and grouped feature selection for single task (MIC-GROUP). The focus of those methods was joint feature selection, but in many applications it is the case that some of the tasks have less data available than other tasks and building supervised learning models from the limited amount of data does not give high predictive accuracies. So, it becomes desirable to "borrow strength" for the tasks with less amount of data from the tasks with lots of data. In other words, we want to have "intra domain" adaptation or Transfer Learning (Ando and Zhang, 2005; Raina et al., 2006).

In this section, we propose a method called TRANSFEAT which addresses the above problem by transferring information between similar tasks by using a feature relevance prior. We demonstrate the effectiveness of TRANSFEAT for the problem of Word Sense Disambiguation (WSD), and show that in this domain TRANSFEAT significantly improves accuracy on tasks with less data. TRANSFEAT, could, of course, be applied to wide variety of domains, but is particularly useful for WSD

as state-of-the-art WSD systems, including the ones that use feature selection, are strongly limited by the paucity of labeled data. For example, the training set of the SENSEVAL-2 English lexical sample task has only ~ 10 labeled examples per sense (Florian and Yarowsky, 2002). Such limited data makes it difficult to build high accuracy models using standard supervised learning techniques and suggests the use of transfer learning to improve performance.

As mentioned above, TRANSFEAT learns a feature relevance prior from "similar" tasks, and gives supervised learning accuracies which are comparable to or better than state-of-the-art WSD systems. Learning this prior for feature relevance of a test task makes those features that have been selected in the models of other "similar" tasks become more likely to be selected. TRANSFEAT does this by using a MDL-based approach similar to the MIC methods presented above.

Task Setting: We are given a set of target words each having an $n \times p$ feature matrix ($\mathbf{X}_{n\times p}$), where n is the total number of observations (instances) and p is the total number of features. We have a $n \times h$ response matrix ($\mathbf{Y}_{n\times h}$) of the h sense labels for each of the n observations. The WSD task is to assign a sense to each test instance. Note that this is a multi-class problem; we have a single task, which is to predict the correct sense of the word and we have h possible choices (the word senses) for that task. So, we approach it differently from the multi-task problem (MIC-MULTI), where we predicted all tasks jointly.

Overview of TRANSFEAT*:* TRANSFEAT builds upon MIC-GROUP and it has several steps:

- Break the $\mathbf{Y}_{n\times h}$ matrix into h, $n \times 1$ matrices, that is, out of one multiclass (h classes) problem we make h binary class problems. The prediction problem now becomes "Is this word sense 1 or not?", etc. The main reason for doing this is that not all senses of all words are similar to all senses of some other word. Thus, transfer learning only makes sense at level of individual word *senses* rather than at the level of whole words.

- Make separate feature matrices for these h prediction problems, because the original feature matrix $\mathbf{X}_{n\times p}$ contained features which would be useful for the multiclass problem of "What is the exact sense of the word?", rather than for the binary problems of "Is this sense 1 or not?" and so on. We do this is by characterizing each binary problem by those features from the original p features which are positively correlated with that particular word sense.[16] This gives h feature matrices $\mathbf{X}_{\{i=1,\ldots,h\}}$ drawn from the original $n \times p$ feature matrix, where each of these matrices need not have the same number of features.

- Next, cluster the different word senses by using "foreground-background" clustering that puts all singleton points into a "background cluster" which we then ignore

- Learn separate MIC-GROUP-SC models for each word sense. (Remember that as mentioned in the Section about MIC-GROUP, WSD is one problem which exhibits group structure and therefore we use it as a base model on which we build TRANSFEAT.)

- For each word sense in a cluster, use TRANSFEAT to learn a feature relevance prior from the remaining word senses in that cluster that have more observations than the target word sense, on the features of that word sense. As we explain later, this feature relevance prior allows us to learn better MIC-GROUP-SC models by relaxing the *uniform prior* assumption that each

16. In general, features with positive coefficients are associated with the given sense and those with negative coefficients with other senses of that word.

group (feature class) and then each feature within that group is equally likely to be selected, that MIC-GROUP-SC makes.

- Given these better MIC-GROUP models for all the word senses, we solve the actual h class WSD problem by choosing the sense whose model gave the highest score as the most likely sense for that word.

We learn the feature relevance prior only from distributionally similar word *senses*; in contrast to Ando (2006) who share knowledge across "all" the senses of "all" the words. Our approach makes sense as it is difficult to find words which are similar in all their senses; however, one can often find words which have one or a few similar senses. For example, one sense of "fire" (as in "fire someone") should share features with one sense of "dismiss" (as in "dismiss someone"), but other senses of "fire" (as in "fire the gun") do not. Similarly, other meanings of "dismiss" (as in "dismiss an idea") should not share features with "fire". Similarly, the words "kill", "capture" and "arrest," share one similar sense. This justifies our choice of breaking down the problem down to the level of individual word senses.

Thus, knowledge can only be fruitfully transferred between the shared senses of different words, even though the models being learned are for disambiguating different senses of a single word. To address this problem, we cluster similar word senses of different words, and then use the models learned for all the word senses in the cluster with more data (observations) than the held out word sense (called "training word senses") to put a feature relevance prior on what features will be more predictive for the held out test word sense. We hold out each word sense in the cluster once and learn a prior from the remaining word senses in that cluster. For example, we can use the models for discriminating the senses of the words "kill" and the senses of "capture", to put a prior on what features should be included in a model to disambiguate senses of the distributionally similar word "arrest", which has considerably less data than the other two words (ONTONOTES data set), hence enabling us to learn high accuracy models for "arrest". If at least one sense of the word "arrest", that we are trying to model is similar to the other word senses (for "kill" and "capture"), some of the same features should be beneficial for all of them.

7.1 TRANSFEAT Formulation

We now describe TRANSFEAT in detail and show how it can be used to learn better feature selection models by relaxing the overly simplistic assumption of the model coding schemes of MIC methods of *uniform prior* by learning a feature relevance prior.

We define a binary random variable $f_i \in \{1,0\}$ that denotes the event of the i^{th} feature being in or not being in the model for the test word sense, and model it as being from a Bernoulli distribution parameterized by θ_i:

$$p(f_i|\theta_i) = \theta_i^{f_i}(1-\theta_i)^{1-f_i}. \tag{16}$$

Given the data for the i^{th} feature for all the training word senses, we can write: $\mathcal{D}_{f_i} = \{f_{i1}, ..., f_{iv}, ..., f_{it}\}$. The model likelihood (under the i.i.d assumption) can be written as:

$$p(\mathcal{D}_{f_i}|\theta_i) = \prod_{v=1}^{t} p(f_{iv}|\theta_i) = \prod_{v=1}^{t} \theta^{f_{iv}}(1-\theta_i)^{1-f_{iv}},$$

and the posteriors can be calculated by putting a prior over the parameters θ_i as:

$$p(\theta_i|\mathcal{D}_{f_i}) = p(\mathcal{D}_{f_i}|\theta_i) \times p(\theta_i|a,b),$$

where a and b are the hyperparameters of the Beta Prior (the conjugate of the Bernoulli distribution). The predictive distribution of θ_i is:

$$p(f_i = 1|\mathcal{D}_{f_i}) = \int_0^1 p(f_i = 1|\theta_i) p(\theta_i|\mathcal{D}_{f_i}) d\theta_i.$$

Substituting from 16 in the above equation we get:

$$p(f_i = 1|\mathcal{D}_{f_i}) = \int_0^1 \theta_i p(\theta_i|\mathcal{D}_{f_i}) d\theta_i = \mathbb{E}[\theta_i|\mathcal{D}_{f_i}].$$

Using the standard results for the mean and the posterior of a Beta distribution we obtain:

$$p(f_i = 1|\mathcal{D}_{f_i}) = \frac{k+a}{k+l+a+b}, \tag{17}$$

where k is the number of times that the i^{th} feature is selected and l is the complement of k, that is, the number of times the i^{th} feature is not selected in the training data.

As can be seen from Equation 17, the probability that a feature is selected for the held out test word sense is a "smoothed" average of the number of times it was selected in the models for the senses of other words that are similar to it.

Using similar reasoning, we can extend the above concept to the groups (feature classes) so that the probability that a group (feature class) is selected is also a "smoothed" average of the number of times it was selected in the models for the senses of other words that are similar to it.

In light of the above reasoning, the modified model cost for MIC-GROUP for coding the i^{th} feature when previously no features have been selected from the j^{th} feature class which contains that feature can be written as follows:

$$S_M^i = -\lg p(G_j = 1|\mathcal{D}_{G_j}) - \lg p(f_i = 1|\mathcal{D}_{f_i}) + 2,$$

and for the case when some features have already been selected from the j^{th} feature class, we can write a modified coding cost as follows:

$$S_M^i = \min\left[-\lg p(G_j = 1|\mathcal{D}_{G_j}), 1 + \lg(Q)\right] - \lg p(f_i = 1|\mathcal{D}_{f_i}) + 2,$$

where the first term represents the probability of selecting at least one feature from the j^{th} feature class, the second term represents the probability of selecting the i^{th} feature, and the third term which is used to code the coefficient values remains the same as earlier.[17] Note that in the case when we have previously selected features from a given feature class, the most efficient way to code the feature class is to use the minimum of the TRANSFEAT cost and the actual "switch" coding cost as described in Section 4.2. Thus TRANSFEAT replaces the implicit uniform prior of MIC-GROUP with a coding scheme which is more informed by the prior learned from similar tasks.

The detailed algorithm for TRANSFEAT is given in Algorithm 3.

17. The negative sign is due to the duality between Bayesian and Information Theoretic interpretation as mentioned earlier.

Algorithm 3 TRANSFEAT

Break the multiclass problem into h binary prediction problems.
Make the feature matrices for each of these problems, that is, $\mathbf{X}_{\{i=1,\ldots,h\}}$.
Cluster the different word senses by "foreground-background" clustering.
$total_clusters = \{1,\ldots,c\}$
$word_senses_k = s_k$ // Number of word senses in k^{th} cluster.
for i in $total_clusters$ **do**
  for t in $word_senses_i$ **do**
    Learn separate MIC-GROUP-SC models for all the word senses. // Uniform prior assumption
  end for
end for
for i in $total_clusters$ **do**
  for t in $word_senses_i$ **do**
    Learn TRANSFEAT model on all word senses in the cluster which have more data (observations) than the t^{th} word sense.
    Use the revised model costs S_M output by TRANSFEAT to learn better MIC-GROUP -SC model for t^{th} word sense.
    // The uniform prior assumption of MIC-GROUP-SC has been relaxed.
  end for
end for
Disambiguate the word as a whole by choosing the correct sense (from h possible senses) as the one whose model gave the highest score.

7.1.1 CHOICE OF HYPERPARAMETERS

The hyperparameters a and b in Equation 17 control the "smoothing" of our probability estimates, that is, how strongly we want the evidence obtained from similar word senses to affect the model that we learn for the test word sense.

In all our experiments we set $a = 1$ and choose b so that in the limiting case of no transfer, that is, ($k = l = 0$ in Equation 17) the coding scheme will reduce to the baseline feature selection described in (Equation 4). Thus, we choose $b = p - 1$ where p is the total number of features/feature classes (depending on what we are coding) in the test word sense.

7.2 Experimental Results

In this section we first describe our data and similarity metric that we used; we then report the results of applying TRANSFEAT to the SENSEVAL-2 and ONTONOTES data sets.

7.2.1 SIMILARITY METRIC

Finding a good similarity metric between different word senses is perhaps one of the biggest challenges that we faced. It is also the part of this section that is specific to the problem of word sense disambiguation. There are many ways in which word senses can be judged as similar, including having similar "meanings" or similar syntactic usages. Human annotated lexicons such as Levin classes (Levin, 1993), hypernyms or synonyms according to WORDNET (Miller, 1990; Lin, 1999),

or VERBNET classes (Kipper et al., 2000; Schuler, 2006) capture different aspects of this similarity, as does INFOMAP(`http://infomap.stanford.edu`) (Raina et al., 2006), which gives distributional similarity score for words in the corpus. We choose instead to define a similarity metric based, as described below on combinations of many different aspects of the lexical and syntactic context of the word.

One might think of doing K-means clustering of the word senses based on their features, but this works poorly, as it assigns all the word senses to some cluster, while in reality, there are in practice many word senses that are not sufficiently similar to any other word sense, either semantically or syntactically and hence many word senses occur in "singleton" clusters. K-means and perhaps surprisingly, hierarchical agglomerative clustering, even after extensive use of different 'K' or thresholds, failed to give reasonable clusters.

We thus need a clustering method that gives tight clusters of word senses, and does not attempt to cluster those word senses which are not similar to any other word sense in the corpus. We do this using a "foreground-background" clustering algorithm as proposed by Kandylas et al. (2007). This algorithm gives highly cohesive clusters of word senses (the *foreground*) and puts all the remaining word senses in the *background*. The parameters that it takes as input are the % of data points to put in *background* (i.e., what would be the singleton clusters) and a similarity threshold which impacts the number of *foreground* clusters. We experimented with putting 20% and 33% data points in background and adjusted the similarity threshold to give us $50-100$ *foreground* clusters. The results reported below have 20% background and $50-100$ *foreground* clusters.

7.2.2 DESCRIPTION OF DATA

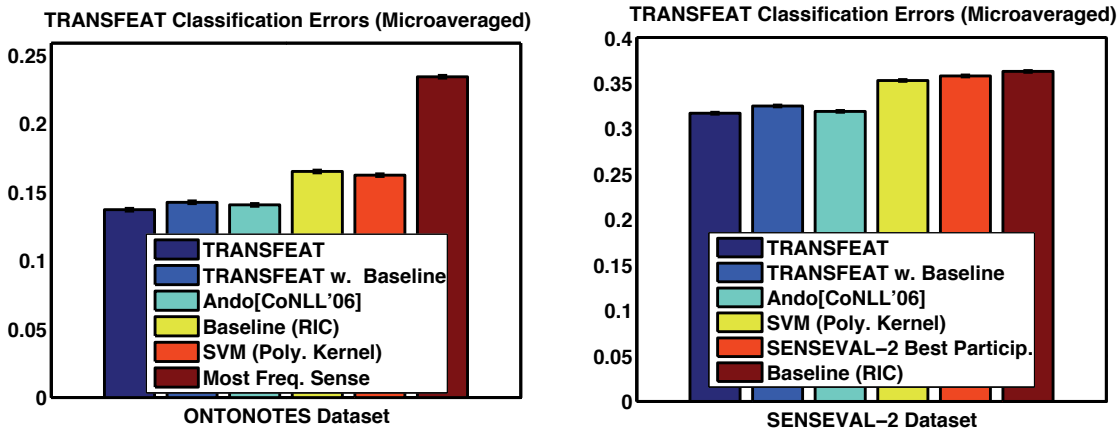
We used the SENSEVAL-2 English lexical sample data, which contains a total of 73 different words (29 nouns, 29 verbs, and 15 adjectives) and the ONTONOTES verb data (the same one used for experiments of MIC-GROUP), containing 172 verbs. The main difference between these two data sets is that SENSEVAL-2 data contains "fine grained" senses of the words and as a result tends to have more senses per word than the "coarse grained" verb senses in ONTONOTES. (See Table 5.)

Data Set	#words	#train	avg #senses per word
SENSEVAL-2 (nouns+verbs+adj.)	73	8611	10.7
ONTONOTES (only verbs)	172	See Note (*in caption*)	3.7

Table 5: Data Statistics of SENSEVAL-2 and ONTONOTES data sets. *Note:* In our experiments we used the standard test-train splits for SENSEVAL-2; ONTONOTES data does not have any standard splits so we report 10-Fold cross validation test-accuracies

7.2.3 RESULTS

We cluster the word senses based on all the features, that is, semantic+syntactic similarity features. We experimented clustering using only syntactic and only semantic features but we got the best results using the combined feature set.



(a) (b)

Figure 5: 10-fold CV (micro-averaged) test accuracies of various methods for ONTONOTES and SENSEVAL-2 (English Lexical Sample) data sets. *Note:* 1.) These are true cross-validation accuracies and no parameters have been tuned on them. 2.) The final accuracies reported are averaged over the entire 172 verbs. 3). We used the standard test-train splits for SENSEVAL-2 as mentioned on the data website and as used in previous studies

All results reported are micro-averaged[18] accuracies. In order to ensure fairness of comparison we compute the predicted sense for each observation by selecting the word sense model (from among the different senses for that word) with the highest score for that observation sense. As in earlier experiments we use TRANSFEAT only to select features and later we use logistic regression for classification. This "one vs all" approach to prediction in multi-class problems is widely used, although higher accuracy can sometimes be obtained by more complex pairwise comparison methods.

We use two versions of TRANSFEAT, as can be seen in Figure 5. The first version is exactly the same as mentioned in Algorithm 3, while the second version, TRANSFEAT w. Baseline, builds upon baseline feature selection (Equation 4) instead of MIC-GROUP-SC. We compare TRANSFEAT methods against baseline feature selection (Equation 4), SVM with a polynomial kernel, Ando[CoNLL'06] (Ando, 2006), computed with the standard implementation of the algorithm from the Berkeley Transfer Learning Toolkit (Rakhlin, 2007), and a simple most frequent sense baseline. For SVM we used the standard libSVM package (Chang and Lin, 2001). We used a polynomial kernel, as it gave better performance on held out data than other kernels including linear and RBF. We tuned the cost parameter 'c' and the degree of polynomial 'd' parameters of the polynomial kernel using a separate cross validation.

18. Our precision and recall are always the same as we assign exactly one sense to each instance. Hence the accuracy that we report is the same as the F-measure or 'micro-averaged' recall as is reported in many WSD studies.

7.2.4 ANALYSIS OF ONTONOTES RESULTS

The results for the different settings for the ONTONOTES data set averaged over the entire 172 verbs are shown in Figure 5 (a). The TRANSFEAT models are significantly better (5% significance level using a paired t-test) than all the competing methods except Ando[CoNLL'06].

Some examples will help to emphasize the point that we made earlier that transfer helps the most in cases in which the target word sense has much less data than the word senses from which knowledge is being transferred. "kill" had roughly 6 times more data than all other word senses in its cluster (i.e., "arrest", "capture", "strengthen", etc.) In this case, TRANSFEAT gave $3.2-8.7\%$ higher accuracies than competing methods on these three words. Both versions of TRANSFEAT do much better than Ando[CoNLL'06] on these select words even though on average over all 172 verbs the difference is slender. Similarly, for the case of word "do" which had roughly 10 times more data than the other word senses in its cluster (e.g.,"die" and "save"), TRANSFEAT gave $4.1-6.2\%$ higher accuracies than other methods. Transfer makes the biggest difference when the target words have much less data than the word senses they are generalizing from, but even in cases where the words have comparable amounts of data we still get a $1.5-2.5\%$ increase in accuracy.

However, as one might expect, transfer learning can sometimes hurt performance; there can be so-called "negative-transfer" (Caruana, 1997). This was the case for 8 verbs out of the 172.

7.2.5 ANALYSIS OF SENSEVAL-2 RESULTS

The results for SENSEVAL-2 data set are shown in Figure 5(b). Here also TRANSFEAT does significantly better (5% significance level using a paired t-test) than the baseline feature selection method and most of the other state-of-the-art algorithms. It is worth noting that a high degree of engineering goes into the state-of-the-art SENSEVAL-2 systems. This is in contrast to TRANSFEAT, which uses information theoretic feature selection and thus has no free parameters to tune. The TRANSFEAT results are comparable to those reported in Ando (2006), which is the state-of-the-art system on SENSEVAL-2. Since Ando (2006), only mentions the overall accuracy and not the accuracy on individual words, we cannot tell whether this slender difference is statistically significant.

For words that had considerably fewer observations than other words in their cluster, TRANSFEAT again gave major benefits. For example, "begin" had $\sim$ 8 times more data (on average per sense) than the other word senses in its cluster (i.e., "work" and "develop"). In this case, TRANSFEAT gave $6.1-7.1\%$ improvement in accuracy over the baseline feature selection. Similarly, "leave" had $\sim$ 2 times more data than "turn" and "strike", and in this case TRANSFEAT gave $5.1-6.2\%$ improvement in accuracy over the baseline. These improvements are considerably larger than the average improvement over all the words as reported in Figure 5(b).

For this data set there was negative transfer on 5 out of 73 words.

8. Conclusion

In this paper we presented a framework for learning sparse models based on the information theoretic Minimum Description Length (MDL) principle. We presented two models based on the MIC (Multiple Inclusion Criterion) which greedily select features using the MDL principle in the single and multi task settings respectively. Both the methods, MIC-MULTI and MIC-GROUP, induce two level sparsity; MIC-GROUP does feature selection at the level of groups and also at the level of features within each group and MIC-MULTI allows each selected feature to be added to the models

of some or all of the tasks. We showed how we can use MDL to specify *customized* coding schemes in scenarios where the problem has complex structure. We also discussed the conditions under which the MDL based methods are consistent and *sparsistent* and also showed that the MDL coding schemes are not arbitrary and have a corresponding Bayesian interpretation. Lastly, we proposed a model, TRANSFEAT which can be used to transfer a feature relevance prior to tasks which have less data available. We evaluated all three methods on a variety of domains including genomics (for both yeast and beast cancer) and natural language processing (Word Sense Disambiguation). Our methods are consistently at least as accurate as state-of-the-art methods, while producing models that are more sparse. Such sparseness is particularly important for applications such as genomics and computational linguistics, where interpretable models are valued.

Appendix A.

No Hypercompression Inequality:

$$\forall K > 0, P_{True}[-\lg P_{True}(X^n) \geq -\lg P_{Model}(X^n) + K] \leq 2^{-K}.$$

This inequality states that the probability of a code compressing the data by more than K bits, than the code corresponding to P_{True} is exponentially small in K, where K is any positive number.

The proof follows by using Markov's inequality and can be found in Grünwald (2007).

KL-Distinguishability: If the actual data was generated by the distribution P_{True} then the distribution P_{Model} is said to be *KL distinguishable* from P_{True} if its relative entropy (KL Divergence) from P_{Model} is greater than δ. In other words P_{Model} lies outside a δ ball of P_{True} in a relative entropy sense. More formally,

$$\forall \delta > 0, KL(P_{True}||P_{Model}) \geq \delta.$$

Universal Coding Schemes: Going back to the standard MDL setting which envisions a Sender and Receiver, assume that the Sender and Receiver have a set of candidate coding schemes $\mathcal{L}$ for $\mathcal{X}^n$ available. Both of them know that one of these available codes will give the highest compression for the sequence $X^n \in \mathcal{X}^n$. In other words:

$$L_{Optimal}(X^n) = \operatorname*{argmin}_i \{L_i(X^n)\} \; \forall L_i \in \mathcal{L}.$$

However, they must decide on a code before the sender observes the actual data X^n and they do not know which is the best code. In the Bayesian terminology, this problem is similar to finding the classifier which has the Optimal Bayes Risk—that is, the classifier with the minimum possible risk among all the candidate classifiers. One thing that the Sender can do is on seeing the data (X^n), he encodes the data using $L_{Optimal}$ as described above. However, this is not feasible as the Receiver does not know what code the Sender used and so he would not be able to decode the message. Therefore it is not possible to find the best code that compresses the data and so in practice people use *universal codes* which compress the data almost as well as $L_{Optimal}$. This corresponds to a classifier whose risk is close to Bayes Risk. It has been shown that the two part MDL codes that we used in this paper to describe MIC based methods are universal codes (Grünwald, 2005; Grünwald, 2007; Rissanen, 1999). Moreover the "uniform prior" code and the "combinatorial code" we used in coding the model for MIC-MULTI and MIC-GROUP are also universal codes and minimax optimal (Grünwald, 2007).

Probabilistic Sources: Consider a sample space $\mathcal{X}$ and let $\mathcal{X}^+ := \bigcup_{n\geq 1} \mathcal{X}^n$ denote the set of all possible samples of each length. Also, define $\mathcal{X}^0 = \{x^0\}$ where x^0 is a special sequence which is called empty sample. Finally $\mathcal{X}^* = \mathcal{X}^+ \bigcup \mathcal{X}^0$. Now, a probabilistic source with outcomes in $\mathcal{X}$ is a function $P : \mathcal{X}^* \longrightarrow [0,\infty)$ such that for all $n \geq 0$, all $x^n \in \mathcal{X}^n$ we have:

- $\sum_{z\in\mathcal{X}} P(x^n, z) = P(x^n)$ (compatibility condition)
- $P(x^0) = 1$

The two conditions say that the "event" that data (x^n, z) arrives is identical to the event that x^n arrives first and data z arrives afterward. Intuitively, probabilistic sources can be thought of as probability distributions over infinite sequences, but defining them as probability distributions over $\mathcal{X}^\infty$ requires measure theory and the interested user can find the details in any advanced probability book.

References

H. Akaike. Information theory and the extension of the maximum likelihood principle. In *2nd International Symposium on Information Theory, Budapest*, pages 261–281, 1973.

R. Ando. Applying alternating structure optimization to word sense disambiguation. In *(CoNLL-X)*, 2006.

R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008. ISSN 0885-6125.

F. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *ICML*, 2004.

A. R. Barron and T. M. Cover. Minimum complexity density estimation. *IEEE Transactions on Information Theory*, 37(4):1034–1054, 1991.

A. R. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–, 1998.

P. Bickel and K. Doksum. *Mathematical Statistics*. Prentice Hall, 2001.

R. Caruana. Multitask learning. In *Machine Learning*, pages 41–75, 1997.

C. C Chang and C.J. Lin. *LIBSVM: a library for support vector machines*, 2001. URL `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

J. Chen and M. Palmer. Towards robust high performance word sense disambiguation of english verbs using rich linguistic features. In *IJCNLP*, pages 933–944, 2005.

J. Chen, A. I. Schein, L. H. Ungar, and M. Palmer. An empirical study of the behavior of active learning for word sense disambiguation. In *HLT-NAACL*, 2006.

T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 2006.

P. S. Dhillon and L. H. Ungar. Transfer Learning, Feature Selection and Word Sense Disambiguation. In *Annual Meeting of the Association of Computational Linguistics, (ACL)*, August 2009.

P. S. Dhillon, D. P. Foster, and L. H. Ungar. Efficient Feature Selection in the Presence of Multiple Feature Classes. In *International Conference on Data Mining (ICDM)*, pages 779–784, 2008.

P. S. Dhillon, B. Tomasik, D. P. Foster, and L. Ungar. Multi-Task Feature Selection Using The Multiple Inclusion Criterion (MIC). In *European Conference on Machine Learning (ECML)-PKDD*, Lecture Notes in Computer Science. Springer, September 2009.

P. S. Dhillon, D. P. Foster, and L. Ungar. Feature selection using multiple streams. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 13, 2010.

B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

R. Florian and D. Yarowsky. Modeling consensus: classifier combination for word sense disambiguation. In *EMNLP '02*, pages 25–32, 2002.

D. P. Foster and E. I. George. The risk inflation criterion for multiple regression. *The Annals of Statistics*, 22(4):1947–1975, 1994. ISSN 00905364.

P. D. Grünwald. A tutorial introduction to the minimum description length principle. In *Advances in Minimum Description Length: Theory and Applications*. MIT Press, 2005.

P. D. Grünwald. *The Minimum Description Length Principle (Adaptive Computation and Machine Learning)*. The MIT Press, 2007. ISBN 0262072815.

E. H. Hovy, M. P. Marcus, M. Palmer, L. A. Ramshaw, and R. M. Weischedel. Ontonotes: The 90% solution. In *HLT-NAACL*, 2006.

J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. In *ICML '09*, 2009.

L. Jacob, G. Obozinski, and J-P. Vert. Group lasso with overlap and graph lasso. In *ICML '09*, 2009.

T. Jebara. Multi-task feature and kernel selection for SVMs. In *Proceedings of the Twenty-first International Conference on Machine Learning*. ACM New York, NY, USA, 2004.

V. Kandylas, S. P. Upham, and L. H. Ungar. Finding cohesive clusters for analyzing knowledge communities. In *ICDM*, pages 203–212, 2007.

K. Kipper, H. T. Dang, and M. Palmer. Class-based construction of a verb lexicon. In *AAAI/IAAI*, pages 691–696, 2000.

S.-I. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *ICML '07*, pages 489–496, 2007. ISBN 978-1-59593-793-3.

B. Levin. *English Verb Classes and Alternations*. University of Chicago Press, 1993.

D. Lin. Review of WordNet: an electronic lexical database by Christiane Fellbaum. The MIT Press 1998. *Comput. Linguist.*, 25(2):292–296, 1999. ISSN 0891-2017.

D. Lin, E. Pitler, D. P. Foster, and L. H. Ungar. In defense of ℓ_0. In *Workshop on Feature Selection,(ICML 2008)*, 2008.

H. Liu and J. Zhang. On the ℓ_1-ℓ_q regularized regression. Technical report, Carnegie Mellon University, 2008.

K. Lounici. Sup-norm convergence rate and sign concentration property of lasso and dantzig estimators. *Electronic Journal of Statistics*, 2:90–102, 2008.

L. Meier, S. van de Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society. Series B*, 70(1):53–71, 2008.

N. Meinshausen and P. Bühlmann. High dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34:1436–1462, 2006.

G. Miller. WordNet: An on-line lexical database. *Special Issue: International Journal of Lexicography*, 4(3), 1990.

V. K. Mootha, C. M. Lindgren, K.-F. Eriksson, A. Subramanian, S. Sihag, J. Lehar, P. Puigserver, E. Carlsson, M. Ridderstrale, E. Laurila, N. Houstis, M. J. Daly, N. Patterson, J. P. Mesirov, T. R. Golub, P. Tamayo, B. Spiegelman, E. S. Lander, J. N. Hirschhorn, D. Altshuler, and L. C. Groop. PGC-1α-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. *Nature Genetics*, 34:267 – 73, 2003. Datasets available at: `http://www.broad.mit.edu/gsea/datasets.jsp`.

Y. Nardi and A. Rinaldo. On the asymptotic properties of the group lasso estimator for linear models. *Electronic Journal of Statistics*, 2:605–633, 2008.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24: 227, 1995.

G. Obozinski, B. Taskar, and M. I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 2009.

E. O. Perlstein, D. M. Ruderfer, D. C. Roberts, S. L. Schreiber, and L. Kruglyak. Genetic basis of individual differences in the response to small-molecule drugs in yeast. *Nat Genet*, 39, 2007.

R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *ICML '06*, pages 713–720, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2.

A. Rakhlin. Transfer learning toolkit, 2007. Software available at: `http://multitask.cs.berkeley.edu`.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *JMLR*, 9:2491–2521, 2008.

J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11(2):416–431, 1983.

J. Rissanen. Hypothesis selection and testing by the mdl principle. *The Computer Journal*, 42: 260–269, 1999.

K. K. Schuler. Verbnet: A broad coverage, comprehensive verb lexicon. In *Ph.D. Thesis, Computer and Information Sciences, University of Pennsylvania*, June 2006.

G.: Schwartz. Estimating the dimensions of a model. *The 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*, 58:267–288, 1996.

J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50:2231–2242, 2004.

B.A. Turlach, W.N. Venables, and S.J. Wright. Simultaneous variable selection. *Technometrics*, 47 (3):349–363, 2005.

L. J. van 't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, January 2002. ISSN 0028-0836.

M. J. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso). *IEEE Trans. Inf. Theor.*, 55(5):2183–2202, 2009. ISSN 0018-9448.

Z. Wu and H. H. Zhou. Model selection and sharp asymptotic minimaxity. *Under Submission*, -(-): –, 2010. ISSN -.

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, February 2006. ISSN 1369-7412.

T. Zhang. On the convergence of mdl density estimation. In *COLT*, pages 315–330, 2004.

T. Zhang. Adaptive forward-backward greedy algorithm for sparse learning with linear models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1921–1928. Curran Associates, Inc., 2009a.

T. Zhang. On the consistency of feature selection using greedy least squares regression. *Journal of Machine Learning Research (JMLR)*, 10:555–568, 2009b. ISSN 1532-4435.

P. Zhao and B. Yu. On model selection consistency of lasso. *Journal of Machine Learning Research (JMLR)*, 7:2541–2563, 2006. ISSN 1532-4435.

P. Zhao, G. Rocha, and B. Yu. Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, 2008.

S. Zhou. Thresholding procedures for high dimensional variable selection and statistical estimation. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2304–2312. 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 12 (2011) 565-592 Submitted 12/09; Revised 10/10; Published 2/11

Variable Sparsity Kernel Learning

Jonathan Aflalo YAFLALO@CS.TECHNION.AC.IL
Faculty of Computer Science
Technion–Israel Institute of Technology
Haifa 32000, ISRAEL

Aharon Ben-Tal ABENTAL@IE.TECHNION.AC.IL
Faculty of Industrial Engineering and Management
Technion–Israel Institute of Technology
Haifa 32000, ISRAEL

Chiranjib Bhattacharyya CHIRU@CSA.IISC.ERNET.IN
Department of Computer Science and Automation
Indian Institute of Science
Bangalore 560012, INDIA

Jagarlapudi Saketha Nath SAKETH@CSE.IITB.AC.IN
Department of Computer Science and Engg.
Indian Institute of Technology Bombay
Mumbai 400076, INDIA

Sankaran Raman RAMANS@CSA.IISC.ERNET.IN
Department of Computer Science and Automation
Indian Institute of Science
Bangalore 560012, INDIA

Editor: Soeren Sonnenburg

Abstract

This paper[1] presents novel algorithms and applications for a particular class of mixed-norm regularization based Multiple Kernel Learning (MKL) formulations. The formulations assume that the given kernels are grouped and employ l_1 norm regularization for promoting sparsity within RKHS norms of each group and $l_s, s \geq 2$ norm regularization for promoting non-sparse combinations across groups. Various sparsity levels in combining the kernels can be achieved by varying the grouping of kernels—hence we name the formulations as Variable Sparsity Kernel Learning (VSKL) formulations. While previous attempts have a non-convex formulation, here we present a convex formulation which admits efficient Mirror-Descent (MD) based solving techniques. The proposed MD based algorithm optimizes over product of simplices and has a computational complexity of $O\left(m^2 n_{\text{tot}} \log n_{\max} / \varepsilon^2\right)$ where m is no. training data points, $n_{\max}, n_{\text{tot}}$ are the maximum no. kernels in any group, total no. kernels respectively and ε is the error in approximating the objective. A detailed proof of convergence of the algorithm is also presented. Experimental results show that the VSKL formulations are well-suited for multi-modal learning tasks like object categorization. Results also show that the MD based algorithm outperforms state-of-the-art MKL solvers in terms of computational efficiency.

Keywords: multiple kernel learning, mirror descent, mixed-norm, object categorization, scalability

1. All authors contributed equally. The author names appear in alphabetical order.

©2011 Jonathan Aflalo, Aharon Ben-Tal, Chiranjib Bhattacharyya, Jagarlapudi Saketha Nath and Sankaran Raman.

1. Introduction

This paper studies the problem of Multiple Kernel Learning (MKL) (Lanckriet et al., 2004; Bach et al., 2004; Sonnenburg et al., 2006; Rakotomamonjy et al., 2008) when the given kernels are assumed to be grouped into distinct components. Further, the focus is on the scenario where prior/domain knowledge warrants that each component is crucial for the learning task at hand. One of the key contributions of this paper is a highly efficient learning algorithm for this problem.

Recently Szafranski et al. (2008) extended the framework of MKL to the case where kernels are partitioned into groups and introduced a generic mixed-norm (that is (r,s)-norm; $r,s \geq 0$) regularization based MKL formulation (refer (11) in Szafranski et al., 2008) in order to handle groups of kernels. The idea is to employ a r-norm regularization over RKHS norms for kernels belonging to the same group and a s-norm regularization across groups. Though a generic formulation was presented, the focus of Szafranski et al. (2008) was on applications where it is known that most of the groups of kernels are noisy/redundant and hence only those mixed-norms promoting sparsity among kernels within and across groups were employed, for example, $0 < r,s < 2$ (following the terminology of Szafranski et al. (2008) this class of mixed-norm MKL formulations are henceforth called as "Composite Kernel Learning (CKL)" formulations). This paper presents a complementary study and focuses on applications where the domain knowledge guarantees that every group of kernels is crucial. Needless to say, all the groups of kernels need not be "equally" important and not all kernels belonging to a group may be important. More specifically, the focus of this paper is on the cases where $r = 1$ and $s \geq 2$ (including the limiting case[2] $s = \infty$). Here, $p = 1$ is employed for promoting sparsity among kernels belonging to the same group and $s \geq 2$ for promoting non-sparse combinations of kernels across groups. Note that the extreme cases: a) all of the kernels belong to one group b) Each group consists of a single kernel; correspond to the extreme sparse and non-sparse combinations of the given kernels. Since by varying the values of s and the groupings of kernels various levels of sparsity in combining the given kernels can be achieved, the formulations studied here are henceforth called as "Variable Sparsity Kernel Learning" (VSKL) formulations. As mentioned earlier, VSKL formulations are not well-studied in literature and this paper presents novel algorithms and applications for these formulations.

The VSKL formulations are motivated by multi-modal learning applications like object categorization where multiple feature representations need to be employed simultaneously for achieving good generalization. For instance, in the case of flower categorization feature descriptors for shape, color and texture need to be employed in order to achieve good visual discrimination as well as significant within-class variation (Nilsback and Zisserman, 2006). Combining feature descriptors for object categorization using the framework of MKL for object categorization has been a topic of interest for many recent studies (Varma and Ray, 2007; Nilsback and Zisserman, 2008) and is shown to achieve state-of-the-art performance. A key finding of Nilsback and Zisserman (2006) is the following: in object categorization tasks, employing few of the feature descriptors or employing a canonical combination of them often leads to sub-optimal solutions. Hence, in the framework of MKL, employing a block l_1 regularization, which is equivalent to selecting the "best" among the given kernels, as well as employing a l_2 regularization, which is equivalent to working with a canonical combination of the given kernels, may lead to sub-optimality. This observation clearly shows that state-of-the-art object categorization techniques (which are based on block l_1 regularized formulation) can further be improved. This paper proposes to employ the VSKL formulations for

2. This limiting case was discussed in an earlier version of this paper (Nath et al., 2009).

object categorization where the kernel are grouped based on the feature descriptor generating them. The $l_s(s \geq 2)$-norm regularization leads to non-sparse combinations of kernels generated from different feature descriptors and the l_1 norm leads to sparse selection of non-redundant/noisy kernels generated from a feature descriptor.

With this motivation, the key aspect investigated in this paper is an efficient algorithm for solving the VSKL formulations which are instances of non-smooth convex optimization problems. Except in the cases where $s = 2$ or $s = \infty$ or no. groups is one, the formulations cannot be solved using standard interior point based convex optimization software. Moreover, even in these special cases the generic interior point algorithms do not scale well to large data sets. The wrapper approach presented in Szafranski et al. (2008) cannot be employed for solving the VSKL formulations (that is, with $l_s, s \geq 2$ regularization across groups) efficiently as it solves a non-convex variant of the original convex formulation! The methods discussed in Szafranski et al. (2008); Kloft et al. (2010) are however efficient in the case $1 \leq s < 2$ (that is, sparse regularization across groups). In summary, efficient techniques for solving VSKL formulations indeed need to be devised. This paper adapts the Mirror-Descent (MD) (Ben-Tal et al., 2001; Beck and Teboulle, 2003; Ben-Tal and Nemirovski, 2001) procedure for solving a specific dual of VSKL leading to extremely scalable algorithms. MD is similar in spirit to the steepest descent algorithm; however involves a prox-function based regularizer rather than Euclidean norm based regularizer in the per-step auxiliary problem solved at each iteration. The prox-function is cleverly chosen based on the geometry of the feasibility set. Here, the feasibility set for the optimization problem tackled by MD turns out to be direct product of simplices, which is not a standard set-up discussed in optimization literature. We propose to employ the entropy function as the prox-function in the auxiliary problem solved by MD at each iteration and justify its suitability for the case of direct product of simplices. The MD based procedure for solving the dual of VSKL is henceforth called as `mirrorVSKL`.

Apart from the derivation of the `mirrorVSKL` algorithm, we also provide a detailed proof of its asymptotic convergence. `mirrorVSKL` is also of independent interest to the MKL community as it can solve the traditional MKL problem; namely the case when the number of groups is unity. The key advantages of `mirrorVSKL` over `simpleMKL` are: **a)** In case of `simpleMKL` in addition to gradient computation, the reduced gradient and step-size need to be determined which requires substantial computational effort; whereas in case of `mirrorVSKL`, pre-dominant computation at each iteration is that of calculating the gradient since the auxiliary problem has an analytical solution and the step-size can be computed easily **b)** It can be shown that the number of iterations with `mirrorVSKL` is nearly-independent of the number of kernels whereas no such a statement can be made in case of `simpleMKL`.

Simulations were performed on three real-world object categorization data sets: Caltech-101 (Fei-Fei et al., 2004), Caltech-256 (Griffin et al., 2007) and Oxford flowers (Nilsback and Zisserman, 2006) for comparing the generalization ability of the VSKL and existing MKL formulations. The results show that the proposed formulation are well-suited for multi-modal tasks like object categorization. In the special case of number of groups unity, the `mirrorVSKL` and `simpleMKL` algorithms were compared in terms of computational effort and scalability. The scalability experiments were performed on few UCI data sets (Blake and Merz, 1998) following the experimental set-up of Szafranski et al. (2008). Results showed that `mirrorVSKL` scales well to large data sets with large no. kernels and in some cases was eight times faster than `simpleMKL`.

The remainder of this paper is organized as follows: in Section 2, the VSKL and related MKL formulations are presented. The section also presents a specific dual of VSKL which admits efficient

MD based solving techniques. The main contribution of the paper, `mirrorVSKL` is presented in Section 3. A detailed proof of convergence of `mirrorVSKL` is also presented. Section 4 presents a summary of the numerical experiments carried for verifying the major claims of the paper. Section 5 concludes the paper with a brief summary and discussion.

2. Variable Sparsity Kernel Learning Formulation

This section presents the VSKL formulation and a specific dual of it. Though the formalism can be extended to various learning tasks we focus on the task of binary classification in the rest of the paper. We begin by introducing some notation: let the training data set be denoted by $\mathcal{D} = \{(\mathbf{x}_i, y_i),\ i = 1,\ldots,m \mid \mathbf{x}_i \in \mathcal{X},\ y_i \in \{-1,1\}\}$. Here, $\mathbf{x}_i$ represents the i^{th} training data point with label y_i. Let $\mathbf{Y}$ denote the diagonal matrix with entries as y_i. Suppose the given kernels are divided into n groups and the j^{th} group has n_j number of kernels. Let the feature-space mapping induced by the k^{th} kernel of the j^{th} component be $\phi_{jk}(\cdot)$ and the corresponding gram-matrix of training data points be $\mathbf{K}_{jk}$.[3] Also, let $\mathbf{Q}_{jk} = \mathbf{Y}\mathbf{K}_{jk}\mathbf{Y}$.

For now, to keep things simple, let us assume that each of the kernels is such that the induced feature mapping is finite dimensional; later on we will generalize and remove this assumption. Each individual example can now be described by a concatenation of all the feature vectors:

$$\mathbf{x}^\top = \left[\phi_{11}(\mathbf{x})^\top \ldots \phi_{jk}(\mathbf{x})^\top \ldots \phi_{nn_n}(\mathbf{x})^\top\right].$$

Consider the problem of learning a linear discriminant function of the form

$$f(\mathbf{x}) = \sum_{j=1}^{n}\sum_{k=1}^{n_j} \mathbf{w}_{jk}^\top \phi_{jk}(\mathbf{x}) - b.$$

Given a training set the idea is to learn a $\mathbf{w} \equiv [\mathbf{w}_{11}^\top \mathbf{w}_{12}^\top \ldots \mathbf{w}_{nn_n}^\top]^\top$ and b which generalizes well. This could be achieved by minimizing an objective of the form (Vapnik, 1998):

$$J(\mathbf{w}) = \Omega(\mathbf{w}) \ + \ C\, L(\mathcal{D}),$$

where Ω is a suitable regularizing function, L is a loss function which penalizes errors on the training set and C is a regularization parameter. SVMs (Vapnik, 1998) usually use $\Omega(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$ and $L = \sum_{i=1}^{m} \max(1 - y_i(\mathbf{w}^\top \mathbf{x}_i - b), 0)$. It is easy to see that this formulation corresponds to employing a kernel which is essentially the sum of all the given kernels. Hence-forth, we denote this formulation by **SVM** and use it as a baseline for comparison in the experiments.

The regularization term can be an extremely useful tool for modelling various different kinds of data. The choice of Ω should be such that this is tractable and yet flexible enough to enforce different relationships between groups dictated by modelling requirements. Recently (Szafranski et al., 2008) employed a regularization of the form

$$\frac{1}{2}\|\mathbf{w}\|_{r,s}^2 \ 0 \le r < 2, 0 \le s < 2,$$

where

$$\|\mathbf{w}\|_{(r,s)} = \left\{ \sum_{j=1}^{n} \left\{ \sum_{k=1}^{n_j} \|\mathbf{w}_{jk}\|_2^r \right\}^{\frac{s}{r}} \right\}^{\frac{1}{s}}.$$

3. The gram-matrices are unit-trace normalized.

Since the primary goal of Szafranski et al. (2008) is to achieve sparsity, the focus was only on the cases $0 \leq r < 2, 0 \leq s < 2$ making most of the individual norms $\|\mathbf{w}_{jk}\|$ zero at optimality. Henceforth, this formulation is denoted by $\mathbf{CKL}_{r,s}$ where r,s represent the within and across group norms respectively.

However as discussed above in case of multi-modal tasks like object categorization, it is often desirable that there is sparsity within the group but all the groups need be active. In view of this we begin by defining

$$\Omega_{(p,q)}(\mathbf{w}) = \frac{1}{2}\left\{\sum_{j=1}^{n}\left\{\sum_{k=1}^{n_j}\|\mathbf{w}_{jk}\|_2^{2p}\right\}^{\frac{q}{p}}\right\}^{\frac{1}{q}}.$$

This can be interpreted as a mixed norm operating on $\|\mathbf{w}_{jk}\|^2$ and the following relationship holds

$$\Omega_{(p,q)}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_{r,s}^2,\ r = 2p, s = 2q.$$

In this paper we analyze the case $p = \frac{1}{2}$ and $q \geq 1$ which is equivalent to considering an l_1 (sparse) norm regularization within kernels of each group and $l_s (s \geq 2)$ (non-sparse) norm across groups. In other words, we consider the following regularization:

$$\Omega(\mathbf{w}) = \frac{1}{2}\left\{\sum_{j=1}^{n}\left\{\sum_{k=1}^{n_j}\|\mathbf{w}_{jk}\|_2\right\}^{2q}\right\}^{\frac{1}{q}},$$

where $q \geq 1$. By varying the groupings of kernels various levels of sparsity can be achieved: no. of groups is unity corresponds to extreme sparse selection of kernels and no. groups equal to no. kernels corresponds to non-sparse combinations of kernels. The flexibility in choice of q offers different modelling perspectives and correspond to various ways for achieving non-sparse combinations across groups. Since this formulation allows for flexibility from sparsity to non-sparsity, it is called as the Variable Sparsity Kernel Learning (VSKL) formulation and denoted by $\mathbf{VSKL}_q$, where $q \geq 1$:

$$\begin{aligned}\min_{\mathbf{w}_{jk},b,\xi_i} \quad & \tfrac{1}{2}\left[\textstyle\sum_j \left(\sum_{k=1}^{n_j}\|\mathbf{w}_{jk}\|_2\right)^{2q}\right]^{\frac{1}{q}} + C\textstyle\sum_i \xi_i \\ \text{s.t.} \quad & y_i\left(\textstyle\sum_{j=1}^{n}\sum_{k=1}^{n_j}\mathbf{w}_{jk}^\top\phi_{jk}(\mathbf{x}_i) - b\right) \geq 1-\xi_i,\ \xi_i \geq 0\ \forall\ i. \end{aligned} \tag{1}$$

In the extreme case $q \to \infty$, the regularization term is to be written as $\frac{1}{2}\max_j\left(\sum_{k=1}^{n_j}\|\mathbf{w}_{jk}\|_2\right)^2$. Note that the traditional MKL formulation using the block l_1 norm regularization (Rakotomamonjy et al., 2008) is a special case of VSKL when the number of groups is unity. We denote this special case by **MKL** and as mentioned earlier, state-of-the-art object categorization performance is achieved using this methodology.

Existing wrapper approaches (Szafranski et al., 2008; Rakotomamonjy et al., 2008) are useful in solving (1) only for the cases $q < 1$. For $1 \leq q < \infty$, the wrapper approaches solve a non-convex variant of the convex formulation and hence are not well-suited. Moreover these wrapper approaches cannot be easily extended to handle the important case $q \to \infty$. In this paper we describe a first order method based on mirror descent procedure which efficiently solves the VSKL formulation for

all values of $q \geq 1$ (including $q \to \infty$) and provably converges to the global optimum. The mirror descent procedure solves a specific dual of the VSKL formulation—details of which are presented in the following.

2.1 Dual of VSKL

This section presents a dual of **VSKL** which admits efficient MD solving techniques. In the rest of the paper $q^* = \frac{q}{q-1}, q \geq 1$ (if $q = 1$ then $q^* = \infty$ and if $q = \infty$ then $q^* = 1$). If $1 \leq r < \infty$, the following sets $\Delta_{d,r} = \left\{\gamma \equiv [\gamma_1 \dots \gamma_d]^\top \mid \sum_{i=1}^d \gamma_i^r \leq 1, \gamma_i \geq 0, i = 1, \dots, d\right\}$ are convex. As $r \to \infty$ one obtains a d-dimensional box $\Delta_{d,\infty} = B_d = \{\gamma \mid 0 \leq \gamma_i \leq 1 \; i = 1, \dots, d\}$. If $r = 1$ we get back a d-dimensional simplex, and to lighten notation we will denote $\Delta_{d,1} = \Delta_d$. At this point it would be useful to recall the following lemma (see Boyd and Vandenberghe, 2004, Section A.1.6):

Lemma 2.1 *Let* $\mathbf{a}$ *be a d-dimensional vector with non-negative components, that is, $a_i \geq 0$ $i = 1, \dots, d$. Then*

$$\|\mathbf{a}\|_r = sup_{\gamma \in \Delta_{d,r^*}} \gamma^\top \mathbf{a},$$

where $r \geq 1$ and r^* verifies $\frac{1}{r} + \frac{1}{r^*} = 1$.

A specialization of this lemma for $r \to \infty$ is:

$$\max_i \{a_i\} = sup_{\gamma \in \Delta_d} \gamma^\top \mathbf{a}.$$

We also note the following result which will be used in later derivations (see Micchelli and Pontil, 2005):

Lemma 2.2 *Let $a_i \geq 0, i = 1, \dots, d$ and $1 \leq r < \infty$. Then, for $\Delta_{d,r}$ defined as before,*

$$\min_{\eta \in \Delta_{d,r}} \sum_i \frac{a_i}{\eta_i} = \left(\sum_{i=1}^d a_i^{\frac{r}{r+1}} \right)^{1+\frac{1}{r}},$$

and the minimum is attained at

$$\eta_i = \frac{a_i^{\frac{1}{r+1}}}{\left(\sum_{i=1}^d a_i^{\frac{r}{r+1}} \right)^{\frac{1}{r}}}.$$

Here, by convention, $a/0$ is 0 if $a = 0$ and is ∞ otherwise. In the limit $r \to \infty$ the following holds

$$\min_{\eta \in B_d} \sum_i \frac{a_i}{\eta_i} = \sum_{i=1}^d a_i,$$

where B_d is defined as before and equality is is attained at $\eta_i = 1 \; \forall \; a_i > 0$.

Proof The proof follows by employing the Karush-Kuhn-Tucker conditions, which are here necessary and sufficient for optimality. ■

Using Lemma 2.1, the objective in (1), for any $q \geq 1$, becomes:

$$\frac{1}{2} \max_{\gamma \in \Delta_{n,q^*}} \sum_{j=1}^n \gamma_j \left(\sum_{k=1}^{n_j} \|\mathbf{w}_{jk}\| \right)^2 + C \sum_i \xi_i. \tag{2}$$

For the case $q \to \infty$ the set Δ_{n,q^*} reduces to a simplex, Δ_n. Further, by Lemma 2.2 (with $d = n, r = 1$):

$$\left(\sum_{i=1}^{n} \sqrt{a_i}\right)^2 = \min_{\lambda \in \Delta_n} \sum_{i=1}^{n} \frac{a_i}{\lambda_i},$$

so (2) can be written equivalently as:

$$\max_{\gamma \in \Delta_{n,q^*}} \min_{\lambda_j \in \Delta_{n_j}} \left[\underbrace{\frac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n_j} \gamma_j \frac{\|\mathbf{w}_{jk}\|^2}{\lambda_{jk}} + C \sum_i \xi_i}_{f(w,\lambda,\gamma,\xi)} \right].$$

The equivalent primal formulation we arrive at is finally

Problem (P)

$$\min_{\xi_i, b, \mathbf{w}_{jk}} \left[\max_{\gamma \in \Delta_{n,q^*}} \min_{\lambda_j \in \Delta_{n_j}} f(\mathbf{w}, \lambda, \gamma, \xi) \right]$$

$$\text{s.t.} \qquad y_i \left(\sum_{j=1}^{n} \sum_{k=1}^{n_j} \mathbf{w}_{jk}^T \phi_{jk}(\mathbf{x}_i) - b \right) \geqslant 1 - \xi_i, \ \forall\, i, \tag{3}$$

$$\xi_i \geqslant 0\ , \ \ \forall i. \tag{4}$$

Note that at optimality, the following relations hold

$$\begin{aligned} \lambda_{jk} = 0 \quad &\Rightarrow \quad \mathbf{w}_{jk} = 0, \\ \text{if } q \neq \infty, \text{then } \gamma_j = 0 \quad &\Leftrightarrow \quad \mathbf{w}_{jk} = 0 \ \forall\, k. \end{aligned}$$

In case $q = \infty$, $\mathbf{w}_{jk} = 0 \ \forall\, k \Rightarrow \gamma_j = 0$ unless $\mathbf{w}_{jk} = 0 \ \forall\ j,k$, which is an un-interesting case. Let us fix the variables ξ, b and $\mathbf{w}$ in problem (P) and consider the $\max_\gamma \min_\lambda$ part in the square brackets:

$$\max_{\gamma} \min_{\lambda} \left\{ f(\mathbf{w}, \lambda, \gamma, \xi) \mid \lambda \in \bigotimes_j \Delta_{n_j}, \gamma \in \Delta_{n,q^*} \right\}.$$

The objective function is concave (linear) in γ and convex in λ, and the feasible sets $\bigotimes_j \Delta_{n_j}$, Δ_{n,q^*} are convex and compact. Hence, by the Sion-Kakutani minmax theorem (Sion, 1958), the maxmin can be interchanged, and when this is done, problem (P) becomes

$$\min_{\xi_i, b, \mathbf{w}_{jk}} \min_{\lambda \in \bigotimes_j \Delta_{n_j}} \max_{\gamma \in \Delta_{n,q^*}} f(\mathbf{w}, \lambda, \gamma, \xi), \qquad \text{s.t. (3), (4),}$$

or similarly

$$\min_{\lambda \in \bigotimes_j \Delta_{n_j}} \left[\min_{\xi_i, b, \mathbf{w}_{jk}} \max_{\gamma \in \Delta_{n,q^*}} f(\mathbf{w}, \lambda, \gamma, \xi), \qquad \text{s.t. (3), (4)} \right]. \tag{5}$$

Now, f is convex in $(\xi, b, \mathbf{w})$ and concave (linear) in γ. The set for feasible $(\xi, b, \mathbf{w})$, expressed in (3), (4) is closed and convex, and $\bigotimes_j \Delta_{n_j}$ is convex compact. Hence, by a minmax theorem (Rockafellar, 1964), the minmax in the square brackets in (5) can be interchanged and we arrive at

$$\min_{\lambda \in \bigotimes_j \Delta_{n_j}} \max_{\gamma \in \Delta_{n,q^*}} \left\{ \min_{\xi_i, b, \mathbf{w}_{jk}} f(\mathbf{w}, \lambda, \gamma, \xi) \mid \text{ s.t. (3), (4)} \right\}. \tag{6}$$

Replacing the convex problem in the curly brackets in (6) by its dual the following theorem is immediate:

Theorem 2.3 *Let* $\mathbf{Q}_{jk}$ *be the* $m \times m$ *matrix*

$$\left(\mathbf{Q}_{jk}\right)_{ih} = y_h y_i \phi_{jk}(\mathbf{x}_i)^\top \phi_{jk}(\mathbf{x}_h) \qquad i,h = 1,\ldots,m.$$

The dual problem of (P) w.r.t. the variables $(\mathbf{w}, b, \xi)$ *is the following:*[4]

$$\text{Problem (D)} \quad \left\{ \begin{array}{l} \min\limits_{\lambda \in \bigotimes_j \Delta_{n_j}} \quad \max\limits_{\alpha \in S_m, \gamma \in \Delta_{n,q^*}} \left\{ \sum \alpha_i - \frac{1}{2}\alpha^T \left(\sum\limits_{j=1}^{n} \sum\limits_{k=1}^{n_j} \frac{\lambda_{jk}\mathbf{Q}_{jk}}{\gamma_j} \right) \alpha \right\}, \\ \text{where } \; S_m = \left\{ \alpha \in \mathbb{R}^m \mid \sum\limits_{i=1}^{m} \alpha_i y_i = 0, \;\; 0 \leqslant \alpha_i \leqslant C, \quad i = 1,\ldots,m \right\}. \end{array} \right.$$

The relation between the primal and dual variables is given by: $\gamma_j \frac{\mathbf{w}_{jk}}{\lambda_{jk}} = \sum_{i=1}^{m} \alpha_i y_i \phi_{jk}(\mathbf{x}_i)$. Note that (D) is only a partial dual (wrt. variables $\mathbf{w}, b, \xi$) of (P) and is not the joint dual. Interestingly the partial dual can be efficiently solved using a non-Euclidean gradient-descent based approach (see Section 3) and hence is explored here. In the following, we generalize this discussion using the functional framework and remove the restriction that the induced feature maps are finite dimensional.

2.1.1 THE FUNCTIONAL FRAMEWORK

We first consider the case $1 \leq q < \infty$. Let K_{jk} be positive kernel functions defined over the same input space $\mathcal{X}$. Each K_{jk} defines a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_{jk}$ with the inner product $\langle .,. \rangle_{\mathcal{H}_{jk}}$. An element $h \in \mathcal{H}_{jk}$ has the norm $\|h\|_{\mathcal{H}_{jk}} = \sqrt{\langle h,h \rangle_{\mathcal{H}_{jk}}}$. Now for any λ_{jk} non-negative, define a new Hilbert space

$$\mathcal{H}'_{jk} = \{h | h \in H_{jk}, \; \frac{\|h\|_{\mathcal{H}_{jk}}}{\lambda_{jk}} < \infty\}$$

with inner product as $\langle .,. \rangle_{\mathcal{H}'_{jk}} = \frac{1}{\lambda_{jk}} \langle .,. \rangle_{\mathcal{H}_{jk}}$. We use the convention that if $\lambda_{jk} = 0$ then the only member of $\mathcal{H}'_{jk}$ is $h = 0$. It is easy to see that $\mathcal{H}'_{jk}$ is an RKHS with kernel as $\lambda_{jk} K_{jk}$ (see Rakotomamonjy et al., 2008). A direct sum of such RKHS, $\mathcal{H}_j = \bigoplus_k \mathcal{H}'_{jk}$ is also an RKHS with the kernel as $K_j = \sum_k \lambda_{jk} K_{jk}$. Now again, for a given γ_j non-negative, consider Hilbert spaces $\mathcal{H}'_j$ derived from $\mathcal{H}_j$ as follows: a) if $\gamma_j = 0$ then $\mathcal{H}'_j$ contains only the zero element and if $\gamma_j > 0$ then elements in $\mathcal{H}'_j$ as same as those in $\mathcal{H}_j$ however $\langle .,. \rangle_{\mathcal{H}'_j} = \gamma_j \langle .,. \rangle_{\mathcal{H}_j}$. Again $\mathcal{H}'_j$ are RKHS with kernels as $\frac{1}{\gamma_j} K_j = \frac{1}{\gamma_j} \sum_k \lambda_{jk} K_{jk}$ and their direct sum is in-turn an RKHS $\mathcal{H}$ with kernel as $K = \sum_{j=1}^{n} \frac{1}{\gamma_j} \sum_{k=1}^{n_j} \lambda_{jk} K_{jk}$. With this functional framework in mind we now let $\mathbf{w}_{jk}$ be an element of $\mathcal{H}_{jk}$ with the norm $\|\mathbf{w}_{jk}\|_{\mathcal{H}_{jk}} = \sqrt{\langle w_{jk}, w_{jk} \rangle}_{\mathcal{H}_{jk}}$ and let $\mathbf{w} \in \mathcal{H}$ where $\mathcal{H}$ is as defined above. The primal (P) in this case reads as follows:

4. Only for the case $q = \infty$, we make an additional assumption that all the base kernels are strictly positive in order to write the dual in the form of problem (D) above.

$$\min_{\xi_i, b, \mathbf{w}_{jk} \in \mathcal{H}_{jk}} \left\{ \max_{\gamma_j \in \Delta_{n,q^*}} \min_{\lambda_j \in \Delta_{n_j}} f(\mathbf{w}, \lambda, \gamma, \xi) \right\} \quad \text{s.t.} \quad y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle_{\mathcal{H}} - b \right) \geqslant 1 - \xi_i, \xi_i \geqslant 0, \tag{7}$$

where $f(\mathbf{w}, \lambda, \gamma, \xi) = \frac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n_j} \gamma_j \frac{\|\mathbf{w}_{jk}\|^2_{\mathcal{H}_{jk}}}{\lambda_{jk}} + C \sum_i \xi_i$.

Following the usual procedure for generalizing linear SVMs to RKHS via Representer theorem one obtains the following generalization of Theorem 2.3:

Theorem 2.4 *Let $\mathbf{Q}_{jk}$ be the $m \times m$ matrix*

$$\left(\mathbf{Q}_{jk}\right)_{ih} = y_h y_i K_{jk}(\mathbf{x}_i, \mathbf{x}_h) \qquad i, h = 1, \ldots, m.$$

The dual problem of (7) with respect to $\{\mathbf{w}, b, \xi\}$ is the following optimization problem:

$$\min_{\lambda_j \in \Delta_{n_j}} \underbrace{\max_{\alpha \in S_m, \gamma \in \Delta_{n,q^*}} \overbrace{\mathbf{1}^T \alpha - \frac{1}{2} \alpha^T \left(\sum_{j=1}^{n} \sum_{k=1}^{n_j} \frac{\lambda_{jk} Q_{jk}}{\gamma_j} \right) \alpha}^{f_\lambda(\alpha, \gamma)}}_{G(\lambda)}, \tag{D}$$

where $S_m = \left\{ \alpha \in \mathbb{R}^m | 0 \leqslant \alpha \leqslant C, y^T \alpha = 0 \right\}$.

We omit the proof as it is straightforward. To be noted that $\frac{\gamma_j}{\lambda_{jk}} \mathbf{w}_{jk}(.) = \sum_i \alpha_i y_i K_{jk}(., x_i)$ and all other conditions remain same.[5]

We will refer (D) as the dual problem. The dual (D) problem provides more insight into the formulation: λ_{jk} can be viewed as a weight given to the kernel K_{jk} and $\frac{1}{\gamma_j}$ can be thought of as an additional weight factor for the entire j^{th} group/descriptor. Since $\lambda_j \in \Delta_{n_j}$ (that is, λ_js are l_1 regularized), most of the λ_js will be zero at optimality and since $\gamma \in \Delta_{n,q^*}$, it amounts to combining kernels across descriptors in a non-trivial (and in case $q^* \geq 2$ in a non-sparse) fashion. Indeed, this is in-sync with findings of Nilsback and Zisserman (2006): kernels from different feature descriptors (components) are combined using non-trivial weights (that is, $\frac{1}{\gamma_j}$); moreover, only the "best" kernels from each feature descriptor (component) are employed by the model. This sparsity feature leads to better interpretability as well as computational benefits during the prediction stage. Note that in the case optimal weights (λ, γ) are known/fixed, then the problem is equivalent to solving an SVM with an effective kernel: $\mathbf{K}_{eff} \equiv \sum_{j=1}^{n} \left(\frac{\sum_{k=1}^{n_j} \lambda_{jk} \mathbf{K}_{jk}}{\gamma_j} \right)$. This observation leads to an efficient algorithm for solving the dual which is described in the subsequent section.

3. Algorithm for Solving the Dual Problem

This section presents the mirror descent based algorithm for efficiently solving the dual (D). A detailed proof of convergence of the algorithm is also presented. We begin by re-writing problem (D) as a minimization problem, rather than a minimax problem:

$$\min\{G(\lambda_1, \lambda_2, \ldots, \lambda_n) \mid \lambda_j \in \Delta_{n_j},\ j = 1, \ldots, n\}, \tag{8}$$

5. Again, for the case $q = \infty$, we make the assumption that all base kernels are strictly positive in order that Theorem 2.4 is true.

where the objective function G is the optimal value function of the following problem:

$$G(\lambda_1,\lambda_2,\ldots,\lambda_n) = \max_{\gamma\in\Delta_{n,q^*},\alpha\in S_m} \underbrace{\left\{ \mathbf{1}^T\alpha - \frac{1}{2}\alpha^T \sum_{j=1}^{n} \left(\frac{\sum_k \lambda_{jk} Q_{jk}}{\gamma_j} \right) \alpha \right\}}_{f_\lambda(\alpha,\gamma)}. \tag{9}$$

The function G is *convex* in $\lambda \in \mathbb{R}^n$ since it is the point-wise maximize of functions which are *linear* in λ. The minimization problem (8) is then that of minimizing a convex (possibly non-differentiable) function over a product of simplices. Problems with these features, even large-scale ones, can be solved efficiently by a Mirror Descent (MD) type algorithm (Ben-Tal et al., 2001; Beck and Teboulle, 2003) which is reviewed in the next subsection. An MD algorithm needs as input in each iteration a sub-gradient $G'(\lambda)$ belonging to the sub-gradient set $\partial G(\lambda)$. Using Danskin's theorem (see Bertsekas, 1999, prop. B.25), these elements are readily available from the solution of the *concave maximization problem* (in vector variables, γ and α) in (9).[6] A procedure for solving this maximization problem efficiently is presented in Section 3.3. Note that the maximum problem is solved numerically and hence the approximate sub-gradient is only obtained. Though we provide convergence analysis, it does not deal with the issue of approximate sub-gradient. Analysis of such situations is more involved and we postpone it to future work (see D'Aspermont, 2008).

3.1 Introduction to Mirror Descent

Consider the following problem.

$$\min f(x) \qquad x \in X, \tag{10}$$

where:

1. $X \subset \mathbb{R}^n$ is convex and closed with nonempty interior.
2. The objective function $f : X \to \mathbb{R}$ is a convex Lipschitz continuous function, with respect to a fixed given norm $\|\cdot\|$, that is:

$$\exists L, |f(x) - f(y)| \leqslant L\|x-y\| \quad \forall x,y \in \mathrm{int}X.$$

3. There exists an *oracle* which given $x \in X$ computes $f(x)$ and $f'(x) \in \partial f(x)$.

For such problems a classical algorithm is the Sub-gradient Projection Algorithm (SPA), which generates iteratively the sequence $\{x^t\}$ via:

$$x^{t+1} = \pi_X(x^t - s_t f'(x^t)),$$

where s_t is a step-size, and $\pi_X(y) = \underset{x\in X}{\mathrm{argmin}} \{\|x-y\|_2\}$ is the projection of y on X. The SPA can be rewritten equivalently as

$$x^{t+1} = \underset{x\in X}{\mathrm{argmin}} \left\{ \langle x, s_t f'(x^t)\rangle + \frac{\|x-x^t\|_2^2}{2} \right\}.$$

6. If α^*,γ^* represent the variables maximizing f for given λ, then the jk^{th} component of the sub-gradient $G'(\lambda)$ is $-\frac{1}{2}\frac{\alpha^{*\top}\mathbf{Q}_{jk}\alpha^*}{\gamma_j^*}$.

The main idea of Mirror Descent Algorithm(MDA) is to replace the distance function $\frac{1}{2}\|x-x^t\|_2^2$ based on the Euclidean norm by a general distance-like function $D(x,x^t)$ (also referred to as prox-function). The basic iteration step then becomes

$$x^{t+1} = \operatorname*{argmin}_{x\in X} \left\{ \langle x, s_t f'(x^t)\rangle + D(x,x^t) \right\}. \tag{11}$$

With the freedom to choose D one can adapt it to the specific constraint set X. The minimal requirements on the "distance function" are

1. D is nonnegative,
2. $D(u,v) = 0$ if and only if $u = v$.

A possible way to construct such a distance-like function is as follows: Let $\Phi : X \to \mathbb{R}$ be strongly convex with parameter $\sigma > 0$ with respect to a norm $\|\ \|$, that is:

$$\langle \nabla\Phi(x) - \nabla\Phi(y), x-y\rangle \geq \sigma\|x-y\|^2, \quad \forall x,y \in X.$$

Then

$$B_\Phi(x,y) = \Phi(x) - \Phi(y) - \langle x-y, \nabla\Phi(y)\rangle$$

is a distance-like function (often called Bregman Divergences). With this choice, the iteration scheme (11) is equivalent (see Beck and Teboulle, 2003) to the following three step procedure

$$\begin{aligned} &1. \quad x^t \leftarrow \nabla\Phi^*(y^t), \\ &2. \quad y^{t+1} \leftarrow \nabla\Phi(x^t) - s_t f'(x^t), \\ &3. \quad x^{t+1} \leftarrow \nabla\Phi^*(y^{t+1}) = \nabla\Phi^*(\nabla\Phi(x^t) - s_t f'(x^t)). \end{aligned} \tag{12}$$

Here $\Phi^*(y) = \max_{x\in X}\{\langle x,y\rangle - \Phi(x)\}$ is the *conjugate* function of Φ.

This procedure yields efficient convergent algorithms for solving (10). More formally we state the following theorem proved in Beck and Teboulle (2003)

Theorem 3.1 *Let $\{x^t\}$ be the sequence generated from a starting point $x^1 \in \operatorname{int} X$ by the MD procedure outlined in (12) with the D being the Bregman Divergence $B_\phi(\cdot,\cdot)$. Let $f^* = \min_{x\in X} f(x)$, and let $x^* \in X$ be a point where the minimum is attained. Then for every $t \geqslant 1$*

1.

$$\min_{1\leqslant\tilde{t}\leqslant t} f(x^{\tilde{t}}) - f^* \leqslant \frac{B_\Phi(x^*,x^1) + 2\sigma^{-1}\sum_{\tilde{t}=1}^t s_{\tilde{t}}^2 \|f'(x^{\tilde{t}})\|_*^2}{\sum_{\tilde{t}=1}^t s_{\tilde{t}}},$$

where σ is the strong-convexity parameter of Φ.

2. In particular if the step size sequence $\{s_t\}$ satisfies

$$\sum_{\tilde{t}=1}^t s_{\tilde{t}} \to \infty, s_t \to 0, t \to \infty,$$

then the method converges, that is:

$$t \to \infty \Rightarrow \min_{1\leq\tilde{t}\leq t} f(x^{\tilde{t}}) - f^* \to 0,$$

3. Moreover if the step-sizes s_t are chosen as

$$s_t = \sqrt{\frac{2\sigma\Gamma(x^1)}{L_f^2 t}},$$

then the following efficiency estimate holds

$$\min_{1\leqslant\tilde{t}\leqslant t} f(x^{\tilde{t}}) - f^* \leqslant L_f\sqrt{\frac{2\Gamma(x^1)}{\sigma t}},$$

where $\Gamma(x^1) = \max_{x\in X} B_\Phi(x,x^1)$ measures the "width" of the feasible set X. □

The above theorem shows that MD procedures require $O(\frac{L_f^2\Gamma(x^1)}{\sigma\varepsilon^2})$ iterations for attaining an ε accurate solution where each iteration is very cheap, requiring just a gradient computation.

3.2 Minimizing G by Mirror Descent Procedures

In the following we discuss the suitability of MD procedures outlined in (12) for minimizing G given in (9).

For an MD procedure to apply we first need to demonstrate that G is convex and Lipschitz continuous. We also need to devise a Distance generating function which is suitable for a feasible set comprised of a product of simplices. We begin with the proposition

Proposition 3.1 *If there exists scalars $0<\tau<1, \mu>0$ such that all eigenvalues of each $\mathbf{Q}_{jk}$ matrix lie within an interval $(\tau\mu,\mu)$, then the function G given by*

$$G(\lambda_1,\cdots,\lambda_n) = \max_{\alpha\in S_m,\gamma\in\Delta_{n,q^*}} \mathbf{1}^T\alpha - \frac{1}{2}\alpha^T\left[\sum_{j=1}^n\left(\frac{\sum_{k=1}^{n_j}\lambda_{jk}\mathbf{Q}_{jk}}{\gamma_j}\right)\right]\alpha$$

is convex and Lipschitz continuous w.r.t. in the l_1 norm for any $q\geq 1$.

Proof See Appendix for a proof. ■

A suitable Distance generating function of the form B_Φ over product of simplices is given in the following

Proposition 3.2 *Let*

$$\Phi_j(\lambda_j) = \sum_{k=1}^{n_j}\lambda_{jk}\ln(\lambda_{jk}),\ \lambda_j\in\Delta_j\ \forall j=1,\ldots,n.$$

The function $\Phi(\lambda) = \sum_{j=1}^n\Phi_j(\lambda_j) = \sum_{j=1}^n\sum_{k=1}^{n_j}\lambda_{jk}\ln(\lambda_{jk})$ is strongly convex with parameter $\frac{1}{n}$ with respect to the l_1 norm. The corresponding distance generating function is given by

$$B_\Phi(\lambda^*,\lambda^1) = \sum_{j=1}^n\sum_{k=1}^{n_j}\lambda^*_{jk}\ln\left(\frac{\lambda^*_{jk}}{\lambda^1_{jk}}\right).$$

Proof The function Φ_j is convex in λ_j as its Hessian is positive definite over the interior of its domain. Since Φ is a sum of such functions it is also convex.

Recall that a necessary and sufficient condition (Rockafellar, 1970) for a convex function Φ to be strongly convex with respect to a norm, $\|.\|$, and parameter σ, is

$$\langle \nabla\Phi(\lambda) - \nabla\Phi(\lambda^*), \lambda - \lambda^* \rangle \geq \sigma \|\lambda - \lambda^*\|^2,$$

where $\nabla\Phi(\lambda)$ is an element in the sub-gradient set of Φ evaluated at λ.

The proof can now be constructed as follows

$$\begin{aligned}\langle \nabla\Phi(\lambda) - \nabla\Phi(\lambda^*), \lambda - \lambda^* \rangle &= \sum_{j=1}^{n} KL(\lambda_j, \lambda_j^*) \\ &\geq \sum_{j=1}^{n} \|\lambda_j - \lambda_j^*\|_1^2 \\ &\geq \frac{1}{n} \|\lambda - \lambda^*\|_1^2,\end{aligned}$$

wherein the first equality $KL(p,q) = \sum_i p_i \log \frac{p_i}{q_i}$. The first inequality is obtained by noting that $KL(p,q) \geq \|p-q\|_1^2$ (see Cover and Thomas, 2006). The second inequality is valid since for any nonnegative a_j one has by Cauchy-Schwartz inequality, $\frac{1}{n}(\sum_{j=1}^n a_j)^2 \leq \sum_{j=1}^n a_j^2$. This proves that Φ is strongly convex with parameter $\frac{1}{n}$ in the l_1 norm.

Finally, the function B_Φ can be written as

$$B_\Phi(\lambda^*, \lambda) = \Phi(\lambda^*) - \Phi(\lambda) - \langle \nabla\Phi(\lambda), \lambda^* - \lambda \rangle.$$

Hence, it is indeed a Bregman-type distance generating function ■

3.2.1 THE CHOICE OF STEP-SIZE

By Theorem 3.1 the choice of step-size is guided by the term $\Gamma(\lambda^1)$, where λ^1 is in the interior of the product of simplices. If one chooses $\lambda_{jk}^1 = \frac{1}{n_j}$ then one can obtain an estimate of $\Gamma(\lambda^1)$ as follows:

$$B_\Phi(\lambda^*, \lambda^1) \leq \sum_{j=1}^{n} \log n_j \leq n \log n_{\max} \text{ where } n_{\max} = \max_j n_j.$$

The first inequality follows from the fact that $\sum_k \lambda_{jk} \log \lambda_{jk} \leq 0, \forall \lambda \in \bigotimes_j \Delta_{n_j}$ and the second inequality follows from the definition of $n_{\max}$. This upper bound immediately yields $\Gamma(\lambda^1) \leqslant n \log n_{\max}$. The candidate step-size (refer Theorem 3.1) now writes as

$$s_t = \frac{\sqrt{2\frac{1}{n} n \log n_{\max}}}{L_G} \frac{1}{\sqrt{t}} = \frac{\sqrt{2 \log n_{\max}}}{L_G} \frac{1}{\sqrt{t}},$$

where L_G is the Lipschitz constant of G. However this step-size estimate is impractical as L_G will not be known a priori. A more pragmatic choice could be

$$s_t = A \sqrt{\Gamma(\lambda^1)\sigma} \frac{1}{\|\nabla_\lambda G(\lambda^t)\|_\infty} \frac{1}{\sqrt{t}} = A \sqrt{\log n_{\max}} \frac{1}{\|\nabla_\lambda G(\lambda^t)\|_\infty} \frac{1}{\sqrt{t}},$$

where A is a constant. It can be shown (Ben-Tal et al., 2001) that even for this step-size an efficiency estimate, similar to the one given in Theorem 3.1, is valid.

3.2.2 A SKETCH OF MD-BASED ALGORITHM

We are now ready to state an MD procedure for computing G. Given a sub-gradient by the oracle, and a suitably chosen step-size, one needs to compute a projection (step 3 in (12)) to complete one iteration of MD. Owing to the clever choice of prox-function, the projection step in our case is very easy to calculate and has an analytical expression given by:

$$\nabla\Phi(\lambda)_{jk} = \left(\ln(\lambda_{jk}) + 1\right),$$

$$\nabla\Phi^*(\tilde{\lambda})_{jk} = \left(\frac{e^{\tilde{\lambda}_{jk}}}{\sum_{l=1}^{n_j} e^{\tilde{\lambda}_{jl}}} \right).$$

The final MD procedure for minimizing G now reads:

Algorithm 1:

Require: $\lambda^1 \in \left\{ \bigotimes_{1 \leqslant j \leqslant n} \Delta_{n_j} \right\}$

repeat

$(\alpha^*, \gamma^*) \leftarrow \text{argmax}_{\alpha \in S_m, \gamma \in \Delta_{n,q^*}} f_{\lambda^t}(\alpha, \gamma)$ (Oracle computation)

$\tilde{\lambda}_{jk}^{t+1} \leftarrow (\nabla\Phi(\lambda^t) - s_t G'(\lambda))_{jk} = \left(\ln(\lambda_{jk}^t) + 1\right) + s_t {\alpha^*}^T \frac{Q_{jk}}{\gamma_j^*} \alpha^*$ (Descent Direction)

$\lambda_{jk}^{t+1} \leftarrow \nabla\Phi^*\left(\tilde{\lambda}^{t+1}\right) = \left(e^{\tilde{\lambda}_{jk}^{t+1}} / \sum_{k=1}^{n_j} e^{\tilde{\lambda}_{jk}^{t+1}} \right)$ (Projection step)

until convergence

By virtue of Theorem 3.1 (and using bound on Lipschitz constant derived in Appendix) this algorithm obtains an ε accurate minimizer of G in $O(n^{2+\frac{2}{q^*}} \log n_{\max}/\varepsilon^2)$ steps. Note that in practice the number of groups n (intuitively, the number of feature descriptors) is never high (typically < 10) and infact one can assume it to be $O(1)$; in which case the number of iterations will be nearly-independent of the number of kernels! The cost of each iteration depends on how efficiently one can maximize $f_\lambda(\alpha, \gamma)$ as a function of α, γ for a fixed λ. Note that gradient computation (that is, maximizing f) is the predominant computation in the mirror-descent based algorithm as the projection and step-size can be computed very easily from the analytical expressions presented above. On passing, we also note that there exist efficient projection algorithms for l_1-l_∞ regularization (Quattoni et al., 2009). In the next section we show that maximizing f can be achieved by solving a series of SVMs.

Again note that in the special case $n = 1$, where **VSKL**$_q$ (for any q) is equivalent to **MKL**, maximizing f is nothing but solving an SVM problem (with effective kernel computed with current weights). Since the per-step computation, in this special case, is predominantly that of solving an SVM (the projection and step-size computations are negligible) and the number of iterations is nearly-independent of the number of kernels, the proposed MD based algorithm is expected to perform far better than traditional reduced (projected) gradient based MKL solvers like `simpleMKL`. Also, in this case, the no. iterations is $O\left(\log n_{\max}/\varepsilon^2\right)$ and $n_{\max} = n_{\text{tot}}$ where n_{tot} is the total number of kernels. Cost of computing the effective kernel at each step depends on the sparsity of λ_j

however a conservative estimate gives $O(m^2 n_{\text{tot}})$ and projection, step size computations are $O(n_{\text{tot}})$ (negligible). Assuming the SVM problem can be solved in $O(m^2)$ time, we have the following complexity bound in case $n = 1$: $O\left(m^2 n_{\text{tot}} \log n_{\text{tot}}/\varepsilon^2\right)$. Also, in the case $q = 1$, the optimal value of γ_j is 1 for all j and hence maximizing f again corresponding to solving an SVM with effective kernel as canonical (equal-weight) sum of all the active kernels in each group. Again, in this case, the overall complexity is $O\left(m^2 n_{\text{tot}} \log n_{\max}/\varepsilon^2\right)$. The next section presents an efficient iterative scheme for maximizing f in a general case (that is, $n > 1, q > 1$).

3.3 Computing the Oracle

The joint maximization in (α, γ) of f_λ in the case $q = \infty$ can be posed as a Quadratically Constrained Quadratic Program (QCQP):

$$\begin{aligned} \max_{\alpha \in S_m, \gamma \in \Delta_n} f_\lambda(\gamma, \alpha) &= \mathbf{1}^T \alpha - \frac{1}{2} \alpha^T \left[\sum_{j=1}^{n} \left(\frac{\sum_{k=1}^{n_j} \lambda_{jk} \mathbf{Q}_{jk}}{\gamma_j} \right) \right] \alpha \\ &= \left. \begin{aligned} &\max_{\alpha \in S_m, \gamma \in \Delta_n, v} \mathbf{1}^T \alpha - \sum_{j=1}^{n} v_j \\ &\text{s.t.} \quad 2\gamma_j v_j \geqslant \alpha^T \left[\sum_{k=1}^{n_j} \lambda_{jk} \mathbf{Q}_{jk} \right] \alpha \ \ \forall j \end{aligned} \right\}. \end{aligned} \tag{13}$$

Using the identity

$$2\gamma_j v_j = \frac{1}{2}(\gamma_j + v_j)^2 - \frac{1}{2}(\gamma_j - v_j)^2,$$

the constraint in problem (13) becomes

$$\alpha^T \left[\sum_k \lambda_{jk} Q_{jk} \right] \alpha + \frac{1}{2}(\gamma_j - v_j)^2 \leq \frac{1}{2}(\gamma_j + v_j)^2,$$

and consequently problem (13) is a *conic quadratic* (CQ) *problem*.

A CQ problem can be solved with efficient solvers like $^{\text{TM}}$Mosek. However for an arbitrary norm, $q > 1$, such a formulation may not be possible and, even for $q = \infty$, very large-scale problems may require a more efficient algorithm. To this end we consider leveraging SVM solvers. Note that for each fixed value of γ one needs to solve an SVM problem in α. Moreover there exist *closed form solutions* when f is maximized over γ for fixed α. Such a Block Coordinate Ascent (BCA) (Tseng, 2001) procedure in general may not lead to convergence, but for the problem at hand we will show that the algorithm does indeed converge to a global maximum.

3.3.1 BLOCK COORDINATE ASCENT

In this section we describe a convergent and efficient algorithm based on the Block Coordinate Ascent (BCA) method. As a consequence of Lemma 2.2 the following is true

Proposition 3.3 *For a fixed* λ, α *the problem*

$$max_{\gamma \in \Delta_{n,q^*}} f_\lambda(\alpha, \gamma)$$

is optimized at

$$\gamma_i = \frac{D_i^{\frac{1}{q^*+1}}}{\left(\sum_{j=1}^n D_j^{\frac{q^*}{q^*+1}}\right)^{\frac{1}{q^*}}} \; i = 1, \ldots, n.$$

If $q = 1$ (that is, $q^* = \infty$), optimality is achieved at $\gamma_i = 1$ iff $D_i > 0$ where $D_j = \sum_{k=1}^{n_j} \lambda_{jk} \alpha^\top Q_{jk} \alpha$.

Proof Recall that

$$\max_{\alpha \in S_m, \gamma \in \Delta_{n,q^*}} f_\lambda(\alpha, \gamma) = \max_{\alpha \in S_m} \alpha^\top e - \frac{1}{2} \min_{\gamma \in \Delta_{n,q^*}} \sum_{j=1}^n \frac{D_j}{\gamma_j},$$

where $D_j = \sum_{k=1}^{n_j} \lambda_{jk} \alpha^\top Q_{jk} \alpha$. For a fixed α, the optimal γ is obtained by

$$\min_{\gamma \in \Delta_{n,q^*}} \sum_{j=1}^n \frac{D_j}{\gamma_j}.$$

The claim follows from Lemma 2.2. ■

This Proposition shows that one can use analytical expressions for γ when maximizing f_λ for a fixed α. Alternatively for a fixed γ, maximizing f_λ is equivalent to solving an SVM. These observations motivate the following algorithm for Oracle computation:

Algorithm 2:

Require: $\gamma^1 \in \Delta_{n,q^*}$
  repeat
    Compute $\alpha^{k+1} = \underset{\alpha \in S_m}{\operatorname{argmax}} \{f_\lambda(\alpha, \gamma^k)\}$ using SVM solver
    Compute $\gamma^{k+1} = \underset{\gamma \in \Delta_{n,q^*}}{\operatorname{argmax}} \{f_\lambda(\alpha^{k+1}, \gamma)\}$ by Proposition 3.3
  until convergence

In the following subsection we establish the convergence of this algorithm.

3.3.2 CONVERGENCE OF BCA ALGORITHM

We begin by introducing some propositions.

Definition 3.1 *We say that $z = (\alpha, \gamma)$ is a strict coordinate-wise maximum point of f over $A \times \Gamma$ if $z \in A \times \Gamma$ and*

$$f(\alpha', \gamma) < f(z) \qquad \forall \alpha' \in A,$$
$$f(\alpha, \gamma') < f(z) \qquad \forall \gamma' \in \Gamma.$$

Lemma 3.2 *Assume that A and Γ are convex sets, and f is a continuously differentiable function over $A \times \Gamma$. If z is a strict coordinate-wise maximum point of f over $A \times \Gamma$, then z is a local maximum point of f over $A \times \Gamma$.*

Proof Let $\alpha' \in A$, then $\forall u \in [0,1], u\alpha + (1-u)\alpha' \in A$ since A is convex. Let us consider $g(u) = f((1-u)\alpha + u\alpha', \gamma)$. g is differentiable and, since z is a strict coordinate-wise maximum point of f over $A \times \Gamma$, then $\forall u \in (0,1], g(0) > g(u)$, and this implies that $g'(0) < 0$, that is:

$$g'(0) = (\alpha' - \alpha)^T \nabla_\alpha f(\alpha, \gamma) < 0 \quad \forall \alpha' \in A, \; \alpha' \neq \alpha.$$

Following the same reasoning for γ, the following statement holds

$$\begin{aligned} &\nabla_\alpha f(\alpha,\gamma)^T(\alpha'-\alpha) < 0 \quad \forall \alpha' \in A,\ \alpha' \neq \alpha, \\ &\nabla_\gamma f(\alpha,\gamma)^T(\gamma'-\gamma) < 0 \quad \forall \gamma' \in \Gamma,\ \gamma' \neq \gamma. \end{aligned} \tag{14}$$

Now, by Taylor expansion,

$$f(\alpha',\gamma') = f(\alpha,\gamma) + \nabla_\alpha f(\alpha,\gamma)^T(\alpha'-\alpha) + \nabla_\gamma f(\alpha,\gamma)^T(\gamma'-\gamma) + O\left(\|\alpha-\alpha'\| + \|\gamma-\gamma'\|\right).$$

Using (14) we see that if (α',γ') is close enough to (α,γ), then $f(\alpha',\gamma') < f(\alpha,\gamma)$. ■

Proposition 3.4 *The BCA procedure (alg. 2) when applied to $f_\lambda(\alpha,\gamma)$ with respect to the blocks α and γ converge to a coordinate-wise maximum point of f_λ.*

Proof We begin by arguing that f_λ is bounded when Q_{jk} are p.d in the interior of simplex defined by γ, that is, $\gamma_j > 0$. Recall that at optimality, γ always lie in the interior for any $q > 1$. Hence for $q > 1$ we can as well restrict our search space to the interior of the simplex. For all such γ we have

$$f_\lambda(\alpha,\gamma) \leq \sum_{i=1}^{m}(\alpha_i - \frac{\tilde{\mu}}{2}\alpha_i^2),$$

where $\tilde{\mu} = \mu(\sum_{j=1}^{n}\gamma_j^{-1})$ and $\mu > 0$ is the greatest lower bound over all minimal eigenvalues of $Q_{j,k}$ matrices. For $q = 1$ case one can apply the above upper bound with $\gamma_i = 1$. Next, consider the following result.

Lemma 3.3 *f_λ is hemivariate over $S_m \times \Delta_n$.*

Proof Recall that a function f_λ is called hemivariate if it is not constant on any line segment of $S_m \times \Delta_n$. We proceed by contradiction. Let us assume that there exist $(\tilde{\alpha}^1,\tilde{\gamma}^1) \in S_m \times \Delta_n$ and $(\tilde{\alpha}^2,\tilde{\gamma}^2) \in S_m \times \Delta_n$ such that $\forall t \in [0,1]$, the following hold

$$g(t) = f_\lambda(t\tilde{\alpha}^1 + (1-t)\tilde{\alpha}^2, t\tilde{\gamma}^1 + (1-t)\tilde{\gamma}^2) = \text{ a constant.}$$

Then, $\forall t \in (0,1)$

$$\dot{g}(t) \equiv \frac{dg}{dt} = B_0 + \sum_j \frac{B_j}{(t + \frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2 - \tilde{\gamma}_j^1})^2} = 0, \tag{15}$$

where

$$B_j = \frac{1}{\tilde{\gamma}_j^2 - \tilde{\gamma}_j^{1\,3}}\left[\tilde{\gamma}_j^2\tilde{\alpha}^1 - \tilde{\gamma}_j^1\tilde{\alpha}^2\right]^T Q_j \left[\tilde{\gamma}_j^2\tilde{\alpha}^1 - \tilde{\gamma}_j^1\tilde{\alpha}^2\right],$$

$$Q_j = \sum_{k=1}^{n_j}\lambda_{jk}\mathbf{Q}_{jk},$$

and

$$B_0 = e^\top(\tilde{\alpha}^2 - \tilde{\alpha}^1) - \frac{1}{2}(\tilde{\alpha}^1 - \tilde{\alpha}^2)^\top \sum_{j=1}^{n}\frac{Q_j}{\tilde{\gamma}_j^2 - \tilde{\gamma}_j^1}(\tilde{\alpha}^1 - \tilde{\alpha}^2).$$

$\dot{g}(t)$ is a rational function of t and is 0 on $(0,1)$. This is possible if and only if $B_0 = 0$ and $\sum_j \frac{B_j}{(t+\frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2-\tilde{\gamma}_j^1})^2} = 0$. To establish this recall that the higher order derivatives of g are also 0. This leads in particular to:

$$\sum_j \frac{B_j}{(t+\frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2-\tilde{\gamma}_j^1})^3} = 0.$$

Let us now consider the sets $\Theta = \left\{ s \in \mathbb{R} \,\middle|\, \exists j, \frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2-\tilde{\gamma}_j^1} = s \right\}$ and $\Omega_s = \left\{ j \in \mathbb{N} \,\middle|\, \frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2-\tilde{\gamma}_j^1} = s \right\}$. We have

$$\sum_j \frac{B_j}{(t+\frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2-\tilde{\gamma}_j^1})^3} = \sum_{s\in\Theta} \sum_{j\in\Omega_s} \frac{B_j}{(t+s)^3}.$$

The family of $\{(t+s)^3\}, s \in \mathbb{R}$ is linearly independent, then, $\forall s \in \Theta, \sum_{j\in\Omega_s} \frac{B_j}{(t+s)^2} = 0$ by (15), and since $s = \frac{\tilde{\gamma}_j^1}{\tilde{\gamma}_j^2-\tilde{\gamma}_j^1}$ and $\forall j, \tilde{\gamma}_j^1 > 0$, then, $\text{sign}(\tilde{\gamma}_j^2 - \tilde{\gamma}_j^1)$ is constant over $\{j \in \Omega_s\}$. We know that Q_j is positive definite, thus, signB_j is constant over $\{j \in \Omega_s\}$. This implies that $\forall j, B_j = 0$. The positiveness of Q_j implies that this is possible only if $\forall j, \tilde{\gamma}_j^2\tilde{\alpha}^1 - \tilde{\gamma}_j^1\tilde{\alpha}^2 = 0$, which is equivalent to $\forall (j,i), (\tilde{\gamma}_j^2\tilde{\alpha}_i^1)^{q^*} = (\tilde{\gamma}_j^1\tilde{\alpha}_i^2)^{q^*}$ and summing over j, $\tilde{\gamma}^2$ and $\tilde{\gamma}^2$ belonging to Δ_{n,q^*}, we obtain $\tilde{\alpha}_1 = \tilde{\alpha}_2$ and then $\tilde{\gamma}^1 = \tilde{\gamma}^2$. Hence, f_λ is hemivariate and, this proves as well that f_λ is strictly concave. ■

We continue now the proof of Proposition 3.4. Let us consider a sequence z^p such that $z^{2p} = (\alpha^{p+1}, \gamma^p)$ and $z^{2p+1} = (\alpha^{p+1}, \gamma^{p+1})$. Since, by definition of our algorithm, $f_\lambda(z^{p+1}) \geqslant f_\lambda(z^p)$, and f_λ is bounded over $S_m \times \Delta_n$, then $f_\lambda(z^p)$ converges. Moreover, $S_m \times \Delta_n$ is compact in $\mathbb{R}^{m+n}$, so by passing to a subsequence if necessary, we can assume that $z^{2\phi(p)+1}$ converges to some z_1. Next we show that z^{2p+1} has a unique cluster point.

First we show that if $z^{2\phi(p)}$ converges to a cluster point z_1 of z^p, so does $z^{2\phi(p)+1}$. Indeed, if not, then $z^{2\phi(p)+1}$ has another cluster point than say $z_2 \neq z_1$).Therefore, we can assume that $\exists\tilde{\phi}$, a subsequence of $\phi(p)$ such that $z^{2\tilde{\phi}(p)+1}$ converges to z_2. Since $f_\lambda(z^p)$ converges, we have

$$\lim_{p\to\infty} f_\lambda(z^{2\tilde{\phi}(p)}) = \lim_{p\to\infty} f_\lambda(z^{2\tilde{\phi}(p)}).$$

Fix any $u \in [0,1]$ and denote $\tilde{z}^p = z^{2\tilde{\phi}(p)} + u(z^{2\tilde{\phi}(p)+1} - z^{2\tilde{\phi}(p)})$. We notice that $\tilde{z}^p \in S_m \times \Delta_n$. It is obvious that $\tilde{z}^p$ converges to $(1-u)z_1 + uz_2$. Since, f_λ is jointly concave with regard to (α,γ), we have

$$f_\lambda(\tilde{z}^p) \geqslant (1-u)f_\lambda(z^{2\tilde{\phi}(p)}) + uf_\lambda(z^{2\tilde{\phi}(p)+1}),$$

and by passing to the limit,

$$f_\lambda(\tilde{z}) \geqslant (1-u)f_\lambda(z_1) + uf_\lambda(z_2).$$

We cannot have $\forall\lambda \in [0,1], f_\lambda(\tilde{z}) = (1-u)f_\lambda(z_1) + uf_\lambda(z_2)$ because f_λ is hemivariate. Hence,

$$\exists\lambda \mid f_\lambda(\tilde{z}) > (1-u)f_\lambda(z_1) + uf_\lambda(z_2). \tag{16}$$

Since: $f(z^{2\tilde{\phi}(p)+1}) = \max_{\gamma\in\Delta_{n,q}}\{f(\alpha^{\tilde{\phi}(p)+1}, \gamma, \lambda)\}$, the following statement holds:

$$\forall\gamma \in \Delta_n, f_\lambda(z^{2\tilde{\phi}(p)+1}) \geqslant f_\lambda(\alpha^{\tilde{\phi}(p)+1}, \gamma),$$

and since $z^{2\tilde{\phi}(p)}, \tilde{z}^p$ and $z^{2\tilde{\phi}(p)+1}$ differ only in their second coordinate block γ, we have $f_\lambda(\tilde{z}^p) \leqslant (1-u)f_\lambda(z^{2\tilde{\phi}(p)}) + uf_\lambda(z^{2\tilde{\phi}(p)+1})$, and by passing to the limit, $f_\lambda(\tilde{z}) \leqslant (1-u)f_\lambda(z_1) + uf_\lambda(z_2)$ which contradicts (16). Hence, $z_1 = z_2$. We showed that $z^{2\phi(p)+1}$ has a unique cluster point z_1, hence it converges to z_1. We next prove that z_1 is a coordinate-wise maximum point of f_λ. Recall that

$$\forall \gamma \in \Delta_n, f_\lambda(x^{2\tilde{\phi}(p)+1}) \geqslant f_\lambda(\alpha^{\tilde{\phi}(p)+1}, \gamma).$$

Passing to the limit, we have:

$$\forall \gamma \in \Delta_n, f_\lambda(z_1) \geqslant f_\lambda(\alpha(z_1), \gamma), \tag{17}$$

where $\alpha(z_1) = \alpha^{\tilde{\phi}(\infty)+1}$. The same reasoning with regard to α shows that

$$\forall \alpha \in S_m, f_\lambda(x_1) \geqslant f_\lambda(\alpha, \gamma(x_1)), \tag{18}$$

where $\gamma(z_1) = \gamma^{\tilde{\phi}(\infty)+1}$. This shows that z_1 is a coordinate-wise maximum point of f_λ and, according to (3.2), z_1 is a local maximum of f_λ over $S_m \times \Delta_n$ and since f_λ is strictly concave outside the line where $\alpha^1\gamma^2 = \alpha^2\gamma^1$, and since f_λ is not constant on any of these lines, z_1 is the unique global maximum of f_λl; hence strict inequalities hold in (17) and (18). ■

Now that the mirror-descent as well as the block coordinate ascent procedures are presented and the respective convergences are proved, we now proceed to present the overall algorithm for solving the dual (D).

3.4 The `mirrorVSKL` Procedure

This section presents the `mirrorVSKL` algorithm for solving the dual (D):

Algorithm 3: `mirrorVSKL`

Require: $\lambda^1 \in \left\{ \bigotimes_{1 \leqslant j \leqslant n} \Delta_{n_j} \right\}$

repeat

$(\alpha^*, \gamma^*) \leftarrow \underset{\alpha \in S_m, \gamma \in \Delta_n}{\operatorname{argmax}} f(\alpha, \gamma, \lambda^t)$ (Use BCA in Alg. 2)

$\tilde{\lambda}_{jk}^{t+1} \leftarrow (\nabla\Phi(\lambda^t) - s_t G'(\lambda))_{jk} = \left(\ln(\lambda_{jk}^t) + 1\right) + s_t \alpha^{*T} \frac{Q_{jk}}{\gamma_j^*} \alpha^*$ (Descent Direction)

$\lambda_{jk}^{t+1} \leftarrow \nabla\Phi^*\left(\tilde{\lambda}^{t+1}\right) = \left(e^{\tilde{\lambda}_{jk}^{t+1}} / \sum_{k=1}^{n_j} e^{\tilde{\lambda}_{jk}^{t+1}} \right)$ (Projection step)

until convergence

The algorithm converges to the optimal of (D) for arbitrary $q \geq 1$. The per-step complexity in the mirror-descent iterations now depends on the number of iterations of the BCA algorithm. However it was observed in practice (see Section 4) that for the values of n encountered, the BCA converges in 2-4 iterations and hence can be assumed to be a constant. With this assumption, even in the general case ($n > 1, q > 1$), the computational complexity of `mirrorVSKL` remains to be $O\left(m^2 n_{\text{tot}} \log n_{\max} / \varepsilon^2\right)$. We conclude this section with the following note: convergence of the mirror descent algorithm is based on the fact that sub-gradients are exactly computable. However in `mirrorVSKL`, the sub-gradients are computed using an oracle numerically and hence is approximate. Convergence analysis with such approximate sub-gradients is non-trivial and a research problem in itself. The work by D'Aspermont (2008) is a good starting point for this.

4. Numerical Experiments

This section presents results of simulations which prove the suitability of employing the proposed VSKL formulations for multi-modal tasks like object categorization. Experimental results which demonstrate the scalability of the `mirrorVSKL` algorithm in solving the traditional block l_1 regularization based MKL formulation are also presented.

4.1 Performance on Object Categorization Data Sets

The experimental results summarized in this section aim at proving the suitability of employing the proposed VSKL formulations for tasks like object categorization. The following benchmark data sets were used in our experiments:

Caltech-101 (Fei-Fei et al., 2004) Collection of 9144 images[7] from 102 categories of objects like faces, watches, ants etc. The minimum, average and maximum number of images per category are 31, 90, 800 respectively.

Caltech-256 (Griffin et al., 2007) Collection of 30607 images[8] from 257 categories of objects. The minimum, average and maximum number of images per category are 80, 119, 827 respectively.

Oxford flowers (Nilsback and Zisserman, 2006) Collection of images of 17 varieties of flowers.[9] The number of images per category is 80.

Following the strategy of Vedaldi et al. (2009), the following four feature descriptors[10] were employed in the case of the Caltech data sets:

1. Geometric blur (Zhang et al., 2006; Berg et al., 2005). These descriptors are initially computed at representative points of the image. Later, the distance between two images is obtained as the average distance of nearest descriptor pairs.

2. PHOW gray/color (Lazebnik et al., 2006). SIFT features are computed densely on a regular grid and quantized in 300 visual words. Spatial histogram with 4×4 subdivisions are then formed. The color variant concatenates SIFT descriptors computed on the HSV channels.

3. Self-similarity (Shechtman and Irani, 2007). Similar to the PHOW features, descriptors are quantized in 300 visual words, and a spatial histogram of size 4×4.

In case of the Oxford flowers data set, the seven feature descriptors employed in Nilsback and Zisserman (2006, 2008) are used here.[11]

Each feature descriptor mentioned above, describes the image in terms of few feature values. As mentioned previously, it was observed in the literature (see Nilsback and Zisserman, 2006) that employing feature values obtained from various descriptors simultaneously is beneficial for object

7. Available at `http://www.vision.caltech.edu/Image_Datasets/Caltech101`.
8. Available at `http://www.vision.caltech.edu/Image_Datasets/Caltech256`.
9. Available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/17flowers.tgz`.
10. Software available at `http://www.robots.ox.ac.uk/~vgg/software/MKL/v1.0/index.html`.
11. Corresponding distance matrices are available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html`.

categorization; however not all of the features obtained using a feature descriptor may be useful. The state-of-the-art performance on these data sets is achieved by a methodology which generates kernels using each of the feature descriptors and then chooses the best among them using the framework of MKL (Varma and Ray, 2007; Nilsback and Zisserman, 2008). The MKL formulation employed in Varma and Ray (2007) Nilsback and Zisserman (2008) is equivalent to the traditional block l_1 regularization based MKL formulation[12] (henceforth denoted by **MKL**). Hence here we compare the performance of VSKL formulations with that of **MKL**. As a baseline we also compare performance with an SVM classifier built using the kernel as sum of all the given kernels (henceforth denoted by **SVM**).

From each feature descriptor, five kernels were generated by varying the width-parameter of the Gaussian kernel (from 10^{-4} to 1 on a log-scale). Since the resulting kernels are naturally grouped according to the descriptor they were generated from and also it is true that each feature descriptor is critical (may not be equally critical) for good categorization, it is obvious to employ the proposed VSKL formulations by assuming kernels are grouped according to descriptors generating them. Thus, in case of the Caltech data sets, $n = 4$ and $n_j = 5 \; \forall \; j$ and in case of Oxford flowers data set, $n = 7$ and $n_j = 5 \; \forall \; j$. Note that **SVM** and $\mathbf{VSKL}_1$ differ exactly in the way the kernels are grouped: for $\mathbf{VSKL}_1$ the kernels are grouped by their generating feature descriptors whereas for **SVM** each group is characterized by a single kernel (that is, for **SVM** $n = 20, n_j = 1 \; \forall \; j$ in case of Caltech data set and $n = 35, n_j = 1 \; \forall \; j$ in case of Oxford flowers data set).

In order that the experimental results are comparable to others in literature, we followed the usual practice of generating training and test sets, in case of each data set, using a fixed number of images from each object category and repeating the experiments with different random selections of images. For the Caltech-101, Caltech-256 and Oxford flowers data sets we have used 15, 25, 60 images per object category as training images and 15, 15, 20 images per object category as testing images respectively. The hyper-parameters of the various formulations were tuned using suitable cross-validation procedures. In case of the Caltech-101 data set, the accuracies reported are the test-set accuracies with the tuned set of hyper-parameters, averaged over 10 randomly sampled training and test splits. Since the Caltech-256 data set has large number of classes and the experiments are computationally intensive, the results are reported only for a single split. In case of Oxford flowers data set, the accuracies are averaged over the 3 standard data splits provided with the source images.[13] Also, we employ the 1-vs-rest methodology in order to handle the multi-class problems arising in these data sets. Table 1 reports the average testset accuracies achieved with the various kernel learning techniques. The numbers in brackets appearing below each accuracy indicate the total number of SVM calls made for solving the corresponding formulation[14] and throw light on the trade-off between accuracy and computation. In addition to comparison with **SVM** and **MKL**, we also report results of comparison with the CKL formulations (Szafranski et al., 2008), which also assume kernels are grouped. Note that the CKL formulations were not previously applied to object categorization and we wish to compare them here with VSKL in order to stress on the need for solving (1) for the cases $q \geq 1$. Recall that if $q < 1$ then (1) can be solved using the wrapper

12. The formulation employed by Varma and Ray (2007) and Nilsback and Zisserman (2008) also has additional constraints for including prior information regarding weights of kernels. Since such constraints lead to independent improvements with all MKL formulations, the experiments here compare MKL formulations without the additional constraints.
13. Available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/datasplits.mat`.
14. Stopping criterion was choosen same across different methods.

VSKL$_q$				**MKL**	**SVM**	**CKL**$_{1,2q}$		
$q=1$	$q=2$	$q=3$	$q=\infty$			$q=0.75$	$q=0.85$	$q=0.99$
Caltech-101								
66.44 (50)	67.03 (79)	67.06 (70)	**67.07** (65)	54.05 (24)	64.61 (1)	65.74 (36)	64.35 (34)	63.21 (36)
Caltech-256								
32.06 (100)	34.71 (201)	35.39 (188)	**36.69** (151)	21.07 (21)	34.04 (1)	34.43 (33)	34.40 (32)	34.43 (34)
Oxford								
85.59 (41)	85.69 (70)	85.69 (68)	85.29 (64)	85.49 (121)	85.98 (1)	**86.08** (54)	**86.08** (50)	**86.08** (51)

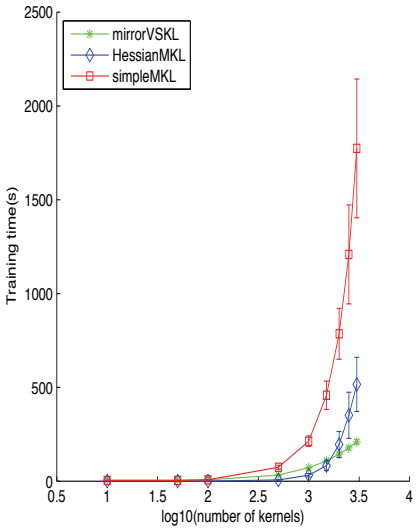
Table 1: Comparison of average testset accuracies achieved by the various formulations

approaches of Szafranski et al. (2008). Also recall the notation that formulation in (1) for $q \geq 1$ corresponds to **VSKL**$_q$ and for $q < 1$ corresponds to **CKL**$_{1,2q}$. The results clearly indicate that the proposed methodology is suitable for object categorization tasks and its performance better than state-of-the-art in case of the Caltech data sets; whereas in case of Oxford data set, the performance is comparable to state-of-the-art. Also, in case of oxford flowers data set, the performance of all the methods is more or less the same. Another important observation, which is especially evident in case of the Caltech-256 data set, is that the performance of VSKL depends on the parameter q and hence it is important to solve the VSKL formulation efficiently for various values of q. This demonstrates the usefulness of the proposed `mirrorVSKL` algorithm, which efficiently solves the formulation at various values of $q \geq 1$. Automatic tuning of q is indeed an open question and calls for further research. Lastly, we note that the accuracies with **SVM** and **VSKL**$_1$, which differ in the way the kernels are grouped, are noticeably different—which is expected.

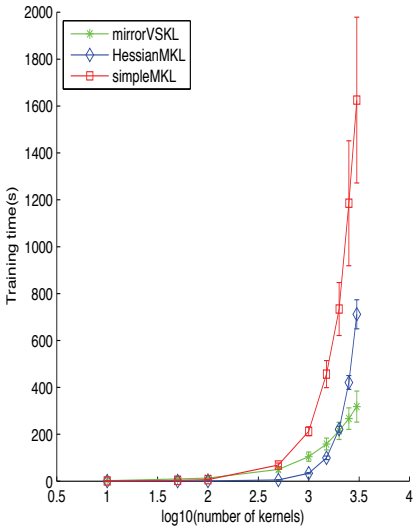
4.2 Scalability Experiments

This section presents results comparing scalability of `mirrorVSKL`,[15] `SimpleMKL`[16] and Hessian-MKL[17] in solving the **MKL** formulation. Note that all these algorithms solve an SVM problem at each step and hence are comparable. For fairness in comparison, the SVM problem arising at each step was solved using the same solver in case of all the three algorithms. The stopping criteria employed in all cases was relative difference in objective value being less than 10^{-4} (that is, $(f_{old} - f_{new})/f_{old} < 10^{-4}$. The evaluation was made on four data sets from the UCI repository (Blake and Merz, 1998): Liver, Wpbc, Ionosphere and Sonar. Following the experimental set-up of Rakotomamonjy et al. (2008), each data set was split into training and test sets using 70% and 30% data points respectively. For each data set, kernels were generated based on individual features using different width parameters for the Gaussian kernel. Figure 1 compares the average time[18] taken for solving the formulation (this excludes time taken for building kernels) over 20 different random training-test splits as a function of the number of kernels. The value of regulariza-

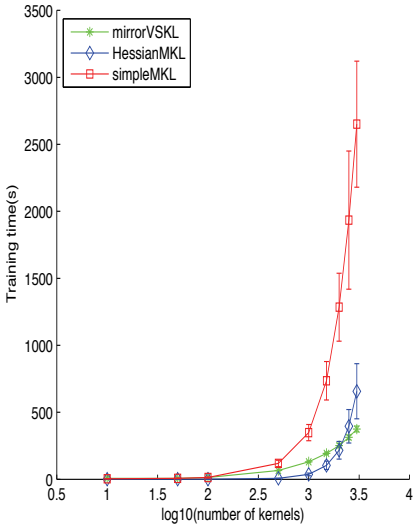
15. Code available at `http://mllab.csa.iisc.ernet.in/vskl.html`.
16. Code available at `http://asi.insa-rouen.fr/enseignants/~arakotom/code/mklindex.html`.
17. Code available at `http://www.chapelle.cc/olivier/ams/`.
18. The standard deviation in the time taken is also shown using vertical bars at each point in the plot.



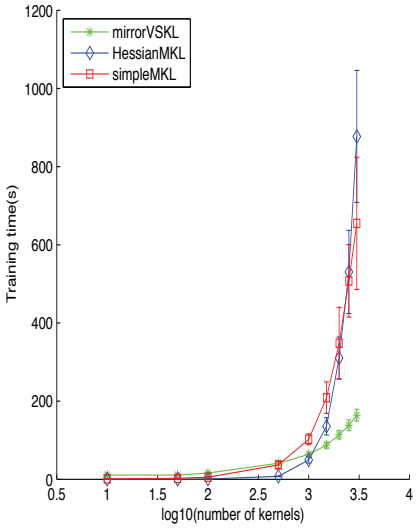
(a) Sonar



(b) Liver



(c) Ionosphere



(d) Wpbc

Figure 1: Plots of average time (in secs) taken by various solvers

tion parameter C was fixed at 1000 in all cases. The figure clearly shows that `mirrorVSKL` scales better than `simpleMKL` and `HessianMKL`. When large number of candidate kernels are available, `mirrorVSKL` outperforms them in terms of computational performance. In some cases, the solving time with the proposed method is as low as around $1/8$ of that with `simpleMKL` and around $1/6$ of that with `HessianMKL`!

The proposed algorithm scales better than `simpleMKL` primarily because of two reasons: firstly, the per-step auxiliary problem in case of the proposed algorithm has an analytical solution and the step-size also can be chosen very easily. Hence the predominant computation at every step is only that of computing the gradient (that is, solving the SVM). However, in case of `simpleMKL`, the reduced gradient needs to be computed and moreover the step-size needs to be computed using a 1-d line search (which may further involve solving few SVMs). Also, in case of `HessianMKL`, the per-step cost is high mainly due to the second order computations. Secondly, the number of iterations in solving the formulation is nearly-independent of the number of kernels in case of the proposed MD based algorithm. However no such statement can be made in case of either `simpleMKL` or `HessianMKL`.

In order to get a better insight, the number of SVM calls made by `simpleMKL` and `mirrorVSKL` (both of which are first order methods and hence comparable wrt. number of iterations/SVM calls) are compared in Figure 2. It is interesting to see that the number of SVM calls more or less remains a low value in case of `mirrorVSKL`; whereas it shoots up steeply in case of `simpleMKL`. The fact that the number of SVMs calls is low also implies that `mirrorVSKL` scales better than `simpleMKL` even wrt. no. of examples and hence is ideal for applications with large data sets as well as large number of candidate kernels.

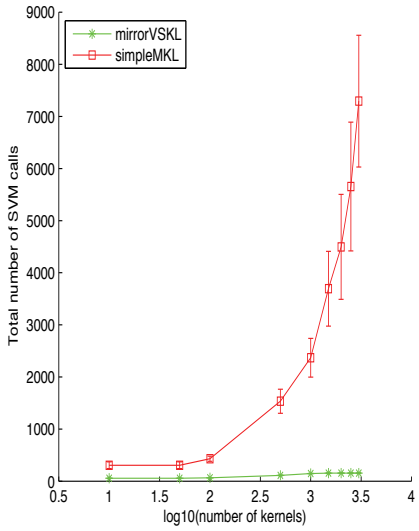
Also, it was observed that the number of iterations required by the BCA algorithm to converge (with various values of q) was typically very small. In case of all data sets, the maximum number of iterations for convergence of BCA was 4 iterations. Hence the number of iterations required by the BCA algorithm can be assumed to be a constant and the computational complexity bound $O\left(m^2 n_{\text{tot}} \log n_{\max}/\varepsilon^2\right)$ indeed is valid.

5. Conclusions

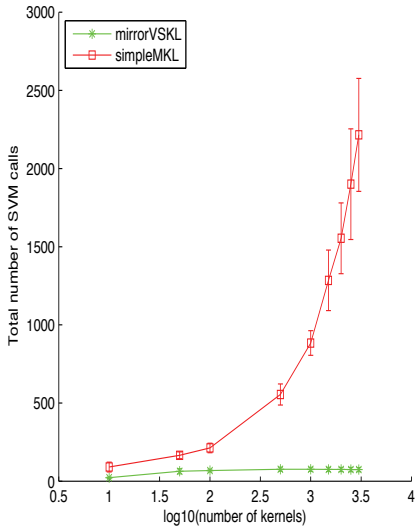
This paper makes two important contributions to the MKL literature: a) a specific mixed-norm regularization based MKL formulation which is well-suited for object categorization and other multi-modal tasks is studied. b) An efficient mirror-descent based algorithm for solving the new formulation is proposed. Since the traditional MKL formulation can be realized as a special of the proposed formulation, the efficient algorithm is also of independent interest to the MKL community. A detailed proof of convergence of the algorithm was also presented. Empirical results show that the new formulation achieves far better generalization than state-of-the-art object categorization techniques. Scaling experiments show that the mirror-descent based algorithm outperforms traditional gradient descent based approaches. In some cases the proposed MD based algorithm achieved a 8 times speed-up over `simpleMKL`!

Acknowledgments

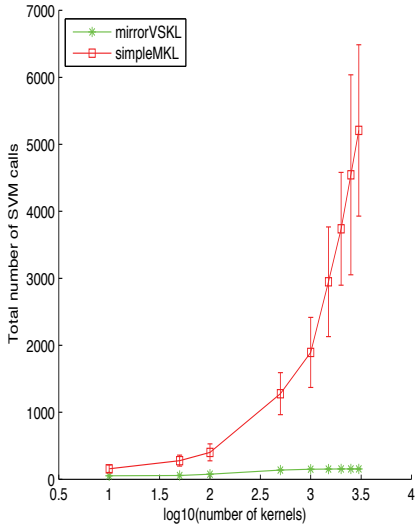
We would like to thank Dinesh (previously at IISc, Bangalore and currently at Bell Labs, Bangalore) for his help with the initial version of the paper and the scripts for running the object categorization experiments. AB was supported by the US-Israel Binational Science Foundation. Part of the research was done while AB was a Visiting Professor at the Center for Mathematics and computer Science (CWI) in Amsterdam. CB was partially supported by the Yahoo! faculty award.



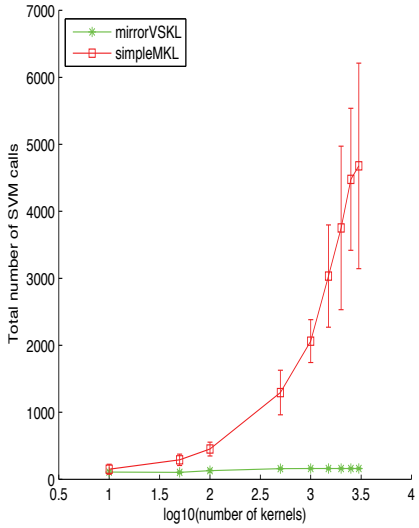
(a) Sonar



(b) Liver



(c) Ionosphere



(d) Wpbc

Figure 2: Plots of average number of SVM calls with `simpleMKL` and `mirrorVSKL`

Appendix A.

In this section we prove proposition 3.1, which says that G is convex and Lipschitz continuous under a mild regularity condition—all the eigenvalues of the given gram-matrices are finite and non-zero:

Proof The convexity of G follows from the fact that it is point-wise maximum over functions of the form

$$f(\alpha,\gamma,\lambda) = \mathbf{1}^T\alpha - \frac{1}{2}\alpha^T\left[\sum_{j=1}^{n}\left(\frac{\sum_{k=1}^{n_j}\lambda_{jk}\mathbf{Q}_{jk}}{\gamma_j}\right)\right]\alpha,$$

which are linear w.r.t λ. A sufficient condition for G to be Lipschitz continuous is the sub-gradient should be norm bounded. Define $D_{jk} = \alpha^{*\top}Q_{jk}\alpha^*$ where α^* and γ^* denote optimal values, that maximize $f_\lambda(\alpha,\gamma)$, for a given λ. From the definition of τ and μ we immediately have the following bound

$$\tau\mu\|\alpha^*\|_2^2 \le D_{jk} \le \mu\|\alpha^*\|_2^2.$$

The sub-gradient vector, evaluated at any λ, can be obtained by differentiating G at α^* and γ^*. The strategy would be to exploit the above limits on D_{jk} to bound the norm of the sub-gradient. To this end we eliminate γ in G (using proposition 3.3) and then examine the sub-gradient:

Case $q > 1$

$$\frac{\partial G}{\partial\lambda_{jk}} = \begin{cases} -\frac{1}{2}D_{jk}\left\{\frac{\sum_{j'}\left(\sum_{k'}\lambda_{j'k'}D_{j'k'}\right)^{\frac{q^*}{q^*+1}}}{\sum_{k'}\lambda_{jk'}D_{jk'}}\right\}^{\frac{1}{q^*}} & \text{if } \sum_{k'}\lambda_{jk'}D_{jk'} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Case $q = 1$

$$\frac{\partial G}{\partial\lambda_{jk}} = \begin{cases} -\frac{1}{2}D_{jk} & \text{if } \sum_{k'}\lambda_{jk'}D_{jk'} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

From these equations, it is easy to see that:

$$\left|\frac{\partial G}{\partial\lambda_{jk}}\right| \le \frac{1}{2}\left(\frac{n}{\tau}\right)^{\frac{1}{q^*}}\left(\mu\|\alpha^*\|_2^2\right)^{\frac{q^*(q^*+1)-1}{q^*(q^*+1)}}.$$

In case $q = 1$, we have $\left|\frac{\partial G}{\partial\lambda_{jk}}\right| \le \frac{1}{2}\mu\|\alpha^*\|_2^2$. Now, we know that $\alpha \in S_m \Rightarrow \alpha_i < C\ \forall\ i \Rightarrow \|\alpha^*\|_\infty \le C \Rightarrow \|\alpha^*\|_2 \le \sqrt{m_{sv}}C$ where m_{sv} is the number of support vectors. These relationships shows that $\|\nabla_\lambda G\|_\infty \le L_G$ where

$$L_G = \begin{cases} \frac{1}{2}\left(\frac{n}{\tau}\right)^{\frac{1}{q^*}}\left(\mu m_{sv}C^2\right)^{\frac{q^*(q^*+1)-1}{q^*(q^*+1)}} & \text{if } q > 1, \\ \frac{1}{2}\mu m_{sv}C^2 & \text{if } q = 1. \end{cases}$$

Now, since $\|\nabla_\lambda G\|_\infty$ is bounded, we have that G is Lipschitz continuous with respect to l_1 norm with Lipschitz constant L_G. ■

References

F. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the International Conference on Machine Learning*, 2004.

A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31:167–175, 2003.

A. Ben-Tal and A. Nemirovski. Lectures on modern convex optimization: Analysis, algorithms and engineering applications. *MPS/ SIAM Series on Optimization*, 1, 2001.

A. Ben-Tal, T. Margalit, and A. Nemirovski. The ordered subsets mirror descent optimization method with applications to tomography. *SIAM Journal of Optimization*, 12(1):79–108, 2001.

A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

C. Blake and C. Merz. Uci repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, 2006.

A. D'Aspermont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19(3):1171–1183, 2008.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Generative-Model Based Vision*, 2004.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report CNS-TR-2007-001, California Institute of Technology, 2007.

M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. Non-sparse regularization and efficient training with multiple kernels. Technical Report UCB/EECS-2010-21, University of California, Berkeley, 2010.

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.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bag of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

C. A. Micchelli and M. Pontil. Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

J. S. Nath, G. Dinesh, S. Raman, C. Bhattacharyya, A. Ben-Tal, and K. R. Ramakrishnan. On the algorithmics and applications of a mixed-norm regularization based kernel learning formulation. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

M-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006.

M-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

A. Quattoni, X. Carreras, M. Collins, and T. Darrell. An efficient projection for $l_{1,\infty}$ regularization. In *Proceedings of the International Conference on Machine Learning*, 2009.

A. Rakotomamonjy, F. Bach, S. Canu, and Y Grandvalet. Simplemkl. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

R. T. Rockafellar. Minimax theorems and conjugate saddle-functions. *Mathematica Scandinava*, 14:151–173, 1964.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

M. Sion. On general minimax theorem. *Pacific Journal of Mathematics*, 1958.

S. Sonnenburg, G. Ratsch, C. Schafer, and B. Scholkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, 2006.

M. Szafranski, Y. Grandvalet, and A. Rakotomamonjy. Composite kernel learning. In *Proceedings of the International Conference on Machine Learning*, 2008.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimisation. *Journal of Optimization Theory and Applications*, 2001.

V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

M. Varma and D. Ray. Learning the discriminative power invariance trade-off. In *Proceedings of the International Conference on Computer Vision*, 2007.

A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proceedings of the International Conference on Computer Vision*, 2009.

H. Zhang, A. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classication for visual category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2126–2136, 2006.

Journal of Machine Learning Research 12 (2011) 593-625 Submitted 6/10; Revised 9/10; Published 2/11

Regression on Fixed-Rank Positive Semidefinite Matrices: A Riemannian Approach

Gilles Meyer G.MEYER@ULG.AC.BE
Department of Electrical Engineering and Computer Science
University of Liège
B-4000 Liège, Belgium

Silvère Bonnabel SILVERE.BONNABEL@MINES-PARISTECH.FR
Robotics center
Mines ParisTech
Boulevard Saint-Michel, 60, 75272 Paris, France

Rodolphe Sepulchre R.SEPULCHRE@ULG.AC.BE
Department of Electrical Engineering and Computer Science
University of Liège
B-4000 Liège, Belgium

Editor: Inderjit Dhillon

Abstract

The paper addresses the problem of learning a regression model parameterized by a fixed-rank positive semidefinite matrix. The focus is on the nonlinear nature of the search space and on scalability to high-dimensional problems. The mathematical developments rely on the theory of gradient descent algorithms adapted to the Riemannian geometry that underlies the set of fixed-rank positive semidefinite matrices. In contrast with previous contributions in the literature, no restrictions are imposed on the range space of the learned matrix. The resulting algorithms maintain a linear complexity in the problem size and enjoy important invariance properties. We apply the proposed algorithms to the problem of learning a distance function parameterized by a positive semidefinite matrix. Good performance is observed on classical benchmarks.

Keywords: linear regression, positive semidefinite matrices, low-rank approximation, Riemannian geometry, gradient-based learning

1. Introduction

A fundamental problem of machine learning is the learning of a distance between data samples. When the distance can be written as a quadratic form (either in the data space (Mahalanobis distance) or in a kernel feature space (kernel distance)), the learning problem is a regression problem on the set of positive definite matrices. The regression problem is turned into the minimization of the prediction error, leading to an optimization framework and gradient-based algorithms.

The present paper focuses on the nonlinear nature of the search space. The classical framework of gradient-based learning can be generalized provided that the nonlinear search space is equipped with a proper Riemannian geometry. Adopting this general framework, we design novel learning algorithms on the space of fixed-rank positive semidefinite matrices, denoted by $S_+(r,d)$, where d

©2011 Gilles Meyer, Silvère Bonnabel and Rodolphe Sepulchre.

is the dimension of the matrix, and r is its rank. Learning a parametric model in $S_+(r,d)$ amounts to jointly learn a r-dimensional subspace and a quadratic distance in this subspace.

The framework is motivated by *low-rank learning* in large-scale applications. If the data space is of dimension d, the goal is to maintain a linear computational complexity $O(d)$. In contrast to the classical approach of first reducing the dimension of the data and then learning a distance in the reduced space, there is an obvious conceptual advantage to perform the two tasks simultaneously. If this objective can be achieved without increasing the numerical cost of the algorithm, the advantage becomes also practical.

Our approach makes use of two quotient geometries of the set $S_+(r,d)$ that have been recently studied by Journée et al. (2010) and Bonnabel and Sepulchre (2009). Making use of a general theory of line-search algorithms in quotient matrix spaces (Absil et al., 2008), we obtain concrete gradient updates that maintain the rank and the positivity of the learned model at each iteration. This is because the update is intrinsically constrained to belong to the nonlinear search space, in contrast to early learning algorithms that neglect the non linear nature of the search space in the update and impose the constraints a posteriori (Xing et al., 2002; Globerson and Roweis, 2005).

Not surprisingly, our approach has close connections with a number of recent contributions on learning algorithms. Learning problems over nonlinear matrix spaces include the learning of subspaces (Crammer, 2006; Warmuth, 2007), rotation matrices (Arora, 2009), and positive definite matrices (Tsuda et al., 2005). The space of (full-rank) positive definite matrices $S_+(d)$ is of particular interest since it coincides with our set of interest in the particular case $r = d$.

The use of Bregman divergences and alternating projection has been recently investigated for learning in $S_+(d)$. Tsuda et al. (2005) propose to use the *von Neumann* divergence, resulting in a generalization of the well-known AdaBoost algorithm (Schapire and Singer, 1999) to positive definite matrices. The use of the so-called *LogDet* divergence has also been investigated by Davis et al. (2007) in the context of Mahalanobis distance learning.

More recently, algorithmic work has focused on scalability in terms of dimensionality and data set size. A natural extension of the previous work on positive definite matrices is thus to consider low-rank positive semidefinite matrices. Indeed, whereas algorithms based on full-rank matrices scale as $O(d^3)$ and require $O(d^2)$ storage units, algorithms based on low-rank matrices scale as $O(dr^2)$ and require $O(dr)$ storage units (Fine et al., 2001; Bach and Jordan, 2005). This is a significant complexity reduction as the approximation rank r is typically very small compared to the dimension of the problem d.

Extending the work of Tsuda et al. (2005), Kulis et al. (2009) recently considered the learning of positive semidefinite matrices. The authors consider Bregman divergence measures that enjoy convexity properties and lead to updates that preserve the rank as well as the positive semidefinite property. However, these divergence-based algorithms intrinsically constrain the learning algorithm to a fixed range space. A practical limitation of this approach is that the subspace of the learned matrix is fixed beforehand by the initial condition of the algorithm.

The approach proposed in the present paper is in a sense more classical (we just perform a line-search in a Riemannian manifold) but we show how to interpret Bregman divergence based algorithms in our framework. This is potentially a contribution of independent interest since a general convergence theory exists for line-search algorithms on Riemannian manifolds. The generality of the proposed framework is of course motivated by the non-convex nature of the rank constraint.

The paper is organized as follows. Section 2 presents the general optimization framework of Riemannian learning. This framework is then applied to the learning of subspaces (Section 4),

positive definite matrices (Section 5) and fixed-rank positive semidefinite matrices (Section 6). The novel proposed algorithms are presented in Section 7. Section 8 discusses the relationship to existing work as well as extensions of the proposed approach. Applications are presented in Section 9 and experimental results are presented in Section 10.

2. Linear Regression on Riemannian Spaces

We consider the following standard regression problem. Given

(i) data points $\mathbf{X}$, in a linear data space $\mathcal{X} = \mathbb{R}^{d \times d}$,

(ii) observations y, in a linear output space $\mathcal{Y} = \mathbb{R}$, (or $\mathbb{R}^d$),

(iii) a regression model $\hat{y} = \hat{y}_{\mathbf{W}}(\mathbf{X})$ parameterized by a matrix $\mathbf{W}$ in a search space $\mathcal{W}$,

(iv) a quadratic loss function $\ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$,

find the optimal fit $\mathbf{W}^*$ that minimizes the *expected cost*

$$F(\mathbf{W}) = \mathbb{E}_{\mathbf{X},y}\{\ell(\hat{y}, y)\} = \int \ell(\hat{y}, y)\, dP(\mathbf{X}, y),$$

where $\ell(\hat{y}, y)$ penalizes the discrepancy between observations and predictions, and $P(\mathbf{X}, y)$ is the (unknown) joint probability distribution over data and observation pairs. Although our main interest will be in the scalar model

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\mathbf{X}),$$

the theory applies equally to vector data points $\mathbf{x} \in \mathbb{R}^d$, $\hat{y} = \mathrm{Tr}(\mathbf{W}\mathbf{x}\mathbf{x}^T) = \mathbf{x}^T\mathbf{W}\mathbf{x}$, to a regression model parameterized by a vector $\mathbf{w} \in \mathbb{R}^d$, $\hat{y} = \mathbf{w}^T\mathbf{x}$, or to a vector output space $\hat{y} = \mathbf{W}\mathbf{x}$.

As it is generally not possible to compute $F(\mathbf{W})$ explicitly, batch learning algorithms minimize instead the *empirical cost*

$$f_n(\mathbf{W}) = \frac{1}{2n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2, \tag{1}$$

which is the average loss computed over a finite number of samples $\{(\mathbf{X}_i, y_i)\}_{i=1}^n$.

Online learning algorithms (Bottou, 2004) consider possibly infinite sets of samples $\{(\mathbf{X}_t, y_t)\}_{t \geq 1}$, received one at a time. At time t, the online learning algorithm minimizes the instantaneous cost

$$f_t(\mathbf{W}) = \frac{1}{2}(\hat{y}_t - y_t)^2.$$

In the sequel, we only present online versions of algorithms to shorten the exposition. The single necessary change to convert an online algorithm into its batch counterpart is to perform, at each iteration, the minimization of the empirical cost f_n instead of the minimization of the instantaneous cost f_t. In the sequel, we denote by f the cost function that is minimized at each iteration.

Our focus will be on *nonlinear* search spaces $\mathcal{W}$. We only require $\mathcal{W}$ to have the structure of a Riemannian matrix manifold. Following Absil et al. (2008), an abstract gradient descent algorithm can then be derived based on the update formula

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}(-s_t\, \mathrm{grad} f(\mathbf{W}_t)). \tag{2}$$

The gradient $\text{grad} f(\mathbf{W}_t)$ is an element of the tangent space $T_{\mathbf{W}_t}\mathcal{W}$. The scalar $s_t > 0$ is the step size. The retraction $R_{\mathbf{W}_t}$ is a mapping from the tangent space $T_{\mathbf{W}_t}\mathcal{W}$ to the Riemannian manifold. Under mild conditions on the retraction R, the classical convergence theory of line-search algorithms in linear spaces generalizes to Riemannian manifolds (see Absil et al., 2008, Chapter 4).

Observe that the standard (online) learning algorithm for linear regression in $\mathbb{R}^d$,

$$\mathbf{w}_{t+1} = \mathbf{w}_t - s_t(\mathbf{w}_t^T\mathbf{x}_t - y_t)\mathbf{x}_t, \tag{3}$$

can be interpreted as a particular case of (2) for the linear model $\hat{y} = \mathbf{w}^T\mathbf{x}$ in the *linear* search space $\mathcal{W} = \mathbb{R}^d$. The Euclidean metric turns $\mathbb{R}^d$ in a (flat) Riemannian manifold. For a scalar function $f : \mathbb{R}^d \to \mathbb{R}$ of $\mathbf{w}$, the gradient satisfies

$$Df(\mathbf{w})[\boldsymbol{\delta}] = \boldsymbol{\delta}^T \text{grad} f(\mathbf{w}),$$

where $Df(\mathbf{w})[\boldsymbol{\delta}]$ is the directional derivative of f in the direction $\boldsymbol{\delta}$, and the natural retraction

$$R_{\mathbf{w}_t}(-s_t \text{ grad} f(\mathbf{w}_t)) = \mathbf{w}_t - s_t \text{ grad} f(\mathbf{w}_t),$$

induces a line-search along “straight lines" which are geodesics (that is paths of shortest length) in linear spaces. With $f(\mathbf{w}) = \frac{1}{2}(\mathbf{w}^T\mathbf{x} - y)^2$, one arrives at (3).

This example illustrates that the main ingredients to obtain a concrete algorithm are convenient formulas for the gradient and for the retraction mapping. This paper provides such formulas for three examples of nonlinear matrix search spaces: the Grassmann manifold (Section 4), the cone of positive definite matrices (Section 5), and the set of fixed-rank positive semidefinite matrices (Section 6). Each of those sets will be equipped with *quotient Riemannian geometries* that provide convenient formulas for the gradient and for the retractions. Line-search algorithms in quotient Riemannian spaces are discussed in detail in the book of Absil et al. (2008). For the readers convenience, basic concepts and notations are introduced in the next section.

3. Line-Search Algorithms on Matrix Manifolds

This section summarizes the exposition of Absil et al. (2008, Chapters 3 and 4).

Restrictions on the search space are generally encoded into optimization algorithms by means of particular constraints or penalties expressed as a function of the search variable. However, when the search space is endowed with a particular manifold structure, it is possible to design an exploration strategy that is consistent with the geometry of the problem and that appropriately turns the problem into an unconstrained optimization problem. This approach is the purpose of optimization algorithms defined on matrix manifolds.

Informally, a manifold $\mathcal{W}$ is a space endowed with a differentiable structure. One usually makes the distinction between embedded submanifolds (subsets of larger manifolds) and quotient manifolds (manifolds described by a set of equivalence classes). An intuitive example of embedded submanifold is the sphere embedded in $\mathbb{R}^d$. A typical example of quotient manifold is the set of r-dimensional subspaces in $\mathbb{R}^d$, viewed as a collection of r-dimensional orthogonal frames that cannot be superposed by a rotation. The rotational variants of a given frame thus define an equivalence class (denoted using square brackets $[\cdot]$), which is identified as a single point on the quotient manifold.

To develop line-search algorithms, the notion of gradient of a scalar cost function needs to be extended to manifolds. For that purpose, the manifold $\mathcal{W}$ is endowed with a metric $g_{\mathbf{W}}(\xi_{\mathbf{W}}, \zeta_{\mathbf{W}})$,

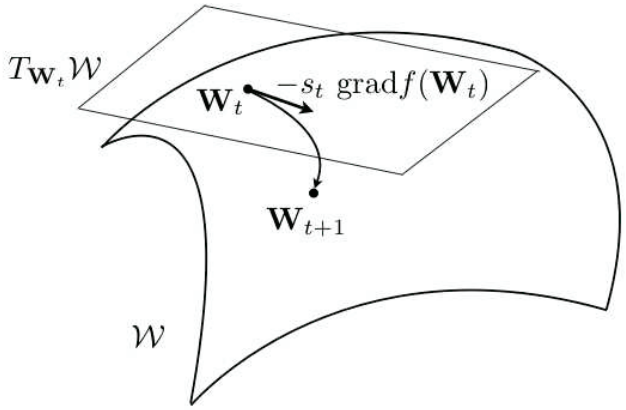


Figure 1: Gradient iteration on a Riemannian manifold. The search direction $-\mathrm{grad} f(\mathbf{W}_t)$ belongs to the tangent space $T_{\mathbf{W}_t}\mathcal{W}$. The updated point $\mathbf{W}_{t+1}$ automatically remains inside the manifold thanks to the retraction mapping.

which is an inner product defined between elements $\xi_{\mathbf{W}}, \zeta_{\mathbf{W}}$ of the tangent space $T_{\mathbf{W}}\mathcal{W}$ at $\mathbf{W}$. The metric induces a norm on the tangent space $T_{\mathbf{W}}\mathcal{W}$ at $\mathbf{W}$:

$$\|\xi_{\mathbf{W}}\|_{\mathbf{W}} = \sqrt{g_{\mathbf{W}}(\xi_{\mathbf{W}}, \xi_{\mathbf{W}})}.$$

The gradient of a smooth scalar function $f : \mathcal{W} \to \mathbb{R}$ at $\mathbf{W} \in \mathcal{W}$ is the only element $\mathrm{grad} f(\mathbf{W}) \in T_{\mathbf{W}}\mathcal{W}$ that satisfies

$$Df(\mathbf{W})[\boldsymbol{\Delta}] = g_{\mathbf{W}}(\boldsymbol{\Delta}, \mathrm{grad} f(\mathbf{W})), \quad \forall \boldsymbol{\Delta} \in T_{\mathbf{W}}\mathcal{W},$$

where $\boldsymbol{\Delta}$ is a matrix representation of a "geometric" tangent vectors ξ, and where

$$Df(\mathbf{W})[\boldsymbol{\Delta}] = \lim_{t \to 0} \frac{f(\mathbf{W} + t\boldsymbol{\Delta}) - f(\mathbf{W})}{t},$$

is the standard directional derivative of f at $\mathbf{W}$ in the direction $\boldsymbol{\Delta}$.

For quotient manifolds $\mathcal{W} = \overline{\mathcal{W}}/\sim$, where $\overline{\mathcal{W}}$ is the total space and $\sim$ is the equivalence relation that defines the quotient, the tangent space $T_{[\mathbf{W}]}\mathcal{W}$ at $[\mathbf{W}]$ is sufficiently described by the directions that do not induce any displacement in the set of equivalence classes $[\mathbf{W}]$. This is achieved by restricting the tangent space at $[\mathbf{W}]$ to horizontal vectors $\xi_{\mathbf{W}} \in T_{\mathbf{W}}\overline{\mathcal{W}}$ at $\mathbf{W}$ that are orthogonal to the equivalence class $[\mathbf{W}]$. Provided that the metric $g_{\mathbf{W}}$ in the total space is invariant along the equivalence classes, it defines a metric in the quotient space

$$g_{[\mathbf{W}]}(\xi_{[\mathbf{W}]}, \zeta_{[\mathbf{W}]}) \triangleq g_{\mathbf{W}}(\xi_{\mathbf{W}}, \zeta_{\mathbf{W}}).$$

The horizontal gradient $\overline{\mathrm{grad} f(\mathbf{W})}$ is obtained by projecting the gradient $\mathrm{grad} f(\mathbf{W})$ in the total space onto the set of horizontal vectors $\xi_{\mathbf{W}}$ at $\mathbf{W}$.

Natural displacements at $\mathbf{W}$ in a direction $\xi_{\mathbf{W}}$ on the manifold are performed by following geodesics (paths of shortest length on the manifold) starting from $\mathbf{W}$ and tangent to $\xi_{\mathbf{W}}$. This is

performed by means of the exponential mapping

$$\mathbf{W}_{t+1} = \mathrm{Exp}_{\mathbf{W}_t}(s_t \xi_{\mathbf{W}_t}),$$

which induces a line-search algorithm along geodesics.

A more general update formula is obtained if we relax the constraint of moving along geodesics. The retraction mapping

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}(s_t \xi_{\mathbf{W}_t}),$$

locally approximates the exponential mapping. It provides an attractive alternative to the exponential mapping in the design of optimization algorithms on manifolds, as it reduces the computational complexity of the update while retaining the essential properties that ensure convergence results. When $\xi_{\mathbf{W}_t}$ coincide with $-\mathrm{grad} f(\mathbf{W}_t)$ a gradient descent algorithm on the manifold is obtained. Figure 1 pictures a gradient descent update on $\mathcal{W}$.

4. Linear Regression on the Grassmann Manifold

As a preparatory step to Section 6, we review the online subspace learning (Oja, 1992; Crammer, 2006; Warmuth, 2007) in the present framework. Let $\mathcal{X} = \mathcal{Y} = \mathbb{R}^d$, and consider the linear model

$$\hat{\mathbf{y}} = \mathbf{U}\mathbf{U}^T\mathbf{x},$$

with $\mathbf{U} \in \mathrm{St}(r,d) = \{\mathbf{U} \in \mathbb{R}^{d\times r} \text{ s.t. } \mathbf{U}^T\mathbf{U} = \mathbf{I}\}$, the Stiefel manifold of r-dimensional orthonormal bases in $\mathbb{R}^d$. The quadratic loss is then

$$f(\mathbf{U}) = \ell(\hat{\mathbf{y}}, \mathbf{x}) = \frac{1}{2}\|\hat{\mathbf{y}} - \mathbf{x}\|_2^2 = \frac{1}{2}\|\mathbf{U}\mathbf{U}^T\mathbf{x} - \mathbf{x}\|_2^2. \tag{4}$$

Because the cost (4) is invariant by orthogonal transformation $\mathbf{U} \mapsto \mathbf{U}\mathbf{O}$, $\mathbf{O} \in O(r)$, where $O(r) = \mathrm{St}(r,r)$ is the orthogonal group, the search space is in fact a set of equivalence classes

$$[\mathbf{U}] = \{\mathbf{U}\mathbf{O} \text{ s.t. } \mathbf{O} \in O(r)\}.$$

This set is denoted by $\mathrm{St}(r,d)/O(r)$. It is a *quotient representation* of the set of r-dimensional subspaces in $\mathbb{R}^d$, that is, the Grassmann manifold $\mathrm{Gr}(r,d)$. The quotient geometries of $\mathrm{Gr}(r,d)$ have been well studied (Edelman et al., 1998; Absil et al., 2004). The metric

$$g_{[\mathbf{U}]}(\xi_{[\mathbf{U}]}, \zeta_{[\mathbf{U}]}) \triangleq g_{\mathbf{U}}(\xi_{\mathbf{U}}, \zeta_{\mathbf{U}}),$$

is induced by the standard metric in $\mathbb{R}^{d\times r}$,

$$g_{\mathbf{U}}(\mathbf{\Delta}_1, \mathbf{\Delta}_2) = \mathrm{Tr}(\mathbf{\Delta}_1^T\mathbf{\Delta}_2),$$

which is invariant along the fibers, that is, equivalence classes. Tangent vectors $\xi_{[\mathbf{U}]}$ at $[\mathbf{U}]$ are represented by horizontal tangent vectors $\xi_{\mathbf{U}}$ at $\mathbf{U}$:

$$\xi_{\mathbf{U}} = \Pi_{\mathbf{U}}\mathbf{\Delta} = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\mathbf{\Delta}, \quad \mathbf{\Delta} \in \mathbb{R}^{d\times r}.$$

Therefore, the gradient admits the simple horizontal representation

$$\overline{\mathrm{grad} f(\mathbf{U})} = \Pi_{\mathbf{U}}\, \mathrm{grad} f(\mathbf{U}), \tag{5}$$

where $\operatorname{grad} f(\mathbf{U})$ is defined by the identity

$$Df(\mathbf{U})[\boldsymbol{\Delta}] = g_{\mathbf{U}}(\boldsymbol{\Delta}, \operatorname{grad} f(\mathbf{U})).$$

A standard retraction in $\mathrm{Gr}(r,d)$ is the exponential mapping, that induces a line-search along geodesics. The exponential map has the closed-form expression

$$\mathrm{Exp}_{\mathbf{U}}(\xi_{\mathbf{U}}) = \mathbf{U}\mathbf{V}\cos(\boldsymbol{\Sigma})\mathbf{V}^T + \mathbf{Z}\sin(\boldsymbol{\Sigma})\mathbf{V}^T, \tag{6}$$

which is obtained from a singular value decomposition of the horizontal vector $\xi_{\mathbf{U}} = \mathbf{Z}\boldsymbol{\Sigma}\mathbf{V}^T$. Following Absil et al. (2004), an alternative convenient retraction in $\mathrm{Gr}(r,d)$ is given by

$$R_{\mathbf{U}}(s\xi_{\mathbf{U}}) = [\mathbf{U} + s\xi_{\mathbf{U}}] = \mathrm{qf}(\mathbf{U} + s\xi_{\mathbf{U}}), \tag{7}$$

where $\mathrm{qf}(\cdot)$ is a function that extracts the orthogonal factor of the QR-decomposition of its argument. A possible advantage of the retraction (7) over the retraction (6) is that, in contrast to the SVD computation, the QR decomposition is computed in a fixed number $O(dr^2)$ of arithmetic operations.

With the formulas (5) and (7) applied to the cost function (4), the abstract update (2) becomes

$$\mathbf{U}_{t+1} = \mathrm{qf}(\mathbf{U}_t + s_t(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathbf{x}_t\mathbf{x}_t^T\mathbf{U}_t),$$

which is Oja's update for subspace tracking (Oja, 1992).

5. Linear Regression on the Cone of Positive Definite Matrices

The learning of a full-rank positive definite matrix is recast as follows. Let $\mathcal{X} = \mathbb{R}^{d\times d}$ and $\mathcal{Y} = \mathbb{R}$, and consider the model

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\mathbf{X}),$$

with $\mathbf{W} \in S_+(d) = \{\mathbf{W} \in \mathbb{R}^{d\times d} \text{ s.t. } \mathbf{W} = \mathbf{W}^T \succ 0\}$. Since $\mathbf{W}$ is symmetric, only the symmetric part of $\mathbf{X}$ will contribute to the trace. The previous model is thus equivalent to

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\mathrm{Sym}(\mathbf{X})),$$

where $\mathrm{Sym}(\cdot)$ extract the symmetric part of its argument, that is, $\mathrm{Sym}(\mathbf{B}) = (\mathbf{B}^T + \mathbf{B})/2$. The quadratic loss is

$$f(\mathbf{W}) = \ell(\hat{y}, y) = \frac{1}{2}(\mathrm{Tr}(\mathbf{W}\mathrm{Sym}(\mathbf{X})) - y)^2.$$

The quotient geometries of $S_+(d)$ are rooted in the matrix factorization

$$\mathbf{W} = \mathbf{G}\mathbf{G}^T, \quad \mathbf{G} \in \mathrm{GL}(d),$$

where $\mathrm{GL}(d)$ is the set of all invertible $d \times d$ matrices. Because the factorization is invariant by rotation, $\mathbf{G} \mapsto \mathbf{G}\mathbf{O}$, $\mathbf{O} \in O(d)$, the search space is once again identified to the quotient

$$S_+(d) \simeq \mathrm{GL}(d)/O(d),$$

which represents the set of equivalence classes

$$[\mathbf{G}] = \{\mathbf{G}\mathbf{O} \text{ s.t. } \mathbf{O} \in O(d)\}.$$

We will equip this quotient with two meaningful Riemannian metrics.

5.1 A Flat Metric on $S_+(d)$

The metric on the quotient $\mathrm{GL}(d)/O(d)$:

$$g_{[\mathbf{G}]}(\xi_{[\mathbf{G}]},\zeta_{[\mathbf{G}]}) \triangleq g_{\mathbf{G}}(\xi_{\mathbf{G}},\zeta_{\mathbf{G}}),$$

is induced by the standard metric in $\mathbb{R}^{d\times d}$,

$$g_{\mathbf{G}}(\mathbf{\Delta}_1,\mathbf{\Delta}_2) = \mathrm{Tr}(\mathbf{\Delta}_1^T\mathbf{\Delta}_2),$$

which is invariant by rotation along the set of equivalence classes. As a consequence, it induces a metric $g_{[\mathbf{G}]}$ on $S_+(d)$. With this geometry, a tangent vector $\xi_{[\mathbf{G}]}$ at $[\mathbf{G}]$ is represented by a horizontal tangent vector $\xi_{\mathbf{G}}$ at $\mathbf{G}$ by

$$\xi_{\mathbf{G}} = \mathrm{Sym}(\mathbf{\Delta})\mathbf{G}, \quad \mathbf{\Delta}\in\mathbb{R}^{d\times d}.$$

The horizontal gradient of

$$f(\mathbf{G}) = \ell(\hat{y},y) = \frac{1}{2}(\mathrm{Tr}(\mathbf{G}\mathbf{G}^T\mathrm{Sym}(\mathbf{X})) - y)^2, \tag{8}$$

is the unique horizontal vector $\overline{\mathrm{grad}\, f(\mathbf{G})}$ that satisfies

$$Df(\mathbf{G})[\mathbf{\Delta}] = g_{\mathbf{G}}(\mathbf{\Delta}, \overline{\mathrm{grad}\, f(\mathbf{G})}).$$

Elementary computations yield

$$\overline{\mathrm{grad}\, f(\mathbf{G})} = 2(\hat{y}-y)\mathrm{Sym}(\mathbf{X})\mathbf{G}.$$

Since the metric is flat, geodesics are straight lines and the exponential mapping is

$$\mathrm{Exp}_{\mathbf{G}}(\xi_{\mathbf{G}}) = [\mathbf{G}+\xi_{\mathbf{G}}] = \mathbf{G}+\xi_{\mathbf{G}}.$$

Those formulas applied to the cost (8) turns the abstract update (2) into the simple formula

$$\mathbf{G}_{t+1} = \mathbf{G}_t - 2s_t(\hat{y}_t - y_t)\mathrm{Sym}(\mathbf{X}_t)\mathbf{G}_t, \tag{9}$$

for an online gradient algorithm and

$$\mathbf{G}_{t+1} = \mathbf{G}_t - 2s_t\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)\mathrm{Sym}(\mathbf{X}_i)\mathbf{G}_t, \tag{10}$$

for a batch gradient algorithm.

5.2 The Affine-Invariant Metric on $S_+(d)$

Because $S_+(d)\simeq \mathrm{GL}(d)/O(d)$ is the quotient of two Lie groups, its (reductive) geometric structure can be further exploited (Faraut and Koranyi, 1994). Indeed the group $\mathrm{GL}(d)$ has a natural action on $S_+(d)$ via the transformation $\mathbf{W}\mapsto \mathbf{A}\mathbf{W}\mathbf{A}^T$ for any $\mathbf{A}\in\mathrm{GL}(d)$. The affine-invariant metric admits interesting invariance properties to these transformations. To build such an affine-invariant metric, the metric at identity

$$g_{\mathbf{I}}(\xi_{\mathbf{I}},\zeta_{\mathbf{I}}) = \mathrm{Tr}(\xi_{\mathbf{I}}\zeta_{\mathbf{I}}),$$

is extended to the entire space to satisfy the invariance property

$$g_{\mathbf{I}}(\xi_{\mathbf{I}},\zeta_{\mathbf{I}}) = g_{\mathbf{W}}(\mathbf{W}^{\frac{1}{2}}\xi_{\mathbf{I}}\mathbf{W}^{\frac{1}{2}},\mathbf{W}^{\frac{1}{2}}\zeta_{\mathbf{I}}\mathbf{W}^{\frac{1}{2}}) = g_{\mathbf{W}}(\xi_{\mathbf{W}},\zeta_{\mathbf{W}}).$$

The resulting metric on $S_+(d)$ is defined by

$$g_{\mathbf{W}}(\xi_{\mathbf{W}},\zeta_{\mathbf{W}}) = \mathrm{Tr}(\xi_{\mathbf{W}}\mathbf{W}^{-1}\zeta_{\mathbf{W}}\mathbf{W}^{-1}). \tag{11}$$

The affine-invariant geometry of $S_+(d)$ has been well studied, in particular in the context of information geometry (Smith, 2005). Indeed, any positive definite matrix $\mathbf{W} \in S_+(d)$ can be identified to the multivariate normal distribution of zero mean $\mathcal{N}(0,\mathbf{W})$, whose probability density is $p(\mathbf{z};\mathbf{W}) = \frac{1}{Z}\exp(-\frac{1}{2}\mathbf{z}^T\mathbf{W}^{-1}\mathbf{z})$, where Z is a normalizing constant. Using such a metric allows to endow the space of parameters $S_+(d)$ with a distance that reflects the proximity of the probability distributions. The Riemannian metric thus distorts the Euclidean distances between positive definite matrices in order to reflect the amount of information between the two associated probability distributions. If $\xi_{\mathbf{W}}$ is a tangent vector to $\mathbf{W} \in S_+(d)$, we have the following approximation for the Kullback-Leibler divergence (up to third order terms)

$$D_{KL}(p(\mathbf{z};\mathbf{W})||p(\mathbf{z};\mathbf{W}+\xi_{\mathbf{W}})) \approx \frac{1}{2}\, g^{FIM}_{\mathbf{W}}(\xi_{\mathbf{W}},\xi_{\mathbf{W}}) = \frac{1}{2}\, g_{\mathbf{W}}(\xi_{\mathbf{W}},\xi_{\mathbf{W}}),$$

where $g^{FIM}_{\mathbf{W}}$ is the well-known Fisher information metric at $\mathbf{W}$, which coincides with the affine-invariant metric (11) (Smith, 2005). With this geometry, tangent vectors $\xi_{\mathbf{W}}$ are expressed as

$$\xi_{\mathbf{W}} = \mathbf{W}^{\frac{1}{2}}\mathrm{Sym}(\mathbf{\Delta})\mathbf{W}^{\frac{1}{2}}, \quad \mathbf{\Delta} \in \mathbb{R}^{d\times d}.$$

The gradient $\mathrm{grad} f(\mathbf{W})$ is given by

$$Df(\mathbf{W})[\mathbf{\Delta}] = g_{\mathbf{W}}(\mathbf{\Delta}, \mathrm{grad} f(\mathbf{W})).$$

Applying this formula to (5) yields

$$\mathrm{grad} f(\mathbf{W}) = (\hat{y}-y)\mathbf{W}\mathrm{Sym}(\mathbf{X})\mathbf{W}. \tag{12}$$

The exponential mapping has the closed-form expression

$$\mathrm{Exp}_{\mathbf{W}}(\xi_{\mathbf{W}}) = \mathbf{W}^{\frac{1}{2}}\exp(\mathbf{W}^{-\frac{1}{2}}\xi_{\mathbf{W}}\mathbf{W}^{-\frac{1}{2}})\mathbf{W}^{\frac{1}{2}}. \tag{13}$$

Its first-order approximation provides the convenient retraction

$$R_{\mathbf{W}}(s\xi_{\mathbf{W}}) = \mathbf{W} - s\xi_{\mathbf{W}}. \tag{14}$$

The formulas (12) and (13) applied to the cost (5) turn the abstract update (2) into

$$\mathbf{W}_{t+1} = \mathbf{W}_t^{\frac{1}{2}}\exp(-s_t(\hat{y}_t-y_t)\mathbf{W}_t^{\frac{1}{2}}\mathrm{Sym}(\mathbf{X}_t)\mathbf{W}_t^{\frac{1}{2}})\mathbf{W}_t^{\frac{1}{2}}.$$

With the alternative retraction (14), the update becomes

$$\mathbf{W}_{t+1} = \mathbf{W}_t - s_t(\hat{y}_t-y_t)\mathbf{W}_t\mathrm{Sym}(\mathbf{X}_t)\mathbf{W}_t,$$

which is the update of Davis et al. (2007) based on the LogDet divergence (see Section 8.1).

5.3 The Log-Euclidean Metric on $S_+(d)$

For the sake of completeness, we briefly review a third Riemannian geometry of $S_+(d)$, that exploits the property

$$\mathbf{W} = \exp(\mathbf{S}), \quad \mathbf{S} = \mathbf{S}^T \in \mathbb{R}^{d\times d}.$$

The matrix exponential thus provides a global diffeomorphism between $S_+(d)$ and the linear space of $d \times d$ symmetric matrices. This geometry is studied in detail in the paper (Arsigny et al., 2007). The cost function

$$f(\mathbf{S}) = \ell(\hat{y}, y) = \frac{1}{2}(\mathrm{Tr}(\exp(\mathbf{S})\mathrm{Sym}(\mathbf{X})) - y)^2,$$

thus defines a cost function in the linear space of symmetric matrices. The gradient of this cost function is given by

$$\mathrm{grad} f(\mathbf{S}) = (\hat{y}_t - y_t)\mathrm{Sym}(\mathbf{X}_t),$$

and the retraction is

$$R_{\mathbf{S}}(s\xi_{\mathbf{S}}) = \exp(\log \mathbf{W} + s\xi_{\mathbf{S}}).$$

The corresponding gradient descent update is

$$\mathbf{W}_{t+1} = \exp(\log \mathbf{W}_t - s_t(\hat{y}_t - y_t)\mathrm{Sym}(\mathbf{X}_t)),$$

which is the update of Tsuda et al. (2005) based on the von Neumann divergence.

6. Linear Regression on Fixed-Rank Positive Semidefinite Matrices

We now present the proposed generalizations to fixed-rank positive semidefinite matrices.

6.1 Linear Regression with a Flat Geometry

The generalization of the results of Section 5.1 to the set $S_+(r,d)$ is a straightforward consequence of the factorization

$$\mathbf{W} = \mathbf{G}\mathbf{G}^T, \quad \mathbf{G} \in \mathbb{R}^{d\times r}_*,$$

where $\mathbb{R}^{d\times r}_* = \{\mathbf{G} \in \mathbb{R}^{d\times r} \text{ s.t. } \det(\mathbf{G}^T\mathbf{G}) \neq 0\}$. Indeed, the flat quotient geometry of the manifold $S_+(d) \simeq \mathrm{GL}(d)/O(d)$ is generalized to the quotient geometry of $S_+(r,d) \simeq \mathbb{R}^{d\times r}_*/O(r)$ by a mere adaptation of matrix dimension, leading to the updates (9) and (10) for matrices $\mathbf{G}_t \in \mathbb{R}^{d\times r}_*$. The mathematical derivation of these updates is a straight application of the material presented in the paper of Journée et al. (2010), where the quotient geometry of $S_+(r,d) \simeq \mathbb{R}^{d\times r}_*/O(r)$ is studied in details. In the next section, we propose an alternative geometry that jointly learns a r-dimensional subspace and a full-rank quadratic model in this subspace.

6.2 Linear Regression with a Polar Geometry

In contrast to the flat geometry, the affine-invariant geometry of $S_+(d) \simeq \mathrm{GL}(d)/O(d)$ does not generalize directly to $S_+(r,d) \simeq \mathbb{R}^{d\times r}_*/O(r)$ because $\mathbb{R}^{d\times r}_*$ is not a group. However, a generalization is possible by considering the polar matrix factorization

$$\mathbf{G} = \mathbf{U}\mathbf{R}, \quad \mathbf{U} \in \mathrm{St}(r,d),\ \mathbf{R} \in S_+(r).$$

It is obtained from the singular value decomposition of $\mathbf{G} = \mathbf{Z}\mathbf{\Sigma}\mathbf{V}^T$ as $\mathbf{U} = \mathbf{Z}\mathbf{V}^T$ and $\mathbf{R} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T$ (Golub and Van Loan, 1996). This gives a polar parameterization of $S_+(r,d)$

$$\mathbf{W} = \mathbf{U}\mathbf{R}^2\mathbf{U}^T.$$

This development leads to the quotient representation

$$S_+(r,d) \simeq (\mathrm{St}(r,d) \times S_+(r))/O(r), \tag{15}$$

based on the invariance of $\mathbf{W}$ to the transformation $(\mathbf{U},\mathbf{R}^2) \mapsto (\mathbf{U}\mathbf{O},\mathbf{O}^T\mathbf{R}^2\mathbf{O})$, $\mathbf{O} \in O(r)$. It thus describes the set of equivalence classes

$$[(\mathbf{U},\mathbf{R}^2)] = \{(\mathbf{U}\mathbf{O},\mathbf{O}^T\mathbf{R}^2\mathbf{O}) \text{ s.t. } \mathbf{O} \in O(r)\}.$$

The cost function is now given by

$$f(\mathbf{U},\mathbf{R}^2) = \ell(\hat{y},y) = \frac{1}{2}(\mathrm{Tr}(\mathbf{U}\mathbf{R}^2\mathbf{U}^T\mathrm{Sym}(\mathbf{X})) - y)^2. \tag{16}$$

The Riemannian geometry of (15) has been recently studied by Bonnabel and Sepulchre (2009). A tangent vector $\xi_{[\mathbf{W}]} = (\xi_{\mathbf{U}},\xi_{\mathbf{R}^2})_{[\mathbf{U},\mathbf{R}^2]}$ at $[(\mathbf{U},\mathbf{R}^2)]$ is described by a horizontal tangent vector $\xi_{\mathbf{W}} = (\xi_{\mathbf{U}},\xi_{\mathbf{R}^2})_{(\mathbf{U},\mathbf{R}^2)}$ at $(\mathbf{U},\mathbf{R}^2)$ by

$$\xi_{\mathbf{U}} = \Pi_{\mathbf{U}}\mathbf{\Delta},\ \mathbf{\Delta} \in \mathbb{R}^{d\times r}, \qquad \xi_{\mathbf{R}^2} = \mathbf{R}\mathrm{Sym}(\mathbf{\Psi})\mathbf{R},\ \mathbf{\Psi} \in \mathbb{R}^{r\times r}.$$

The metric

$$\begin{aligned} g_{[\mathbf{W}]}(\xi_{[\mathbf{W}]},\zeta_{[\mathbf{W}]}) &\triangleq g_{\mathbf{W}}(\xi_{\mathbf{W}},\zeta_{\mathbf{W}}) \\ &= \frac{1}{\lambda}\, g_{\mathbf{U}}(\xi_{\mathbf{U}},\zeta_{\mathbf{U}}) + \frac{1}{1-\lambda}\, g_{\mathbf{R}^2}(\xi_{\mathbf{R}^2},\zeta_{\mathbf{R}^2}), \end{aligned} \tag{17}$$

where $\lambda \in (0,1)$, is induced by the metric of $\mathrm{St}(r,d)$ and the affine-invariant metric of $S_+(r)$,

$$g_{\mathbf{U}}(\mathbf{\Delta}_1,\mathbf{\Delta}_2) = \mathrm{Tr}(\mathbf{\Delta}_1^T\mathbf{\Delta}_2), \qquad g_{\mathbf{R}^2}(\mathbf{\Psi}_1,\mathbf{\Psi}_2) = \mathrm{Tr}(\mathbf{\Psi}_1\mathbf{R}^{-2}\mathbf{\Psi}_2\mathbf{R}^{-2}).$$

The proposed metric is invariant along the set of equivalence classes and thus induces a quotient structure on $S_+(r,d)$. Alternative metrics on $S_+(r)$ can be considered as long as the metric remains invariant along the set of equivalence classes. For instance, the log-Euclidean metric discussed in Section 5.3 would qualify as a valid alternative.

A retraction is provided by distinct retractions on $\mathbf{U}$ and $\mathbf{R}^2$,

$$R_{\mathbf{U}}(s\xi_{\mathbf{U}}) = \mathrm{qf}(\mathbf{U} + s\xi_{\mathbf{U}}) \tag{18}$$

$$R_{\mathbf{R}^2}(s\xi_{\mathbf{R}^2}) = \mathbf{R}\exp(s\mathbf{R}^{-1}\xi_{\mathbf{R}^2}\mathbf{R}^{-1})\mathbf{R}. \tag{19}$$

One should observe that this retraction is not the exponential mapping of $S_+(r,d)$. This illustrates the interest of considering more general retractions than the exponential mapping. Indeed, as discussed in the paper of Bonnabel and Sepulchre (2009), the geodesics (and therefore the exponential

Batch regression	Online regression
Input: $\{(\mathbf{X}_i, y_i)\}_{i=1}^n$	**Input:** $\{(\mathbf{X}_t, y_t)\}_{t\geq 1}$
Require: $\mathbf{G}_0$ or $(\mathbf{U}_0, \mathbf{R}_0), \lambda$	**Require:** $\mathbf{G}_0$ or $(\mathbf{U}_0, \mathbf{R}_0), \lambda, p, s, t_0, T$
$t = 0$	$t = 0, \texttt{count} = p$
repeat	**while** $t \leq T$ **do**
	if $\texttt{count} > 0$ **then**
	Accumulate gradient
	$\texttt{count} = \texttt{count} - 1$
	else
Compute Armijo step s_A from (22)	Compute step size s_t according to (23)
Perform update (10) or (21) using s_A	Perform update (9) or (20) using s_t
	$\texttt{count} = p$
	end if
$t = t + 1$	$t = t + 1$
until stopping criterion (24) is satisfied	**end while**
return $\mathbf{G}_t$	**return** $\mathbf{G}_T$

Figure 2: Pseudo-codes for the proposed batch and online algorithms.

mapping) do not appear to have a closed form in the considered geometry. Combining the gradient of (16) with the retractions (18) and (19) gives

$$\begin{aligned}\mathbf{U}_{t+1} &= \mathrm{qf}\left(\mathbf{U}_t - 2\lambda s_t(\hat{y}_t - y_t)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t^2\right),\\ \mathbf{R}_{t+1}^2 &= \mathbf{R}_t \exp\left(-(1-\lambda)s_t(\hat{y}_t - y_t)\mathbf{R}_t\mathbf{U}_t^T\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t\right)\mathbf{R}_t.\end{aligned}$$

A factorization $\mathbf{R}_{t+1}\mathbf{R}_{t+1}^T$ of $\mathbf{R}_{t+1}^2$ is obtained thanks to the property of matrix exponential, $\exp(\mathbf{A})^{\frac{1}{2}} = \exp(\frac{1}{2}\mathbf{A})$. Updating $\mathbf{R}_{t+1}$ instead of $\mathbf{R}_{t+1}^2$ is thus more efficient from a computational point of view, since it avoids the computation of a square root a each iteration. This yields the online gradient descent algorithm

$$\begin{aligned}\mathbf{U}_{t+1} &= \mathrm{qf}\left(\mathbf{U}_t - 2\lambda s_t(\hat{y}_t - y_t)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t^2\right),\\ \mathbf{R}_{t+1} &= \mathbf{R}_t \exp\left(-\frac{1}{2}(1-\lambda)s_t(\hat{y}_t - y_t)\mathbf{R}_t\mathbf{U}_t^T\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t\right),\end{aligned} \tag{20}$$

and the batch gradient descent algorithm

$$\begin{aligned}\mathbf{U}_{t+1} &= \mathrm{qf}\left(\mathbf{U}_t - 2\lambda s_t\frac{1}{n}\sum_{i=1}^n(\hat{y}_i - y_i)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathrm{Sym}(\mathbf{X}_i)\mathbf{U}_t\mathbf{R}_t^2\right),\\ \mathbf{R}_{t+1} &= \mathbf{R}_t \exp\left(-\frac{1}{2}(1-\lambda)s_t\frac{1}{n}\sum_{i=1}^n(\hat{y}_i - y_i)\mathbf{R}_t\mathbf{U}_t^T\mathrm{Sym}(\mathbf{X}_i)\mathbf{U}_t\mathbf{R}_t\right).\end{aligned} \tag{21}$$

7. Algorithms

This section documents implementation details of the proposed algorithms. Generic pseudo-codes are provided in Figure 2 and Table 1 summarizes computational complexities.

Data type	Input space	Batch flat (10)	Batch polar (21)	Online flat (9)	Online polar (20)
$\mathbf{X}$	$\mathbb{R}^{d\times d}$	$O(d^2rn)$	$O(d^2r^2n)$	$O(d^2rp)$	$O(d^2r^2p)$
$\mathbf{x}\mathbf{x}^T$	$\mathbb{R}^d$	$O(drn)$	$O(dr^2n)$	$O(drp)$	$O(dr^2p)$

Table 1: Computational costs of the proposed algorithms.

7.1 From Subspace Learning to Distance Learning

The update expressions (21) and (20) show that λ, the tuning parameter of the Riemannian metric (17), acts as a weighting factor on the search direction. A proper tuning of this parameter allows us to place more emphasis either on the learning of the subspace $\mathbf{U}$ or on the distance in that subspace $\mathbf{R}^2$. In the case $\lambda = 1$, the algorithm only performs subspace learning. Conversely, in the case $\lambda = 0$, the algorithm learns a distance for a fixed range space (see Section 8.1). Intermediate values of λ continuously interpolate between the subspace learning problem and the distance learning problem at fixed range space.

A proper tuning of λ is of interest when a good estimate of the subspace is available (for instance a subspace given by a proper dimension reduction technique) or when too few observations are available to jointly estimate the subspace and the distance within that subspace. In the latter case, one has the choice to favor either subspace or distance learning.

Experimental results of Section 10 recommend the value $\lambda = 0.5$ as the default setting.

7.2 Invariance Properties

A nice property of the proposed algorithms is their invariance with respect to rotations $\mathbf{W} \mapsto \mathbf{O}^T\mathbf{W}\mathbf{O}$, $\forall \mathbf{O} \in O(d)$. This invariance comes from the fact that the chosen metrics are invariant to rotations. A practical consequence is that a rotation of the input matrix $\mathbf{X} \mapsto \mathbf{O}\mathbf{X}\mathbf{O}^T$ (for instance a whitening transformation of the vectors $\mathbf{x} \mapsto \mathbf{O}\mathbf{x}$ if $\mathbf{X} = \mathbf{x}\mathbf{x}^T$) will not affect the behavior of the algorithms.

Besides being invariant to rotations, algorithms (20) and (21) are invariant with respect to scalings $\mathbf{W} \mapsto \mu\mathbf{W}$ with $\mu > 0$. Consequently, a scaling of the input data $(\mathbf{X}, y) \mapsto (\mu\mathbf{X}, \mu y)$, such as a change of units, will not affect the behavior of these algorithms.

7.3 Mini-Batch Extension of Online Algorithms

We consider a mini-batch extension of stochastic gradient algorithms. It consists in performing each gradient step with respect to $p \geq 1$ examples at a time instead of a single one. This is a classical speedup and stabilization heuristic for stochastic gradient algorithms. In the particular case $p = 1$, one recovers plain stochastic gradient descent. Given p samples $(\mathbf{X}_{t,1}, y_{t,1}), ..., (\mathbf{X}_{t,p}, y_{t,p})$, received at time t, the abstract update (2) becomes

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}\left(-s_t \, \frac{1}{p}\sum_{i=1}^{p} \mathrm{grad}\ell(\hat{y}_{t,i}, y_{t,i})\right).$$

7.4 Strategies for Choosing the Step Size

We here present strategies for choosing the step size in both the batch and online cases.

7.4.1 BATCH ALGORITHMS

For batch algorithms, classical backtracking methods exist (see Nocedal and Wright, 2006). In this paper, we use the Armijo step s_A defined at each iteration by the condition

$$f(R_{\mathbf{W}_t}(-s_A \operatorname{grad} f(\mathbf{W}_t))) \leq f(\mathbf{W}_t) + c\|\operatorname{grad} f(\mathbf{W}_t)\|^2_{\mathbf{W}_t}, \tag{22}$$

where $\mathbf{W}_t \in S_+(r,d)$ is the current iterate, $c \in (0,1)$, f is the empirical cost (1) and $R_{\mathbf{W}}$ is the chosen retraction. In this paper, we choose the particular value $c = 0.5$ and repetitively divide by 2 a specified maximum step size s_{max} until condition (22) is satisfied for the considered iteration. In order to reduce the dependence on s_{max} in a particular problem, it is chosen inversely proportional to the norm of the gradient at each iteration,

$$s_{max} = \frac{s_0}{\|\operatorname{grad} f(\mathbf{W}_t)\|_{\mathbf{W}_t}}.$$

A typical value of $s_0 = 100$ showed satisfactory results for all the considered problems.

7.4.2 ONLINE ALGORITHMS

For online algorithms, the choice of the step size is more involved. In this paper, the step size schedule s_t is chosen as

$$s_t = \frac{s}{\hat{\mu}_{grad}} \times \frac{nt_0}{nt_0 + t}, \tag{23}$$

where $s > 0$, n is the number of considered learning samples, $\hat{\mu}_{grad}$ is an estimate of the average gradient norm $\|\operatorname{grad} f(\mathbf{W}_0)\|_{\mathbf{W}_0}$, and $t_0 > 0$ controls the annealing rate of s_t. During a pre-training phase of our online algorithms, we select a small subset of learning samples and try the values 2^k with $k = -3, ..., 3$ for both s and t_0. The values of s and t_0 that provide the best decay of the cost function are selected to process the complete set of learning samples.

7.5 Stopping Criterion

Batch algorithms are stopped when the value or the relative change of the empirical cost f is small enough, or when the relative change in the parameter variation is small enough,

$$f(\mathbf{W}_{t+1}) \leq \varepsilon_{tol}, \quad \text{or} \quad \frac{f(\mathbf{W}_{t+1}) - f(\mathbf{W}_t)}{f(\mathbf{W}_t)} \leq \varepsilon_{tol}, \quad \text{or} \quad \frac{\|\mathbf{G}_{t+1} - \mathbf{G}_t\|_F}{\|\mathbf{G}_t\|_F} \leq \varepsilon_{tol}. \tag{24}$$

We found $\varepsilon_{tol} = 10^{-5}$ to be a good trade-off between accuracy and convergence time.

Online algorithms are run for a fixed number of epochs (number of passes through the set of learning samples). Typically, a few epochs are sufficient to attain satisfactory results.

7.6 Convergence

Gradient descent algorithms on matrix manifolds share the well-characterized convergence properties of their analog in $\mathbb{R}^d$. Batch algorithms converge linearly to a local minimum of the empirical cost that depends on the initial condition. Online algorithms converge asymptotically to a local minimum of the expected loss. They intrinsically have a much slower convergence rate than batch algorithms, but they generally decrease faster the expected loss in the large-scale regime (Bottou

and Bousquet, 2007). The main idea is that, given a training set of samples, an inaccurate solution may indeed have the same or a lower expected cost than a well-optimized one.

When learning a matrix $\mathbf{W} \in S_+(d)$, the problem is convex and the proposed algorithms converge toward a global minimum of the cost function, regardless of the initial condition. When learning a low-rank matrix $\mathbf{W} \in S_+(r,d)$, with $r < d$, the proposed algorithms converge to a local minimum of the cost function. This is not the case for heuristic methods proposed in the literature, which first reduce the dimensionality of the data before fitting a full-rank model on the reduced data (Davis and Dhillon, 2008; Weinberger and Saul, 2009).

For batch algorithms, the local convergence results follow from the convergence theory of line-search algorithms on Riemannian manifolds (see, for example, Absil et al., 2008).

For online algorithms, one can prove that the algorithm based on the flat geometry enjoys almost sure asymptotic convergence to a local minimum of the expected cost. In that case, the parameter $\mathbf{G}$ belongs to an Euclidean space and the convergence results presented by Bottou (1998) apply (see Appendix A for the main ideas of the proof).

In contrast, when the polar parameterization is used, the convergence results presented by Bottou (1998) do not apply directly because of the quotient nature of the search space. Because the extension would require technical arguments beyond the scope of the present paper, we refrain from stating a formal convergence result for the online algorithm based on the polar geometry, even though the result is quite plausible.

Due to the nonconvex nature of the considered rank-constrained problems, the convergence results are only local and little can be presently said about the global convergence of the algorithms. A global analysis of the critical points of the cost functions studied in the present paper is nevertheless not hopeless and could be facilitated by the considered low-rank parameterizations. For instance, global convergence properties have been established for PCA algorithms from an explicit analysis of the critical points (Chen et al., 1998). Also, recent results suggest good global convergence properties for closely related rank minimization problems (Recht et al., 2010). Experimental results suggest the same conclusions for the algorithms considered in this paper, which means that further research on global convergence results is certainly deserved.

8. Discussion

This section presents connections with existing works and extensions of the regression model.

8.1 Closeness-Based Approaches

A standard derivation of learning algorithms is as follows (Kivinen and Warmuth, 1997). The (online) update at time t is viewed as an (approximate) solution of

$$\mathbf{W}_{t+1} = \operatorname*{arg\,min}_{\mathbf{W} \in \mathcal{W}} D(\mathbf{W}, \mathbf{W}_t) \; + \; s_t \; \ell(\hat{y}, y_t), \tag{25}$$

where D is a well-chosen measure of closeness between elements of $\mathcal{W}$ and s_t is a trade-off parameter that controls the balance between the conservative term $D(\mathbf{W}, \mathbf{W}_t)$ and the innovation (or data fitting) term $\ell(\hat{y}, y_t)$. One solves (25) by solving the algebraic equation

$$\operatorname{grad} D(\mathbf{W}, \mathbf{W}_t) = -s_t \operatorname{grad} \ell(\hat{y}_{t+1}, y_t), \tag{26}$$

which is a first-order (necessary) optimality condition. If the search space $\mathcal{W}$ is a Riemannian manifold and if the closeness measure $D(\mathbf{W},\mathbf{W}_t)$ is the Riemannian distance, the solution of (26) is

$$\mathbf{W}_{t+1} = \mathrm{Exp}_{\mathbf{W}_t}(-s_t \text{ grad } \ell(\hat{y}_{t+1}, y_t)).$$

Because $\hat{y}_{t+1}$ must be evaluated in $\mathbf{W}_{t+1}$, this update equation is implicit. However, $\hat{y}_{t+1}$ is generally replaced by $\hat{y}_t$ (which is equal to $\hat{y}_{t+1}$ up to first order terms in s_t), which gives the update (2) where the exponential mapping is chosen as a retraction.

Bregman divergences have been popular closeness measures for $D(\mathbf{W},\mathbf{W}_t)$ because they render the optimization of (25) convex. Bregman divergences on the cone of positive definite matrices include the von Neumann divergence

$$D_{vN}(\mathbf{W},\mathbf{W}_t) = \mathrm{Tr}(\mathbf{W}\log\mathbf{W} - \mathbf{W}\log\mathbf{W}_t - \mathbf{W} + \mathbf{W}_t),$$

and the LogDet divergence

$$D_{ld}(\mathbf{W},\mathbf{W}_t) = \mathrm{Tr}(\mathbf{W}\mathbf{W}_t^{-1}) - \log\det(\mathbf{W}\mathbf{W}_t^{-1}) - d.$$

We have shown in Section 5 that the resulting updates can be interpreted as line-search updates for the log-Euclidean metric and the affine-invariant metric of $S_+(d)$ and for specific choices of the retraction mapping.

Likewise, the algorithm (9) can be recast in the framework (25) by considering the closeness

$$D_{flat}(\mathbf{W},\mathbf{W}_t) = \|\mathbf{G} - \mathbf{G}_t\|_F^2,$$

where $\mathbf{W} = \mathbf{G}\mathbf{G}^T$ and $\mathbf{W}_t = \mathbf{G}_t\mathbf{G}_t^T$. Algorithm (20) can be recast in the framework (25) by considering the closeness

$$D_{pol}(\mathbf{W},\mathbf{W}_t) = \lambda \sum_{i=1}^{r} \theta_i^2 \;+\; (1-\lambda)\,\|\log\mathbf{R}_t^{-1}\mathbf{R}^2\mathbf{R}_t^{-1}\|_F^2.$$

where the θ_i's are the principal angles between the subspaces spanned by $\mathbf{W}$ and $\mathbf{W}_t$ (Golub and Van Loan, 1996), and the second term is the affine-invariant distance of $S_+(d)$ between matrices $\mathbf{R}^2$ and $\mathbf{R}_t^2$ involved in the polar representation of $\mathbf{W}$ and $\mathbf{W}_t$.

Obviously, these closeness measures are no longer convex due to the rank constraint. However they reduce to the popular divergences in the full-rank case, up to second order terms. In particular, when $\lambda = 1$, the subspace is fixed and one recovers the setup of learning low-rank matrices of a fixed range space (Kulis et al., 2009). Thus, the algorithms introduced in the present paper can be viewed as generalizations of the ones presented in the paper of Kulis et al. (2009), where the issue of adapting the range space is presented as an open research question. Each of the proposed algorithms provides an efficient workaround for this problem at the expense of the (potential) introduction of local minima.

8.2 Handling Inequalities

Inequalities $\hat{y} \leq y$ or $\hat{y} \geq y$ can be considered by treating them as equalities when they are not satisfied. This is equivalent to the minimization of the continuously differentiable cost function

$$f(\mathbf{W}) = \ell(\hat{y}, y) = \frac{1}{2}\max(0, \rho(\hat{y} - y))^2,$$

where $\rho = +1$ if $\hat{y} \leq y$ is required and $\rho = -1$ if $\hat{y} \geq y$ is required.

8.3 Kernelizing the Regression Model

In this paper, we have not considered the kernelized model

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\phi(\mathbf{x})\phi(\mathbf{x})^T),$$

whose predictions can be extended to new input data $\phi(\mathbf{x})$ in the feature space $\mathcal{F}$ induced by the nonlinear mapping $\phi : \mathbf{x} \in \mathcal{X} \mapsto \phi(\mathbf{x}) \in \mathcal{F}$. This is potentially a useful extension of the regression model that could be investigated in the light of recent theoretical results in this area (for example Chatpatanasiri et al., 2010; Jain et al., 2010).

8.4 Connection with Multidimensional Scaling Algorithms

Given a set of m dissimilarity measures $\mathcal{D} = \{\delta_{ij}\}^m$ between n data objects, multidimensional scaling algorithms search for a r-dimensional embedding of the data objects into an Euclidean space representation $\mathbf{G} \in \mathbb{R}^{n \times r}$ (Cox and Cox, 2001; Borg and Groenen, 2005). Each row $\mathbf{g}$ of $\mathbf{G}$ is the coordinates of a data object in a Euclidean space of dimension r.

Multidimensional scaling algorithms based on gradient descent are equivalent to algorithms (9) and (10) when $\mathbf{X} = (\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T$, where $\mathbf{e}_i$ is the i-th unit vector (see Section 9.1), and when the multidimensional scaling reduction criterion is the SSTRESS

$$SSTRESS(\mathbf{G}) = \sum_{(i,j)\in\mathcal{D}} (\|\mathbf{g}_i - \mathbf{g}_j\|_2^2 - \delta_{ij})^2.$$

Vectors $\mathbf{g}_i$ and $\mathbf{g}_j$ are the i-th and j-th rows of matrix $\mathbf{G}$. Gradient descent is a popular technique in the context of multidimensional scaling algorithms. A stochastic gradient descent approach for minimizing the SSTRESS has also been proposed by Matsuda and Yamaguchi (2001). A potential area of future work is the application of the proposed online algorithm (9) for adapting a batch solution to slight modifications of the dissimilarities over time. This approach has a much smaller computational cost than recomputing the offline solution at every time step. It further allows to keep the coordinate representation coherent over time since the solution do not brutally jumps from a local minimum to another.

9. Applications

The choice of an appropriate distance measure is a central issue for many distance-based classification and clustering algorithms such as nearest neighbor classifiers, support vector machines or k-means. Because this choice is highly problem-dependent, numerous methods have been proposed to learn a distance function directly from data. In this section, we present two important distance learning applications that are compatible with the considered regression model and review some relevant literature on the subject.

9.1 Kernel Learning

In kernel-based methods (Shawe-Taylor and Cristianini, 2004), the data samples $\mathbf{x}_1, ..., \mathbf{x}_n$ are first transformed by a nonlinear mapping $\phi : \mathbf{x} \in \mathcal{X} \mapsto \phi(\mathbf{x}) \in \mathcal{F}$, where $\mathcal{F}$ is a new feature space that is expected to facilitate pattern detection into the data.

The kernel function is then defined as the dot product between any two samples in $\mathcal{F}$,

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j).$$

In practice, the kernel function is represented by a positive semidefinite matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ whose entries are defined as $\mathbf{K}_{ij} = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$. This inner product information is used solely to compute the relevant quantities needed by the algorithms based on the kernel. For instance, a distance is implicitly defined by any kernel function as the Euclidean distance between the samples in the new feature space

$$d_\phi(\mathbf{x}_i, \mathbf{x}_j) = \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 = \kappa(\mathbf{x}_i, \mathbf{x}_i) + \kappa(\mathbf{x}_j, \mathbf{x}_j) - 2\kappa(\mathbf{x}_i, \mathbf{x}_j),$$

which can be evaluated using only the elements of the kernel matrix by the formula

$$d_\phi(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij} = \mathrm{Tr}\left(\mathbf{K}(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\right),$$

which fits into the considered regression model.

Learning a kernel consists in computing the kernel (or Gram) matrix from scratch or improving a existing kernel matrix based on side-information (in a semi-supervised setting for instance). Data samples and class labels are generally exploited by means of equality or inequality constraints involving pairwise distances or inner products.

Most of the numerous kernel learning algorithms that have been proposed work in the so-called transductive setting, that is, it is not possible to generalize the learned kernel function to new data samples (Kwok and Tsang, 2003; Lanckriet et al., 2004; Tsuda et al., 2005; Zhuang et al., 2009; Kulis et al., 2009). In that setting, the total number of considered samples is known in advance and determines the size of the learned matrix. Recently, algorithms have been proposed to learn a kernel function that can be extended to new points (Chatpatanasiri et al., 2010; Jain et al., 2010). In this paper, we only consider the kernel learning problem in the transductive setting.

When low-rank matrices are considered, kernel learning algorithms can be regarded as dimensionality reduction methods. Very popular unsupervised algorithms in that context are kernel principal component analysis (Schölkopf et al., 1998) and multidimensional scaling (Cox and Cox, 2001; Borg and Groenen, 2005). Other kernel learning techniques include the maximum variance unfolding algorithm (Weinberger et al., 2004) and its semi-supervised version (Song et al., 2007), and the kernel spectral regression framework (Cai et al., 2007) which encompasses many reduction criterion (for example, linear discriminant analysis (LDA), locality preserving projection (LPP), neighborhood preserving embedding (NPE)). See the survey of Yang (2006) for a more complete state-of-the-art in this area.

Since our algorithms are able to compute a low-rank kernel matrix from data, they can be used for unsupervised or semi-supervised dimensionality reduction, depending whether or not the class labels are exploited through the imposed constraints.

9.2 Mahalanobis Distance Learning

Mahalanobis distances generalize the usual Euclidean distance as it allows to transform the data with an arbitrary rotation and scaling before computing the distance. Let $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ be two data samples, the (squared) Mahalanobis distance between these two samples is parameterized by a positive definite matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ and writes as

$$d_\mathbf{A}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{A} \, (\mathbf{x}_i - \mathbf{x}_j). \tag{27}$$

In the particular case of $\mathbf{A}$ being equal to the identity matrix, the standard Euclidean distance is obtained. A frequently used matrix is $\mathbf{A} = \mathbf{\Sigma}^{-1}$, the inverse of the sample covariance matrix. For

centered data features, computing this Mahalanobis distance is equivalent to perform a whitening of the data before computing the Euclidean distance.

For low-rank Mahalanobis matrices, computing the distance is equivalent to first perform a linear data reduction step before computing the Euclidean distance on the reduced data.[1] Learning a low-rank Mahalanobis matrix can thus be seen as learning a linear projector that is used for dimension reduction.

In contrast to kernel functions, Mahalanobis distances easily generalize to new data samples since the sole knowledge of $\mathbf{A}$ determines the distance function.

In recent years, Mahalanobis distance learning algorithms have been the subject of many contributions that cannot be all enumerated here. We review a few of them, most relevant for the present paper. The first proposed methods have been based on successive projections onto a set of large margin constraints (Xing et al., 2002; Shalev-Shwartz et al., 2004). The method proposed by Globerson and Roweis (2005) seeks a Mahalanobis matrix that maximizes the between classes distance while forcing to zero the within classes distance. A simpler objective is pursued by the algorithms that optimize the Mahalanobis distance for the specific k-nearest neighbor classifier (Goldberger et al., 2004; Torresani and Lee, 2006; Weinberger and Saul, 2009). Bregman projection based methods minimize a particular Bregman divergence under distance constraints. Both batch (Davis et al., 2007) and online (Jain et al., 2008) formulations have been proposed for learning full-rank matrices. Low-rank matrices have also been considered with Bregman divergences but only when the range space of the matrix is fixed in the first place (Davis and Dhillon, 2008; Kulis et al., 2009).

10. Experiments

Data Set	Samples	Features	Classes	Reference
GyrB	52	-	3	Tsuda et al. (2005)
Digits	300	16	3	Asuncion and Newman (2007)
Wine	178	13	13	Asuncion and Newman (2007)
Ionosphere	351	33	2	Asuncion and Newman (2007)
Balance Scale	625	4	3	Asuncion and Newman (2007)
Iris	150	4	3	Asuncion and Newman (2007)
Soybean	532	35	17	Asuncion and Newman (2007)
USPS	2,007	256	10	LeCun et al. (1989)
Isolet	7,797	617	26	Asuncion and Newman (2007)
Prostate	322	15,154	2	Petricoin et al. (2002)

Table 2: Considered data sets

In this section, we illustrate the potential of the proposed algorithms on several benchmark experiments. First, the proposed algorithms are evaluated on toy data. Then, they are compared to state-of-the-art kernel learning and Mahalanobis distance learning algorithms on real data sets. Overall, the experiments support that a joint estimation of a subspace and low-dimensional distance in that subspace is a major advantage of the proposed algorithms over methods that estimate the matrix for a subspace that is fixed beforehand.

1. In the low-rank case, one should rigorously refer to (27) as a pseudo-distance. Indeed, one has $d_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_j) = 0$ with $\mathbf{x}_i \neq \mathbf{x}_j$ whenever $(\mathbf{x}_i - \mathbf{x}_j)$ lies in the null space of $\mathbf{A}$.

Table 2 summarizes the different data sets that have been considered. As a normalization step, the data features are centered and rescaled to unit standard deviation.

The implementation of the proposed algorithms,[2] as well as the experiments of this paper are performed with Matlab. The implementations of algorithms MVU,[3] KSR,[4] LMNN,[5] and ITML,[6] have been rendered publicly available by Weinberger et al. (2004), Cai et al. (2007), Weinberger and Saul (2009) and Davis et al. (2007) respectively. Algorithms POLA (Shalev-Shwartz et al., 2004), LogDet-KL (Kulis et al., 2009) and LEGO (Jain et al., 2008) have been implemented on our own.

10.1 Toy Data

In this section, the proposed algorithms are evaluated on synthetic regression problems. The data vectors $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$ and the target matrix $\mathbf{W}^* \in S_+(r,d)$ are generated with entries drawn from a standard Gaussian distribution $\mathcal{N}(0,1)$. Observations follow

$$y_i = (\mathbf{x}_i^T \mathbf{W}^* \mathbf{x}_i)(1+\nu_i), \quad i = 1, ..., n, \tag{28}$$

where ν_i is drawn from $\mathcal{N}(0, 0.01)$. A multiplicative noise model is preferred over an additive one to easily control that observations remain nonnegative after the superposition of noise.

10.1.1 LEARNING THE SUBSPACE VS. FIXING THE SUBSPACE UP FRONT

As an illustrative example, we show the difference between two approaches for fitting the data to observations when a target model $\mathbf{W}^* \in S_+(3,3)$ is approximated with a parameter $\mathbf{W} \in S_+(2,3)$.

A naive approach to tackle that problem is to first project the data $\mathbf{x}_i \in \mathbb{R}^3$ on a subspace of reduced dimension and then to compute a full-rank model based on the projected data. Recent methods compute that subspace of reduced dimension using principal component analysis (Davis and Dhillon, 2008; Weinberger and Saul, 2009), that is, a subspace that captures a maximal amount of variance in the data. However, in general, there is no reason why the subspace spanned by the top principal components should coincide with the subspace that is defined by the target model. Therefore, a more appropriate approach consists in learning jointly the subspace and a distance in that subspace that best fits the data to observations within that subspace.

To compare the two approaches, we generate a set of learning samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{200}$, with $\mathbf{x}_i \in \mathbb{R}^3$ and y_i that follows (28). The target model is

$$\mathbf{W}^* = \tilde{\mathbf{U}} \mathbf{\Lambda} \tilde{\mathbf{U}}^T$$

where $\tilde{\mathbf{U}}$ is a random 3×3 orthogonal matrix and $\mathbf{\Lambda}$ is a diagonal matrix with two dominant values $\Lambda_{11}, \Lambda_{22} \gg \Lambda_{33} > 0$ (for this specific example, $\Lambda_{11} = 4, \Lambda_{22} = 3$ and $\Lambda_{33} = 0.01$). Observations y_i are thus nearly generated by a rank-2 model, such that $\mathbf{W}^*$ should be well approximated with a matrix $\mathbf{W} \in S_+(2,3)$ that minimizes the train error.

Results are presented in Figure 3. The top plot shows that the learned subspace (which identifies with the target subspace) is indeed very different from the subspace spanned by the top two principal components. Moreover, the bottom plots clearly demonstrate that the fit is much better when the

2. The source code is available from `http://www.montefiore.ulg.ac.be/~meyer`.
3. MVU is available from `http://www.cse.wustl.edu/~kilian/Downloads/MVU.html`.
4. KSR is available from `http://www.cs.uiuc.edu/homes/dengcai2/SR/`.
5. LMNN is available from `http://www.cse.wustl.edu/~kilian/Downloads/LMNN.html`.
6. ITML is available from `http://www.cs.utexas.edu/users/pjain/itml/`.

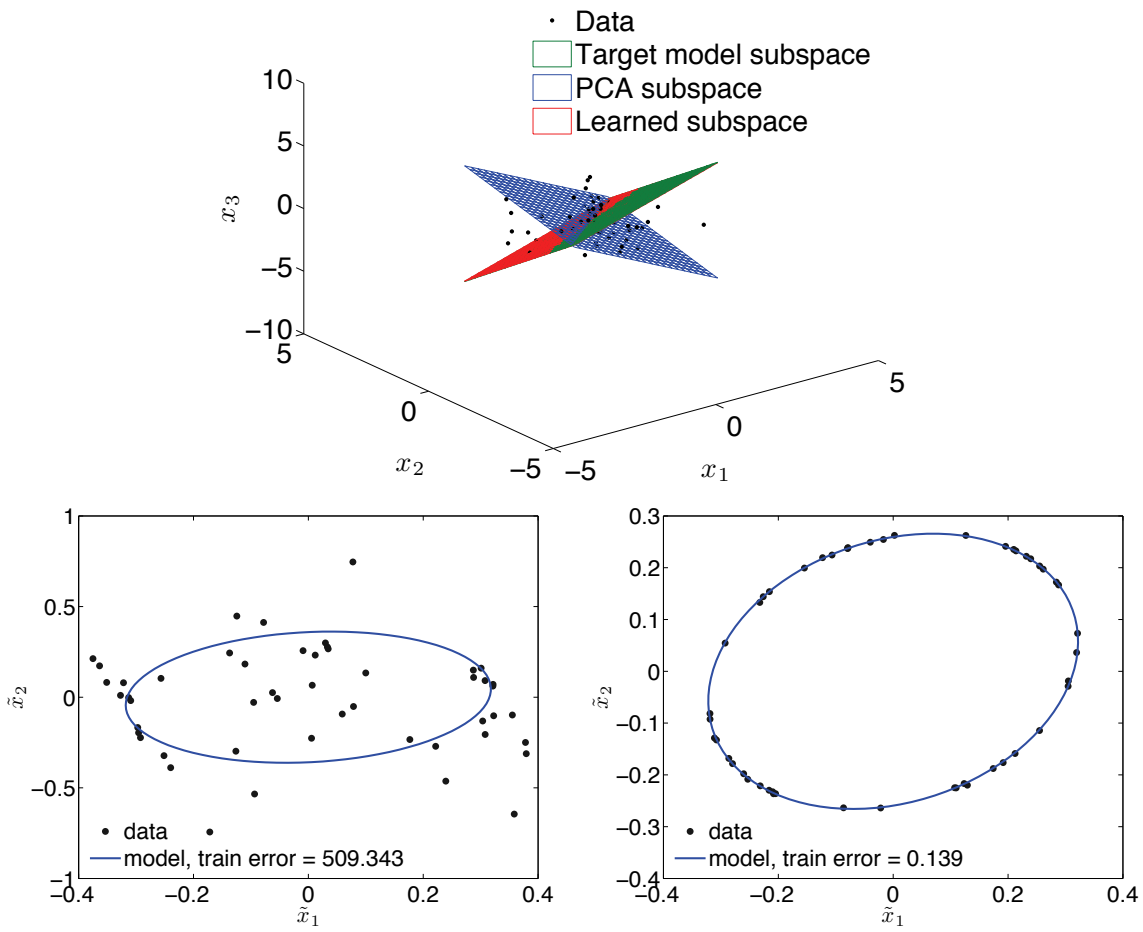


Figure 3: Learning vs fixing the subspace. **Top**: the learned subspace is very different from the subspace computed from a classical heuristic. **Bottom left**: fit after projection of the data onto a subspace fixed up front. **Bottom right**: fit obtained with a join estimation of the subspace and a distance within that subspace.

subspace and the distance in that subspace are learned jointly. The difference is also significant in terms of the train error. This simple example shows that heuristic methods that fix the range space in the first place may converge to a solution that is very different from a minimum of the desired cost function. For visualization purpose, the two dimensional model is represented by the ellipse

$$\mathcal{E} = \{\tilde{\mathbf{x}}_i \in \mathbb{R}^2 : \tilde{\mathbf{x}}_i^T \mathbf{R}^2 \tilde{\mathbf{x}}_i = 1\}, \qquad \text{where} \quad \tilde{\mathbf{x}}_i = \frac{\mathbf{U}^T \mathbf{x}_i}{\sqrt{y_i}},$$

and $(\mathbf{U}, \mathbf{R}^2)$ are computed with algorithm (21), either in the setting $\lambda = 0$ that fixes the subspace to the PCA subspace (left) or in the setting $\lambda = 0.5$ that simultaneously learned $\mathbf{U}$ and $\mathbf{B}$ (right). A perfect fit is obtained when all $\tilde{\mathbf{x}}_i$ are located on $\mathcal{E}$, which is the locus of points where $\hat{y}_i = y_i$.

10.1.2 INFLUENCE OF λ ON THE ALGORITHM BASED ON THE POLAR GEOMETRY

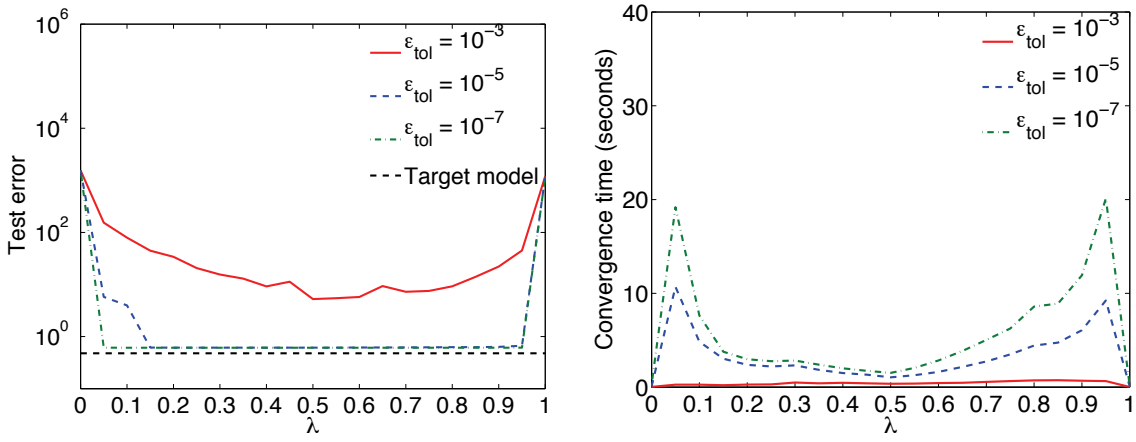


Figure 4: Influence of λ.

In theory, the parameter λ should not influence the algorithm since it has no effect on the first-order optimality conditions except for its two extreme values $\lambda = 0$ and $\lambda = 1$. In practice however, a sensitivity to this parameter is observed due to the finite tolerance of the stopping criterion: the looser the tolerance, the more sensitive to λ.

To investigate the sensitivity to λ, we try to recover a target parameter $\mathbf{W}^* \in S_+(5,10)$ using pairs $(\mathbf{x}_i, y_i)$ generated according to (28). We generate 10 random regression problems with 1000 samples partitioned into 500 learning samples and 500 test samples. We compute the mean test error and the mean convergence time as a function of λ for different values of ε_{tol}. The results are presented in Figure 4. As ε_{tol} decrease, the test error becomes insensitive to λ, but an influence is observed on the convergence time of the algorithm.

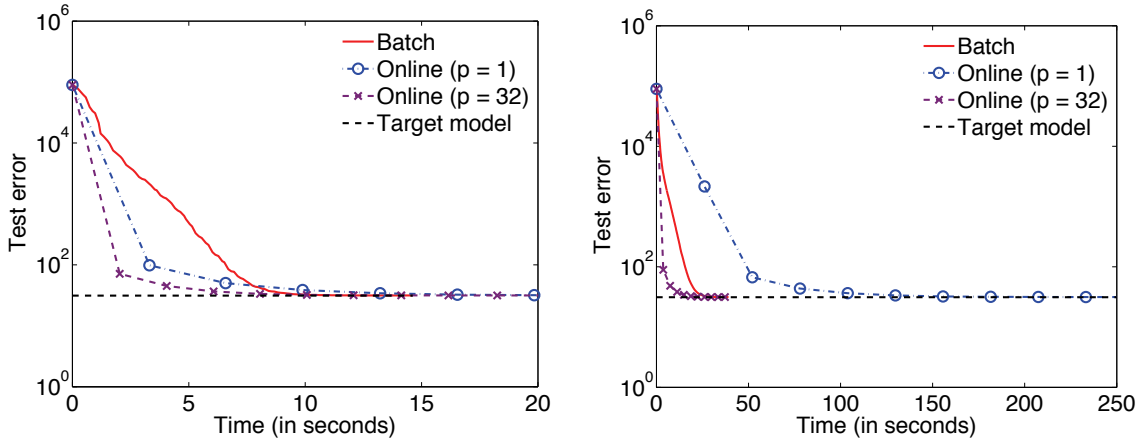
In view of these results, we recommend the value 0.5 as the default setting for λ. Unless specified otherwise, we therefore use this particular value for all experiments in this paper.

10.1.3 ONLINE VS. BATCH

This experiment shows that when a large amount of sample is available (80,000 training samples and 20,000 test samples for learning a parameter $\mathbf{W}^*$ in $S_+(10,50)$), online algorithms minimize the test error more rapidly than batch ones. It further shows that the mini-batch extension allows to improve significantly the performance compared to the plain stochastic gradient descent setting ($p = 1$). We observe that the mini-batch size $p = 32$ generally gives good results. Figure 5 report the test error as a function of the learning time, that is, the time after each iteration for batch algorithm and the time after each epoch for online algorithms. For the algorithm based on the polar geometry, the mini-batch extension is strongly recommended to amortize the larger cost of each update.

10.2 Kernel Learning

In this section, the proposed algorithms are applied to the problem of learning a kernel matrix from pairwise distance constraints between data samples. As mentioned earlier, we only consider



(a) Flat geometry (b) Polar geometry

Figure 5: Online vs Batch. For a large number of samples, online algorithms reduce the test error much more rapidly than batch ones. Using the mini-batch extension generally improve significantly the performance.

this problem in the transductive setting, that is, all samples $\mathbf{x}_1, \ldots \mathbf{x}_n$ are available up front and the learned kernel do not generalize to new samples.

10.2.1 EXPERIMENTAL SETUP

After transformation of the data with the kernel map $\mathbf{x} \mapsto \phi(\mathbf{x})$, the purpose is to compute a fixed-rank kernel matrix based on a limited amount of pairwise distances in the kernel feature space and on some information about class labels.

Distance constraints are generated as $\hat{y}_{ij} \leq y_{ij}(1-\alpha)$ for identically labeled samples and $\hat{y}_{ij} \geq y_{ij}(1+\alpha)$ for differentially labeled samples, where $\alpha \geq 0$ is a scaling factor, $y_{ij} = \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2$ and $\hat{y}_{ij} = \mathrm{Tr}(\mathbf{W}(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T) = (\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{W}(\mathbf{e}_i - \mathbf{e}_j)$.

We investigate both the influence of the amount of side-information provided, the influence of the approximation rank and the computational time required by the algorithms.

To quantify the performance of the learned kernel matrix, we perform either a classification or a clustering of the samples based on the learned kernel. For classification, we compute the test set accuracy of a k-nearest neighbor classifier ($k = 5$) using a two-fold cross-validation protocol (results are averaged over 10 random splits). For clustering, we use the K-means algorithm with the number of clusters equal to the number of classes in the problem. To overcome K-means local minima, 10 runs are performed in order to select the result that has lead to the smaller value of the K-means objective. The quality of the clustering is measured by the normalized mutual information (NMI) shared between the random variables of cluster indicators C and target labels T (Strehl et al., 2000),

$$NMI = \frac{2\, I(C;T)}{(H(C) + H(T))},$$

where $I(X_1;X_2) = H(X_1) - H(X_1|X_2)$ is the mutual information between the random variables X_1 and X_2, $H(X_1)$ is the Shannon entropy of X_1, and $H(X_1|X_2)$ is the conditional entropy of X_1 given X_2. This score ranges from 0 to 1, the larger the score, the better the clustering quality.

10.2.2 COMPARED METHODS

We compare the following methods:

1. Batch algorithms (10) and (21), adapted to handle inequalities (see Section 8.2),
2. The kernel learning algorithm LogDet-KL (Kulis et al., 2009) which learn kernel matrices of fixed range space for a given set of distance constraints.
3. The kernel spectral regression (KSR) algorithm of Cai et al. (2007) using a similarity matrix $\mathbf{N}$ constructed as follows. Let $\mathbf{N}$ be the adjacency matrix of a 5-NN graph based on the initial kernel. We modify $\mathbf{N}$ according to the set of available constraints: $\mathbf{N}_{ij} = 1$ if samples $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class (must-link constraint), $\mathbf{N}_{ij} = 0$ if samples $\mathbf{x}_i$ and $\mathbf{x}_j$ do not belong to the same class (cannot-link constraint).
4. The Maximum Variance Unfolding (MVU) algorithm (Weinberger et al., 2004),
5. The Kernel PCA algorithm (Schölkopf et al., 1998).

The last two algorithms are unsupervised techniques that are provided as baselines.

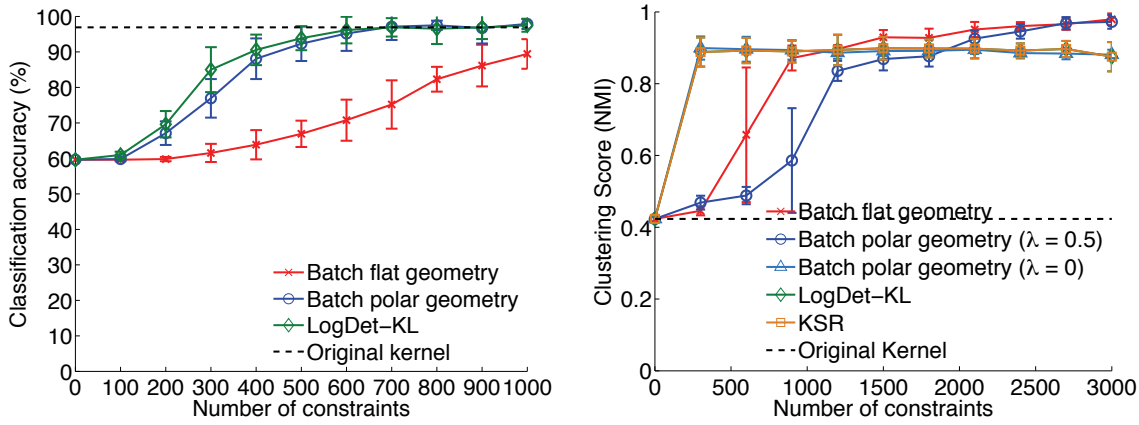


Figure 6: **Left**: full-rank kernel learning on the Gyrb data set. The algorithm based on the polar geometry competes with LogDet-KL. **Right**: low-rank kernel learning on the Digits data set. The proposed algorithms outperform the compared methods as soon as a sufficiently large number of constraints is provided.

10.2.3 RESULTS

The first experiment is reproduced from Tsuda et al. (2005) and Kulis et al. (2009). The goal is to reconstruct the GyrB kernel matrix based on distance constraints only. This matrix contains

information about the proteins of three bacteria species. The distance constraints are randomly generated from the original kernel matrix with $\alpha = 0$. We compare the proposed batch methods with the LogDet-KL algorithm, the only competing algorithm that also learns directly from distance constraints. This algorithm is the best performer reported by Kulis et al. (2009) for this experiment. All algorithms start from the identity matrix that do not encode any domain information. Figure 6 (left) reports the k-NN classification accuracy as a function of the number of distance constraints provided. In this full-rank learning setting, the algorithm based on the polar geometry compete with the LogDet-KL algorithm. The convergence time of the algorithm based on the polar geometry is however much faster (0.15 seconds versus 58 seconds for LogDet-KL when learning 1000 constraints). The algorithm based on the flat geometry has inferior performance when too few constraints are provided. This is because in the kernel learning setting, updates of this algorithm only involve the rows and columns that correspond to the set of points for which constraints are provided. It may thus result in a partial update of the kernel matrix entries. This issue disappears as the number of provided constraints increases.

The second experiment is reproduced from Kulis et al. (2009). It aims at improving an existing low-rank kernel using limited information about class labels. A rank-16 kernel matrix is computed for clustering a database of 300 handwritten digits randomly sampled from the 3, 8 and 9 digits of the Digits data set (since we could not find out the specific samples that have been selected by Kulis et al. (2009), we made our own samples selection). The distance constraints are randomly sampled from a linear kernel on the input data $\mathbf{K} = \mathbf{X}\mathbf{X}^T$ and $\alpha = 0.25$. The results are presented in Figure 6 (right). The figure shows that KSR, LogDet-KL and the algorithm based on the polar geometry with $\lambda = 0$ perform similarly. These methods are however outperformed by the proposed algorithms (flat geometry and polar geometry with $\lambda = 0.5$) when the number of constraints is large enough. This experiment also enlightens the flexibility of the polar geometry, which allows us to fix the subspace in situations where too few constraints are available.

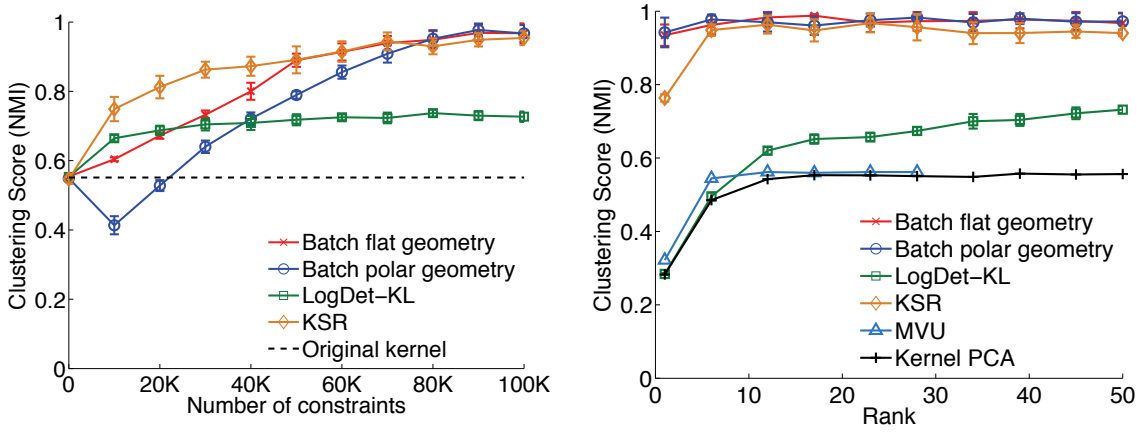


Figure 7: Clustering the USPS data set. **Left:** clustering score versus number of constraints. **Right:** clustering score versus approximation rank. When the number of provided constraints is large enough, the proposed algorithms perform as good as the KSR algorithm. It outperforms the LogDet-KL algorithm and baselines.

Finally, we tackle the kernel learning problem on a larger data set. We use the test set of the USPS data set,[7] which contains 2007 samples of handwritten zip code digits. The data are first transformed using the kernel map $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$ with $\gamma = 0.001$ and we further center the data in the kernel feature space. Pairwise distance constraints are randomly sampled from that kernel matrix with $\alpha = 0.5$. Except KSR that has its own initialization procedure, algorithms start from the kernel matrix provided by kernel PCA.

Figure 7 (left) shows the clustering performance as a function of the number of constraints provided when the approximation rank is fixed to $r = 25$. Figure 7 (right) reports the clustering performance as a function of the approximation rank when the number of constraints provided is fixed to $100K$. When the number of provided constraints is large enough, the proposed algorithms perform as good as KSR and outperform the LogDet-KL method that learn a kernel of fixed-range space. Average computational times for learning a rank-6 kernel from $100K$ constraints are 0.57 seconds for KSR, 3.25 seconds for the algorithm based on the flat geometry, 46.78 seconds for LogDet-KL and 47.30 seconds for the algorithm based on the polar geometry. In comparison, the SDP-based MVU algorithm takes 676.60 seconds to converge.

10.3 Mahalanobis Distance Learning

In this section, we tackle the problem of learning from data a Mahalanobis distance for supervised classification and compare our methods to state-of-the-art Mahalanobis metric learning algorithms.

10.3.1 EXPERIMENTAL SETUP

For the considered problem, the purpose is to learn the parameter $\mathbf{W}$ of a Mahalanobis distance $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{W} (\mathbf{x}_i - \mathbf{x}_j)$, such that the distance satisfies as much as possible a given set of constraints. As in the paper of Davis et al. (2007), we generate the constraints from the learning set of samples as $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) \leq l$ for same-class pairs and $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) \geq u$ for different-class pairs. The scalars u and l estimate the 95-th and 5-th percentiles of the distribution of Mahalanobis distances parameterized by a chosen baseline $\mathbf{W}_0$. The performance of the learned distance is then quantified by the test error rate of a k-nearest neighbor classifier based on the learned distance. All experiments use the setting $k = 5$, breaking ties arbitrarily. Unless for the Isolet data set for which a specific train/test partition is provided, error rates are computed using two-fold cross validation. Results are averaged over 10 random partitions.

10.3.2 COMPARED METHODS

We compare the following distance learning algorithms:

1. Batch algorithms (10) and (21),
2. ITML (Davis et al., 2007),
3. LMNN (Weinberger and Saul, 2009),
4. Online algorithms (9) and (20),
5. LEGO (Jain et al., 2008),

7. We use the ZIP code data from `http://www-stat-class.stanford.edu/~tibs/ElemStatLearn/data.html`.

6. POLA (Shalev-Shwartz et al., 2004).

When some methods require the tuning of an hyper-parameter, this is performed by a two-fold cross-validation procedure. The slack parameter of ITML as well as the step size of POLA are selected in the range of values 10^k with $k = -3, ..., 3$. The step size of LEGO is selected in this same range of value for the UCI data sets, and in the range of value 10^k with $k = -10, ..., -5$ for the larger data sets Isolet and Prostate.

10.3.3 RESULTS

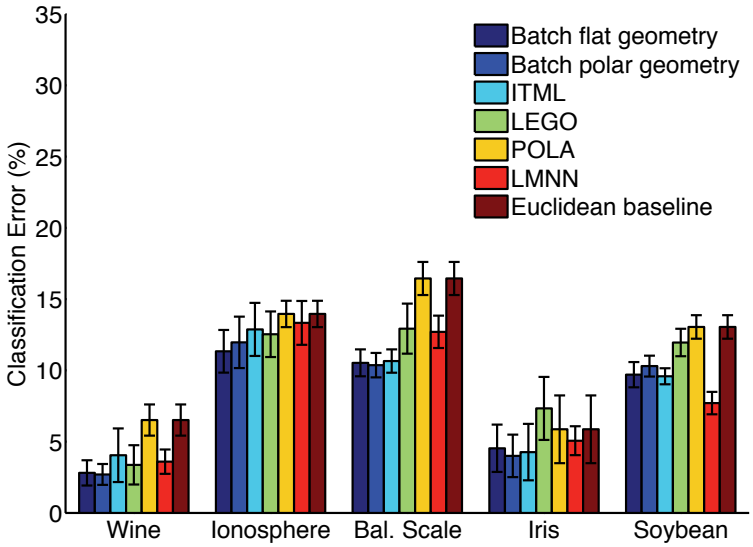
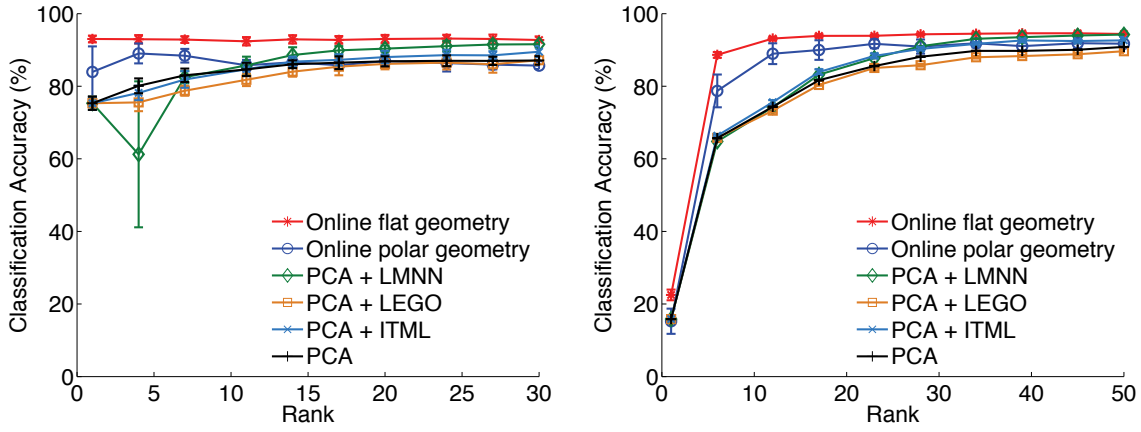


Figure 8: Full-rank distance learning on the UCI data sets. The proposed algorithms compete with state-of-the-art methods for learning a full-rank Mahalanobis distance.

Reproducing a classical benchmark experiment from Kulis et al. (2009), we demonstrate that the proposed batch algorithms compete with state-of-the-art full-rank Mahalanobis distance learning algorithms on several UCI data sets (Figure 8). We have not included the online versions of our algorithms in this comparison because we consider that the batch approaches are more relevant on such small data sets. Except POLA and LMNN which do not learn from provided pairwise constraints, all algorithms process $40c(c-1)$ constraints, where c is the number of classes in the data. We choose the Euclidean distance ($\mathbf{W}_0 = \mathbf{I}$) as the baseline distance for initializing the algorithms. Figure 8 reports the results. The two proposed algorithms compete favorably with the other full-rank distance learning techniques, achieving the minimal average error for 4 of the 5 considered data sets.

We finally evaluate the proposed algorithms on higher-dimensional data sets in the low-rank regime (Figure 9). The distance constraints are generated as in the full-rank case, but the initial baseline matrix is now computed as $\mathbf{W}_0 = \mathbf{G}_0\mathbf{G}_0^T$, where $\mathbf{G}_0$'s columns are the top principal directions of the data. For the Isolet data set, $100K$ constraints are generated, and $10K$ constraints are generated for the Prostate data set. For scalability reasons, algorithms LEGO, LMNN and ITML must proceed in two steps: the data are first projected onto the top principal directions and then



(a) Prostate (b) Isolet

Figure 9: Low-rank Mahalanobis distance learning. For low values of the rank, the proposed algorithms perform much better than the methods that project the data on the top principal directions and learn a full-rank distance on the projected data.

a full-rank distance is learned within the subspace spanned by these top principal directions. In contrast, our algorithms are initialized with the top principal direction, but they operate on the data in their original feature space. Overall, the proposed algorithms achieve much better performance than the methods that first reduce the data. This is particularly striking when the rank is very small compared to problem size. The performance gap reduces as the rank increases. However, for high-dimensional problems, one is usually interested in efficient low-rank approximations that gives satisfactory results.

11. Conclusion

In this paper, we propose gradient descent algorithms to learn a regression model parameterized by a fixed-rank positive semidefinite matrix. The rich Riemannian geometry of the set of fixed-rank PSD matrices is exploited through a geometric optimization approach.

The resulting algorithms overcome the main difficulties encountered by the previously proposed methods as they scale to high-dimensional problems, and they naturally enforce the rank constraint as well as the positive definite property while leaving the range space of the matrix free to evolve during optimization.

We apply the proposed algorithms to the problem of learning a distance function from data, when the distance is parameterized by a fixed-rank positive semidefinite matrix. The good performance of the proposed algorithms is illustrated over several benchmarks.

Acknowledgments

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Bel-

gian State, Science Policy Office. The scientific responsibility rests with its authors. Gilles Meyer is supported as an FRS-FNRS research fellow (Belgian Fund for Scientific Research).

Appendix A. Convergence Proof of Algorithm (9)

Bottou (1998) reviews the mathematical tools required to prove almost sure convergence, that is asymptotic convergence with probability one, of stochastic gradient algorithms. Almost sure convergence follows from the following five assumptions:

(A1) $F(\mathbf{G}) = \mathbb{E}_{\mathbf{X},y}\{\ell(\hat{y},y)\} \geq 0$ is three times differentiable with bounded derivatives,

(A2) the step sizes satisfy $\sum_{t=1}^{\infty} \eta_t^2 < \infty$ and $\sum_{t=1}^{\infty} \eta_t = \infty$,

(A3) $\mathbb{E}_{\mathbf{X},y}\{\|\text{grad} f(\mathbf{G})\|_F^2\} \leq k_1 + k_2\|\mathbf{G}\|_F^2$, where $f(\mathbf{G}) = \ell(\hat{y},y)$,

(A4) $\exists h_1 > 0, \inf_{\|\mathbf{G}\|_F^2 > h_1} \text{Tr}(\mathbf{G}^T \mathbb{E}_{\mathbf{X},y}\{\text{grad} f(\mathbf{G})\}) > 0$,

(A5) $\exists h_2 > h_1, \forall (\mathbf{X},y) \in \mathcal{X} \times \mathcal{Y}, \sup_{\|\mathbf{G}\|_F^2 < h_2} \|\text{grad} f(\mathbf{G})\|_F \leq k_3$,

where $\|\cdot\|_F$ is the Frobenius norm. Provided that algorithm (9) is equipped with an adaptive step size $s_t = \eta_t / \max(\|\mathbf{G}_t\|_F^2, 1)$, where η_t satisfy (A2), we have the following convergence result.

Proposition 1 *For bounded data* $(\mathbf{X},y)$*, algorithm* (9) *equipped with the step size* s_t *defined above converges almost surely to the set of stationary points of the cost function* $\mathbb{E}_{\mathbf{X},y}\{(\hat{y}-y)^2/2\}$.

Proof The proof is completed in two steps. First, it is shown that the stochastic sequence

$$u_t = \max(h_2, \|\mathbf{G}_t\|_F^2),$$

defines a Lyapunov process (always positive and decreasing on average) which is bounded almost surely by h_2. This implies that $\mathbf{G}_t$ is almost surely confined within distance $\sqrt{h_2}$ from the origin and provides almost sure bounds on all continuous functions of $\mathbf{G}_t$. In Bottou (1998), confinement is essentially based on (A3) and (A4). In the current proof, we rely on the fact that $\mathbb{E}_{\mathbf{X},y}\{\|\text{grad} f(\mathbf{G}) / \max(\|\mathbf{G}\|_F^2, 1)\|_F^2\} \leq k_1 + k_2\|\mathbf{G}\|_F^2$.

Second, the Lyapunov process $v_t = F(\mathbf{G}_t) \geq 0$ is proved to converge almost surely. Convergence of $F(\mathbf{G}_t)$ is then used to show that $w_t = \text{grad}\, F(\mathbf{G}_t)$ tends to zero almost surely. Technical details are adapted from the paper of Bottou (1998). ■

In practice, saddle points and local maxima are unstable solutions while convergence to asymptotic plateaus is excluded by (A4). As a result, almost sure convergence to a local minimum of the expected cost is obtained.

References

P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Appl. Math.*, 80(2):199–220, 2004.

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

R. Arora. On learning rotations. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 55–63. MIT Press, 2009.

V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM Journal on Matrix Analysis and Applications*, 29 (1):328–347, 2007.

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`. University of California, Irvine, School of Information and Computer Sciences.

F. Bach and M. I. Jordan. Predictive low-rank decomposition for kernel methods. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, 2005.

S. Bonnabel and R. Sepulchre. Riemannian metric and geometric mean for positive semidefinite matrices of fixed rank. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1055–1070, 2009.

I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, second edition, 2005.

L. Bottou. Online algorithms and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. Cambridge University Press, 1998.

L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, number 3176 in Lecture Notes in Artificial Intelligence, LNAI-3176, pages 146–168. Springer Verlag, 2004.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 161–168. MIT Press, 2007.

D. Cai, X. He, and J. Han. Efficient kernel discriminant analysis via spectral regression. In *Proceedings of the International Conference on Data Mining (ICDM07)*, 2007.

R. Chatpatanasiri, T. Korsrilabutr, P. Tangchanachaianan, and B. Kijsirikul. A new kernelization framework for Mahalanobis distance learning algorithms. *Neurocomputing*, 73(10-12):1570–1579, 2010.

T. Chen, Y. Hua, and W.-Y. Yan. Global convergence of Oja's subspace algorithm for principal component extraction. *IEEE Transaction Neural Networks*, 9(1):58–67, 1998.

T. F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

K. Crammer. Online tracking of linear subspaces. In *Proceedings of 19th Annual Conference on Learning Theory (COLT)*, 2006.

J. V. Davis and I. S. Dhillon. Structured metric learning for high dimensional problems. In *Proceedings of the 14th ACM SIGKDD conference on Knowledge Discovery and Data Mining*, 2008.

J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, 2007.

A. Edelman, T.A. Arias, and S.T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

J. Faraut and A. Koranyi. *Analysis on Symmetric Cones*. Oxford University Press, 1994.

S. Fine, K. Scheinberg, N. Cristianini, J. Shawe-Taylor, and B. Williamson. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

A. Globerson and S. Roweis. Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 451–458. MIT Press, 2005.

J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 513–520. MIT Press, 2004.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21, pages 761–768. MIT Press, 2008.

P. Jain, B. Kulis, and I. S. Dhillon. Inductive regularized learning of kernel functions. 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 946–954. MIT Press, 2010.

M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization for semidefinite convex problems. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

J. Kivinen and M.K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Journal of Information and Computation*, 132(1):1–64, January 1997.

B. Kulis, M. Sustik, and I. S. Dhillon. Low-rank kernel learning with Bregman matrix divergences. *Journal of Machine Learning Research*, 10:341–376, 2009.

J. Kwok and I. Tsang. Learning with idealized kernels. *Proceedings of the 20th International Conference on Machine learning (ICML)*, 2003.

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. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In David Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan Kaufman, 1989.

Y. Matsuda and K. Yamaguchi. Global mapping analysis: stochastic gradient algorithm in SSTRESS and classical MDS stress. In *Proceedings of International Conference on Neural Information Processing*, 2001.

J. Nocedal and S. J. Wright. *Numerical Optimization, Second Edition*. Springer, 2006.

E. Oja. Principal components, minor components, and linear neural networks. *Neural Networks*, 5: 927 – 935, 1992.

E.F. Petricoin, D.K. Ornstein, C.P. Paweletz, A.M. Ardekani, P.S. Hackett, B.A. Hitt, A. Velassco, C. Trucco, L. Wiegand, K. Wood, C.B. Simone, P.J. Levine, W.M. Linehan, M.R. Emmert-Buck, S.M. Steinberg, E.C Kohn, and L.A. Liotta. Serum proteomic patterns for detection of prostate cancer. *Journal of the National Cancer Institute*, 94(20):1576–1578, 2002.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum rank solutions to linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.

B. Schölkopf, A. J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(3):1299–1319, 1998.

S. Shalev-Shwartz, Y. Singer, and A. Ng. Online and batch learning of pseudo-metrics. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 2004.

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

S.T. Smith. Covariance, subspace, and intrinsic Cramér-Rao bounds. *IEEE Transactions on Signal Processing*, 53(5):1610–1630, 2005.

L. Song, A. Smola, K. Borgwardt, and A. Gretton. Colored maximum variance unfolding. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 1385–1392. MIT Press, 2007.

A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search (AAAI)*, 2000.

L. Torresani and K. Lee. Large margin component analysis. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19, pages 1385–1392. MIT Press, 2006.

K. Tsuda, G. Ratsch, and M. Warmuth. Matrix exponentiated gradient updates for on-line learning and Bregman projection. *Journal of Machine Learning Research*, 6:995–1018, 2005.

M. Warmuth. Winnowing subspaces. *Proceedings of the 24th international conference on Machine learning (ICML)*, 2007.

K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

K. Weinberger, F. Sha, and L. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. *Proceedings of the 21st International Conference on Machine Learning (ICML)*, pages 839–846, 2004.

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 505–512. MIT Press, 2002.

L. Yang. Distance metric learning: a comprehensive survey. Technical report, Michigan State University, 2006.

J. Zhuang, I. Tsang, and S. Hoi. SimpleNPKL: simple non-parametric kernel learning. *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009.

Journal of Machine Learning Research 12 (2011) 627-662 Submitted 12/08; Revised 7/10; Published 2/11

Parameter Screening and Optimisation for ILP using Designed Experiments

Ashwin Srinivasan ASHWIN.SRINIVASAN@WOLFSON.OXON.ORG
School of Mathematical Sciences & ICT
South Asian University
New Delhi 110067, India *

Ganesh Ramakrishnan GANESH@CSE.IITB.AC.IN
Dept. of Computer Science and Engineering
Indian Institute of Technology Bombay
Mumbai, India

Editor: Luc De Raedt

Abstract

Reports of experiments conducted with an Inductive Logic Programming system rarely describe how specific values of parameters of the system are arrived at when constructing models. Usually, no attempt is made to identify sensitive parameters, and those that are used are often given "factory-supplied" default values, or values obtained from some non-systematic exploratory analysis. The immediate consequence of this is, of course, that it is not clear if better models could have been obtained if some form of parameter selection and optimisation had been performed. Questions follow inevitably on the experiments themselves: specifically, are all algorithms being treated fairly, and is the exploratory phase sufficiently well-defined to allow the experiments to be replicated? In this paper, we investigate the use of parameter selection and optimisation techniques grouped under the study of experimental design. Screening and response surface methods determine, in turn, sensitive parameters and good values for these parameters. Screening is done here by constructing a stepwise regression model relating the utility of an ILP system's hypothesis to its input parameters, using systematic combinations of values of input parameters (technically speaking, we use a two-level fractional factorial design of the input parameters). The parameters used by the regression model are taken to be the sensitive parameters for the system for that application. We then seek an assignment of values to these sensitive parameters that maximise the utility of the ILP model. This is done using the technique of constructing a local "response surface". The parameters are then changed following the path of steepest ascent until a locally optimal value is reached. This combined use of parameter selection and response surface-driven optimisation has a long history of application in industrial engineering, and its role in ILP is demonstrated using well-known benchmarks. The results suggest that computational overheads from this preliminary phase are not substantial, and that much can be gained, both on improving system performance and on enabling controlled experimentation, by adopting well-established procedures such as the ones proposed here.

Keywords: inductive logic programming, parameter screening and optimisation, experimental design

∗. A.S. also holds an adjunct position at the School of CSE, University of New South Wales, Sydney; and visiting position at the Oxford University Computing Laboratory, Oxford.

©2011 Ashwin Srinivasan and Ganesh Ramakrishnan.

1. Introduction

We are concerned in this paper with Inductive Logic Programming (ILP) primarily as a tool for constructing models. Specifications of the appropriate use of a tool, its testing, and analysis of benefits and drawbacks over others of a similar nature are matters for the engineer concerned with its routine day-to-day use. Much of the literature on the applications of ILP have, to date, been once-off demonstrations of either the model construction abilities of a specific system, or of the ability of ILP systems to represent and use complex domain-specific relationships (Bratko and Muggleton, 1995; Dzeroski, 2001). It is not surprising, therefore, that there has been little reported on practical issues that arise with the actual use of an ILP system.

Assuming some reasonable solution has been found to difficult practical problems like the appropriateness of the representation, choice of relevant "background knowledge", poor user-interfaces, and efficiency,[1] we are concerned here with a substantially simpler issue. Like all model-building methods, an ILP system's performance is affected by values assigned to input parameters (the term is used here in the sense understood by the computer scientist, and not the statistician). For example, the model constructed by an ILP system may be affected by the maximal length of clauses, the minimum precision allowed for any clause in the theory, the maximum number of new variables that could appear in any clause, and so. The ILP practitioner is immediately confronted with two questions: (a) Which of these parameters are relevant for the particular application at hand?; and (b) What should their values be in order to get a good model? In an industrial setting, an engineer confronted with similar questions about a complex system—a chemical plant, for example—would try to perform some form of sensitivity analysis to determine an answer to (a), and follow it with an attempt to identify optimal values for the parameters identified. As it stands, experimental applications of ILP usually have not used any such systematic approach. Typically, parameters are given "factory-supplied" default values, or values obtained from a limited investigation of performance across a few pre-specified values. The immediate consequence of this is that it is not clear if better models could have been obtained if some form of parameter selection and optimisation had been performed. A measure of the unsatisfactory state of affairs is obtained by considering whether it would be acceptable for a chemical engineer to take a similar approach when attempting to identify optimal operating conditions to maximise the yield of his plant.

Here take up the questions of screening and optimisation of parameters directly with the only restrictions being that parameter and goodness values are quantitative in nature. The methods we use have origins in optimising industrial processes (Box and Wilson, 1951) and been developed under the broad area concerned with the design and analysis of experiments. This area is concerned principally with discovering something about a black-box system by designing deliberate changes to the system's input variables, and analysing changes in its output response. The representation of a system is usually as shown in Figure 1(a) (from Montgomery, 2005). The process being modelled transforms some input into an output that is characterised a measurable response y. The system has some controllable factors, and some uncontrollable ones and the goals of an experiment could be to answer questions like: which of the controllable factors are most influential on y; and what levels should these factors be for y to reach an optimal value. The relevance of the setting to the ILP problem we are considering here will be evident in Section 2.

1. In Srinivasan (2001a), experience gained from applications of ILP to problems in biochemistry were used to extract some guiding principles of relevance to these problems for any ILP application.

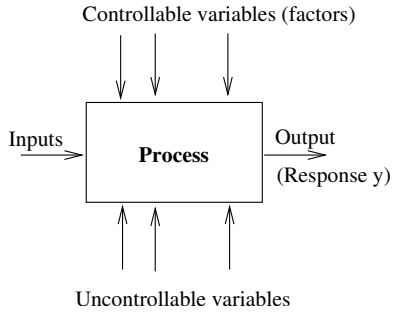


Figure 1: Model of a system used in experimental design (from Montgomery, 2005). The process can be a combination of systems, each modelled by some input-output behaviour.

There are a wide variety of techniques developed within the area of experimental design: we will be concentrating here on some of the simplest, based around the use of regression models. Specifically, using designed variations of input variables, we will use a stepwise linear regression strategy to identify variables most relevant to the ILP system's output response. This resulting linear model, or response surface, is then used to change progressively the values of the relevant variables until a locally optimal value of the output is reached. We demonstrate this approach empirically on some ILP benchmarks.

The rest of this paper is organised as follows. Section 2 describes a black-box view of ILP systems that we adopt in this paper. Section 3 describes work in ILP and the broader area of Machine Learning related to the goals of this paper. Section 4 describes details of techniques from the field of experimental design that are relevant to the paper. Section 5 describes, first, two empirical studies. The studies demonstrate how, for a given set of inputs, parameter screening and selection using designed experiments yields a better model than simply using default values, or performing an exhaustive combination of pre-determined values for parameters. They also demonstrate how, if inputs are changed, then both the set of relevant parameters and their values can change. These experiments are then followed up with others that use six other well-known benchmark data sets. The results confirm the findings from the primary investigation; and also demonstrate the relevance of this work to the controlled comparisons of ILP systems. Section 6 concludes the paper. The paper is accompanied by two appendices that provide standard material from the literature concerned with the construction of linear models, and with specific aspects of the optimisation method used here.

2. An ILP System as a Black-Box

Inductive Logic Programming (ILP) has been largely characterised by two classes of programs. The first, predictive ILP, has been concerned with constructing discriminative models (sets of rules; or first-order variants of classification or regression trees) for distinguishing accurately amongst two sets of examples ("positive" and "negative"), or more generally, amongst examples classified into one of several classes. The second category of ILP programs, descriptive ILP, has been concerned with generative models of relationships that hold amongst the background knowledge and examples. This latter category includes programs that identifies logical constraints in a database (DeRaedt and

Bruynooghe, 1992) and more recently, programs that capture complex probabilistic relationships amongst objects (the area of statistical relational learning: see Getoor and Taskar, 2007).

While much effort has been invested in clarifying, in the form of a specification, what constitutes different kinds ILP systems (see, for example Muggleton and Raedt, 1994), in this paper, we take an engineer's view. In this, an ILP implementation is simply a machine learning (ML) system that, given some inputs—in usual ILP terminology, background knowledge and examples—and settings for parameters, some of which are under the control of the engineer, produces an output model by performing some form of optimisation (see Figure 2). For example, many ILP systems that explore the space of alternatives imposed by the inverse entailment setting proposed in Muggleton (1995) could be seen as performing a form of discrete optimisation, using some approximation to a branch-and-bound search procedure. The task of the system engineer is then to tune the parameters under his or her control to enable the system to return the best performance.[2] In Srinivasan (2001b), for example, it is demonstrated how widely varying performance can be obtained by varying a single parameter (the minimum accuracy of clauses found in a search).

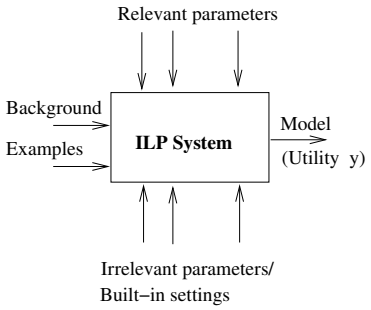


Figure 2: An system engineer's view of an ILP system. We are assuming here that "Background" includes syntactic and semantic constraints on acceptable models. "Built-in settings" are the result of decisions made in the design of the ILP system. An example is the optimisation function used by the system.

The immediate difficulty is, of course, that it is usually impractical to examine the system's performance by enumerating every possible combination of values for the controllable parameters. With ILP systems there are two further difficulties. First, it may often not be known beforehand which parameters are actually relevant to system for the problem being solved. The system Aleph (Srinivasan, 1999) provides perhaps the most clear instance of this: see Figure 3. Second, models constructed, and hence system performance, can vary even if all inputs and parameters have fixed values: for example, the system may use a search strategy that employ some random choices (Zelezny et al., 2002 provides an example of such a strategy).

2. This is different to improving the optimisation procedure performed by the system itself. Rather, it is concerned with enabling the existing optimisation procedure find better results, usually by changing the space of alternatives in some principled manner. It is beyond the engineer's remit to alter either the system's inputs or its optimisation criterion as a means of improving system performance.

```
1. The following parameters can affect the size of the search space:
      i, clauselength, nodes, minpos, minacc,
      noise, explore, best, openlist, splitvars.
2. The following parameters affect the type of search:
      search, evalfn, refine, samplesize.
3. The following parameters have an effect on the speed of execution:
      caching, lazy_negs, proof_strategy, depth,
      lazy_on_cost, lazy_on_contradiction, searchtime, prooftime.
4. The following parameters alter the way things are presented to the user:
      print, record, portray_hypothesis, portray_search,
      portray_literals, verbosity,
5. The following parameters are concerned with testing theories:
      test_pos, test_neg, train_pos, train_neg.
```

Figure 3: A categorisation of some of the parameters of the ILP system Aleph (reproduced from Srinivasan, 1999). Not all of these are relevant to every problem being solved.

3. Related Work on Parameter Screening and Optimisation

Within ILP, no significant attention has been paid to the problem of parameter screening or optimisation. Reports in the literature rarely contain any discussion of sensitive parameters of the system or their values. Of 100 experimental studies reported in papers presented between 1998 and 2008 to the principal conference in the area, none attempt any form of screening for relevant parameters. 17 describe settings for some pre-selected parameters—usually one—from performance estimates obtained during an enumerative search over some small set of possible values (that is, effectively using the wrapper approach of Kohavi and John, 1995). 38 reports, however, mention values assigned to *some* parameters, without elucidating how these values were reached (on occasions, these were just the default values provided by the system). The work in Srinivasan (2001b) can be seen as addressing the question of optimal values for several input parameters somewhat indirectly by first constructing an "operating characteristic curve" that describes the performance of an ILP system across a range of values for the relevant parameters. While no method is proposed for identifying the parameters themselves, the characteristic curve provides a way of optimally selecting amongst models, provided model goodness is restricted to a specific class (that of cost functions that are linear in the error-rates). . Since each model is obtained from a particular combination of values for relevant parameters, we are able to identify the values that resulted in the best model for the task. The procedure is somewhat reminiscent of putting the cart before the horse though, requiring us to identify all models on the characteristic curve first.

Turning to the broader literature in ML, we have not been able to uncover any reports explicitly concerned with screening for relevant parameters. There have, however, been some reports of techniques for optimal assignment for a set of relevant parameters. Bengio (2000) is most closely related to the optimisation goals of this paper, in that it presents a methodology to optimise several parameters (Bengio and the following papers call them hyperparameters, to avoid confusion with the statistical term), based on the computation of the gradient of a model selection criterion expressed in terms of the parameters. The main restriction is that this criterion must be a known, continuous and

differentiable function of the parameters (almost) everywhere. While Bengio assumes a training criterion that is quadratic in the parameters, Keerthi et al. (2006) present a fast method for computing the gradient of a validation function with respect to parameters for a range of SVM models. Their method only needs a single linear system of equations to be solved. Unfortunately, it is not possible to directly adapt these methods to ILP systems. In almost all ILP settings, the training criterion cannot be even expressed in closed form, let alone being a differentiable and continuous function of the parameters. That is, what can be done at best is to treat the ILP system is a black box (as we have done in the previous section) and empirically observe variations in its response to changes in the values of the parameters.

Methods have been developed that use such empirically observed responses to direct the assignment of values to relevant parameters. The seminal work in Kohavi and John (1995) introduced the "wrapper" approach to parameter optimisation, in which responses from a ML system are used to direct a heuristic search through combinations of possible values for the parameters. For tractability, these values are discretised *a priori*, and approach essentially performs a sub-optimal search through a finite space of what are called k-level full-factorial designs in this paper (the k refers to the number of discrete values: more on such designs in the next section). In this paper, we use a exhaustive search through such a space as a baseline for comparison against a gradient-based optimisation method. The results from the exhaustive search clearly represent an upper-bound on the results achievable by any heuristic search through the same space.

The work described in Baz et al. (2007) is concerned with determining parameter values that minimise the computation time of mixed integer linear programming (MILP) systems. As with the ILP systems we consider here, the MILP solvers have many parameters, with no clear relationships known amongst them; and the objective function cannot be expressed as a closed form function of these parameters. Their approach is to select an initial set of values for the hyperparameters using some sampling design. The response of the MILP solver is then obtained, from which a ML system is used to construct a model relating the response to parameter values. This model is then used to suggest new values for some subset of the parameters. For example, if the model used is a regression tree, then the parameters used in the top 2 levels of the tree (the choice of 2 is arbitrary, but no method is proposed for automating this choice) are selected and additional sets of values obtained for the parameters (the exact procedure of how this is done is not elaborated upon). The set that results in the best performance is returned. This work can be seen as a case of exhaustive enumeration of responses in a k-level full factorial design, followed by a single stage of *ad hoc* non-linear regression-based parameter screening and optimisation.

The problem of screening and tuning of parameters to optimise a system's performance has been studied extensively in areas of industrial engineering, using results obtained from the design and analysis of experiments. It is our intention in this paper to investigate the application of techniques developed in these areas to ILP, and we summarise some of the relevant ideas next.

4. Design and Analysis of Experiments

The area broadly concerned with the design of experiments (DOE) deals with devising deliberate variations in the values of input variables, or *factors*, and analysing the resulting variations in a set of one or more output, or *response*, variables. The objectives of conducting such experiments are usually: (a) Understand how variation or uncertainty in input values affects the output value. The goal here is the construction of robust systems in which a system's output is affected minimally

by external sources of variation; and (b) Maximise (or minimise) a system's performance. In turn, these raise questions like the following: which factors are important for the response variables; what values should be given to these factors to maximise (or minimise) the values of the response variables; what values should be give to the factors in order that variability in the response is minimal, and so on.

In this paper, we will restrict ourselves to a single response variable and the analysis of experimental designs by multiple regression. It follows, therefore, that we are restricted in turn to quantitative factors only. Further, by "experimental design" we will mean nothing more than a selection of points from the factor-space, in order that a statistically sound relationship between the factors and the response variable can be obtained. Each factor-level combination will constitute an experiment, and a design will therefore require us to specify the experiments and, if necessary, the number of replications of each experiment.

4.1 Screening using Factorial Designs

We first consider designs appropriate for screening. By this, we mean deciding which of a set of potentially relevant factors are really important, statistically speaking. The usual approach adopted is what is termed a 2-level factorial design. In this, each factor is taken to have just two levels (encoded as "-1" and "+1", say),[3] and the effect observed on the response variable of changing the levels of each factor. It is evident that with k factors, this will result in 2^k experiments, each of which may need to be repeated in case there is some source of random variation in the response variable. For example, with two factors, conducting a 2^2 full factorial design will result in a table such as the one shown in Figure 4

We are then able to construct a regression model relating the response variable to the factors:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1 x_2.$$

The model describes the effect of each factor $x_{1,2}$ and interactive effect $x_1 x_2$ of the two factors on y.[4] It is usual also to add "centre points" to the design in the form of experiments that obtain values for y for $x_1 = 0$ and $x_2 = 0$. The results of these experiments will not contribute to estimation of the coefficients $b_{1,2,3}$ (since the x_i are all 0s), but allows us to obtain a better estimate for the value of b_0. Further, it is also the case that with a 2-level full factorial design only linear effects can be estimated (that is, the effect of terms like x_i^2 cannot be obtained: in general, a n^{th} order polynomial will require $n+1$ levels for each factor). In this paper, we will use the coefficients of the regression model to guide the screening of parameters: that is, parameters with coefficients significantly different from 0 will be taken to be relevant (more on this in Appendix A).

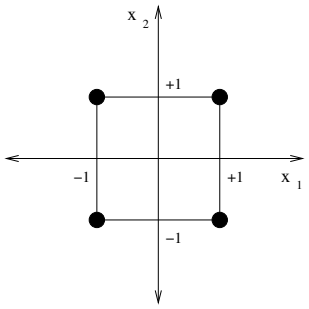
Clearly, the number of experiments required in a full factorial design constitute a substantial computational burden, especially as the number of factors increase. Consider, however, the role these experiments play in the regression model. Some are necessary for estimating the effects of each factor (that is, the coefficients of $x_1, x_2, x_3, \ldots$: usually called the "main effects"), others for estimating the coefficients for two-way interactions (the coefficients of $x_1 x_2, x_1 x_3, \ldots$) , others for three-way interactions ($x_1 x_2 x_3, \ldots$) and so on. However, in a screening stage, all that we wish to do is to identify the main effects. This can usually be done with fewer than the 2^k experiments needed

3. One way to achieve the coded value x of a factor X is as follows. Let X^- and X^+ be the minimum and maximum values of X (these are pre-specified). Then $x = \frac{X - (X^+ + X^-)/2}{(X^+ - X^-)/2}$.

4. Interaction effects happen if the effect of a factor, say X_1 on the response depends on the level of another factor X_2.

Expt.	**Factor** x_1	**Factor** x_2	**Response** y
E1	-1	-1	...
E2	-1	+1	...
E3	+1	-1	...
E4	+1	+1	...

(a)



(b)

Figure 4: (a) A 2-level full factorial design for two factors; and (b) a graphical representation of the design.

for a full factorial design with k factors. The result is a 2-level "fractional" factorial design. Figure 5 below illustrates a 2-level fractional factorial design for 3 factors that uses half the number of experiments to estimate the main effects (from Steppan et al., 1998).

Expt.	x_1	x_2	x_3	y
E1	-1	-1	-1	...
E2	-1	-1	+1	...
E3	-1	+1	-1	...
E4	-1	+1	+1	...
E5	+1	-1	-1	...
E6	+1	-1	+1	...
E7	+1	+1	-1	...
E8	+1	+1	+1	...

Expt.	x_1	x_2	x_3	y
E2	-1	-1	+1	...
E3	-1	+1	-1	...
E5	+1	-1	-1	...
E8	+1	+1	+1	...

Figure 5: A full 2-level factorial design for 3 factors (left) and a "half fraction" design (right).

The experiments in the fractional design have been selected so that $x_1x_2x_3 = +1$. Closer examination of the table on the right will make it clear that the following equalities also hold for this table: $x_1 = x_2x_3$; $x_2 = x_1x_3$; and $x_3 = x_1x_2$. That is, main effects and interaction terms are con-

founded with each other. This has some direct implications when constructing regression models using the fractional table. In effect, instead of the full regression model:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_1x_2 + b_5x_1x_3 + b_6x_2x_3$$

we are reduced to obtaining the following model:

$$y = b_0 + b'_1(x_1 + x_2x_3) + b'_2(x_2 + x_1x_3) + b'_3(x_3 + x_1x_2).$$

In fact, a regression program will be unable, for example, to distinguish the regression model above from this one:

$$y = b_0 + b''_1x_1 + b''_2x_2 + b''_3x_3$$

or even this:

$$y = b_0 + b'''_1x_1 + b'''_2x_2 + b'''_3x_1x_2.$$

The b''_i and b'''_i will differ from the b'_i by a factor of 2, but this will not change the model's fit of the data, since the corresponding independent variables in the regression equation would be halved (x_1 instead of $x_1 + x_2x_3$ and so on). Thus, the price for fractional experiments is therefore, that we will in general, be unable to distinguish the effects of all the terms in the full regression model. However, if it is our intention—as it is in the screening stage—only to estimate the main effects (such models are also called "first-order" models), then we can ignore interactions (see Figure 6). Main effects can be estimated with a table that is a fraction required by the full factorial design: for example, the half fraction in Figure 5 is sufficient to obtain a regression equation with just the main effects x_1, x_2 and x_3.[5]

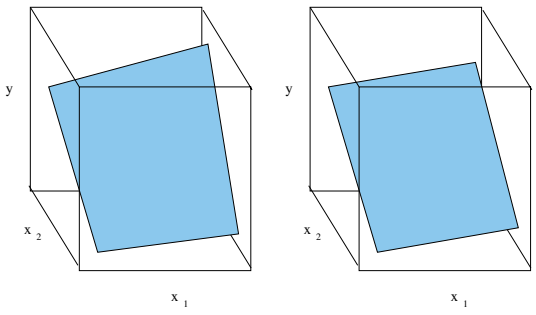


Figure 6: A surface with a "twist" arising from interactions between the factors (left) and a planar approximation that ignores this twist (right). For the purpose of estimating the main effects, the surface on the right is adequate, as it shows that x_2 has a much bigger effect than x_1 on the response y (we are assuming here that x_1 and x_2 represent coded values on the same scale).

5. This is apparent from the fact that n distinct data points are needed to fit a regression model with n terms. Thus, when fitting a model with just x_1, x_2, and x_3, we need 4 data points.

More details on fractional designs are provided in Appendix B. We use the techniques and results described there to direct the screening of factors by focusing on a linear model that contains the main effects only:

$$y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_k x_k.$$

Depending on the number of factors, this can be done with a fractional designs of "Resolution III" or above (see Appendix B). Standard tests of significance can be performed on each of the coefficients $b_1, b_2, \ldots, b_k$ to screen factors for relevance (the null and alternative hypotheses in each case are $H_0 : b_i = 0$ and $H_1 : b_i \neq 0$). In fact, this test is the basis for inclusion or exclusion of factors by stepwise regression procedures (see Appendix A). Using such a procedure would naturally return a model with only the relevant factors (the use of stepwise regression is also the preferred method for sensitivity analysis suggested at the end of the extensive survey in Helton et al., 2006).

4.2 Optimisation Using the Response Surface

Suppose screening in the manner just described yields a set of k relevant factors from a original set of n factors (which we will denote here as $x_1, x_2, \ldots, x_k$ for convenience). We are now in the position of describing the functional relationship between the expected value of the response variable and the relevant factors, by the "response surface":

$$E(y) = f(x_1, x_2, \ldots, x_k).$$

Usually, f is taken to be some low-order polynomial, either a first-order model involving only the main effects $x_1, x_2, \ldots$ (recall that if stepwise regression procedure is used at the screening stage, then this is the model that would be obtained):

$$y = b_0 + \sum_{i=1}^{k} b_i x_i$$

or a second-order model involving quadratic terms like $x_1^2, x_2^2, \ldots$ and linear interaction terms like $x_1 x_2, x_1 x_3, \ldots$:

$$y = b_0 + \sum_{i=1}^{k} b_i x_i + \sum_{i=1}^{k} b_{ii} x_i^2 + \sum_{i=1}^{k} \sum_{j>i} b_{ij} x_i x_j.$$

Clearly, if first-order models are adequate (this can be checked by an analysis of how well the model fits the data: see Appendix A) then much of the effort expended in the screening stage can be re-used (for example, we can use the model constructed by stepwise regression as the response surface model). A second-order model, on the other hand, will require experiments involving additional levels for each factor, and some effort has been invested in the literature on determining these levels. Since first-order models are all that are used in this paper, we do not pursue this further here, and refer the reader to a standard text like Montgomery (2005) for more details.

The principal approach adopted in optimising using the response surface is a sequential one. First, a local approximation to the true response surface is constructed, using a first-order model. Next, factors are varied along a path that improves the response the most (more on this in a moment). Experiments are conducted along this direction and the corresponding responses obtained until no

further increase in the response is observed. At this point, a new first-order response surface is constructed, and the process repeated until it is evident that a first-order model is inadequate (or no more increases are possible). If the fit of the first-order model is poor, a more detailed model is then obtained—usually a second-order model, using additional levels for factors—and its stationary point obtained. The basic idea is illustrated in Figure 7 (from Montgomery, 2005).

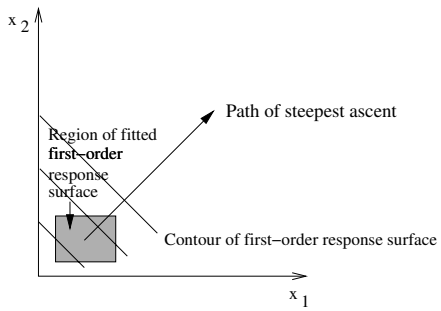


Figure 7: Sequential optimisation of the response surface using the path of steepest ascent. A first-order response surface is obtained in the shaded region. The factors are then changed to move along a direction that gives the maximum increase in the response variable.

Now, we can view the response y to be given by a scalar field f that at each point $x_1, x_2, \ldots, x_k$ gives the response $f(x_1, x_2, \ldots, x_k)$. Then, from standard vector calculus, the gradient of f at the point gives the direction in which the response will change most quickly (that is, the direction of steepest ascent: see Appendix B). This gradient, usually denoted ∇f, is given by $\left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_k}\right)$. The sequential optimisation of the response surface just described involves calculating the gradient of the first-order model at the centre, or origin, of the experimental design ($x_1 = x_2 = \cdots = 0$). For a model of the form $f(x_1, \ldots, x_k) = b_0 + b_1 x_1 + \cdots + b_k x_k$, ∇f is simply $(b_1, \ldots, b_k)$. For convenience, let us take b_1 to have the largest absolute value. Then, along the direction of ∇f, a unit change in x_1 will result in a change of b_2/b_1 units of x_2, b_3/b_1 units of x_3 and so on. Sequential response optimisation proceeds by starting at the origin and increasing the x_i along ∇f until increases in the response y is observed. Each such increase results in a new experiment to be performed (see Figure 8, for an example with 3 factors).

4.3 Screening and Optimisation for ILP

We are now in a position to put together the material in the previous sections to state more fully a procedure for screening and optimisation of parameters for an ILP system:

SO: Screen quantitative parameters using a two-level fractional factorial design, and optimise values using the response surface.

ScreenFrac. Screen for relevant parameters using the following steps:

*S*1. Decide on a set of n quantitative parameters of the ILP system that are of potential relevance. These are the factors x_i in the sense just described. Take some quantitative summary of the model constructed by the system—for example, some estimate

Expt.	**Factor** x_1	**Factor** x_2	**Factor** x_3	**Response** y
E9	0	0	0	...
E10	δ	$\frac{b_2}{b_1}\delta$	$\frac{b_3}{b_1}\delta$	...
E11	2δ	$\frac{2b_2}{b_1}\delta$	$\frac{2b_3}{b_1}\delta$	...
E12	3δ	$\frac{3b_2}{b_1}\delta$	$\frac{3b_3}{b_1}\delta$	...
...	...	...	...	...

Figure 8: Sequential experiments that obtain new values for y by moving in the direction of the gradient to $b_0 + b_1x_1 + b_2x_2 + b_3x_3$. Experiments E1–E8 are as in Figure 5.

of its predictive accuracy—as the response variable y (we will assume here that we wish to maximise the response).

*S*2. Decide on on two levels ("low" and "high" values) for each of the factors. These are then coded as ± 1.

*S*3. Devise a two-level fractional factorial design of Resolution III or higher, and obtain values of y for each experiment (or replicates of values of y, if so required).

*S*4. Construct a first-order regression model to estimate the role of the main effects x_i on y. Retain only those factors that are important, by examining the magnitude and significance of the coefficients of the x_i in the regression model (alternatively, only those factors found by a stepwise regression procedure are retained: see Appendix A).

OptimiseRSM. Optimise values of relevant parameters using the following steps:

*O*1. Construct a first-order response surface using the relevant factors only (this is not needed if stepwise regression was used at the screening stage). If no adequate model is obtained, then return the combination of factor-values that gave the best response at the screening stage. Otherwise go to Step O2.

*O*2. Progressively obtain new values for y by changing the relevant parameters along the gradient to the response surface. Stop when no increases in y are observed.[6]

*O*3. If needed, construct a new first-order response surface. If this surface is adequate, then return to Step O2. Otherwise, go to Step O4.

*O*4. If needed, construct a second-order response surface. Return the optimum values of the relevant factors using the second-order surface, or from the last set of values from Step *O*2.[7]

6. In practice, this is taken to mean that no increases have been observed for some number of consecutive experimental runs: the so-called "k-in-a-row" stopping rule.

7. We note that the use of gradient ascent in this manner is only capable of finding local maxima in y values. A question is raised about what is to be done if the local maximum found in this manner is *lower* than a response value known already—for example, from an experiment from the screening stage. A modification would be return the combination of factor-values that give the best y value obtained over all experiments. This would be at variance with standard response-surface optimisation, and we do not consider it here.

We contrast OptimiseRSM with the multi-level full factorial design below, which has been used on a few occasions within the ILP literature:

OptimiseFact. Optimise values of relevant parameters using the following steps:

- $O1'$. Decide on on multiple levels for each of the relevant factors.
- $O2'$. Devise a full factorial design by combing each of the levels of the factors against those of the others. For each such combination, obtain values of y for each experiment (or replicates of values of y, if so required).
- $O3'$. Select the combination of values that yielded the highest value of y (including those obtained at the screening stage).

This procedure, a multi-level full factorial design, is the basis of the wrapper-based optimisation method in Kohavi and John (1995), recast in the terminology of experimental design. A simplified analysis gives us some feel of the complexity of **SO**. **SO** conducts some fraction of 2^n experiments in the *ScreenFrac* stage, followed by those conducted in OptimiseRSM. Suppose we always conduct a 2^{n-p}-fractional design at the screening stage, and that this stage results in no more than r variables being selected as relevant. Further, let each round of sequential optimisation consist of s experiments in Step O2. Let there be m such rounds of sequential optimisation, each followed by a new first-order model in Step O3 (since there are r variables, building this model will require an additional $r+1$ experiments). Finally a second-order model is constructed (Step O4), using a central composite design. Then the total number of experiments conducted by **SO** is: 2^{n-p} (screening) + ms (sequential optimisation) + $(m-1)(r+1)$ (new first-order models) + $2r+1$ (second-order model). In the case that only one round of sequential experimentation is performed (that is, $m = 1$) and no additional first- or second-order models are constructed, the number of experiments is simply $2^{n-p}+s$. It is evident that a procedure **SO**$'$ that employs *ScreenFrac* followed by *OptimiseFact* would always perform $2^{n-p}+l^r$ experiments (assuming, for simplicity, that all relevant factors are taken to have l levels during the optimisation stage). This is no more than $2^{n-p}+l^n$.

Clarification is needed on the following additional questions:

1. What is to be done if a first-order model cannot be constructed in the screening stage? The usual approach in response-surface methodology is then to examine a second-order response surface. We take the position in this paper that none of the parameters are especially relevant, and simply assign them their default values.

2. Is the value of the response variable obtained after optimisation a good estimate of the performance of the ILP system? We distinguish here between the following two performance estimates: (a) The estimate of the ILP system's accuracy used *during* parameter optimisation; and (b) The estimate of the ILP systems's accuracy *after* parameter optimisation. Clearly, if (a) and (b) are the same, we run the risk that the value obtained will be an optimistic estimate of the ILP system's accuracy. We attempt to minimise this by ensuring that the estimate (b) is not, in any manner influenced by the estimate (a) (details are in Section 5.3 below). For clarity, we will call estimate (a) the experimental performance of the ILP system and estimate (b) it's final performance. Of course, overuse of an estimate—whether experimental or otherwise—can result in overfitting: especially as the number of experiments increase. That is, we will sooner or later find an experiment that "looks good" simply by chance. By

employing a gradient ascent method, we are clearly attempting to minimise the number of experiments by moving along the direction of maximum change. Experimental evidence of overfitting usually also comes to light by increasing the number of data sets on which the procedure is tested (see Section 5.4).

3. What is to be done if the local maximum reached, either by optimising the response-surface or in the multi-level factorial design is not unique? That is, a number of different parameter settings return a maximal value, and we take all of these as being equally likely. The final performance values will thus be the average of the final performance values from each of these settings.

5. Empirical Evaluation

We will first briefly state the aims of the experimental evaluation. Descriptions of the materials and our experimental methodology will follow. We will finally present detailed experimental results.

5.1 Aims

Our aim here is to demonstrate the utility of the screening and optimisation procedure **SO** that we have described in Section 4.3 (that is, **SO** is *ScreenFrac* followed by *OptimiseRSM*). We assess this utility by comparing the ILP system when it employs **SO** against the performance of the system when it uses one of following alternatives: **Default**, in which no screening or optimisation is performed and default values provided for all parameters are used; and **SO′**, in which screening is performed as in **SO**, but a multi-level full factorial design is used for optimisation (that is, **SO′** is *ScreenFrac* followed by *OptimiseFact*). Specifically, we intend to investigate the following conjectures:

*C*1**.** Using **SO** is better than using **Default**; and

*C*2**.** Using **SO** is better than using **SO′**.

In both cases, "better" is short-form for stating that an ILP system that uses **SO** has better final performance; or in the case of ties, requires fewer experiments than the alternative.

5.2 Materials

In this section we explain (i) the two datasets, (ii) the systems for experimental design and ILP and (iii) the hardware employed in our experiments.

5.2.1 DOMAINS

The investigation is conducted first on the well-studied ILP biochemical problems concerned with identifying mutagenic and carcinogenic chemicals. Although we will extend it later to other data sets used in the literature, we have selected to focus on these problems first since they constitute perhaps the most commonly used inputs for demonstrating the performance of ILP systems. The data have been described extensively elsewhere (for example, see King et al., 1996 for mutagenesis; and King and Srinivasan, 1996 for carcinogenesis) and we refer the reader to these reports for details. For each application, the input to an ILP can vary depending on the background information used. We

investigate the conjectures $C1$ and $C2$ with minimal and maximal amount of background knowledge contained in these benchmarks. That is:

Mutagenesis. We consider background information in the sets M0 and M0–M4, descriptions of which are reproduced below from Srinivasan (2001b):

M0. Molecular description at the atomic level. This includes the atom and bond structure, the partial charges on atoms, and arithmetic constraints (equalities and inequalities). There are 5 predicates in this group;

M1. Structural properties identified by experts as being related to mutagenic activity. These are: the presence of three or more benzene rings, and membership in a class of compounds called acenthrylenes. There are 2 predicates in this group;

M2. Chemical properties identified by experts as being related to mutagenic activity, along with arithmetic constraints (equalities and inequalities) The chemical properties are: the energy level of the lowest unoccupied molecular orbital ("LUMO") in the compound, an artificial property related to this energy level (see Debnath et al., 1991), and the hydrophobicity of the compound. There are 6 predicates in this group;

M3. Generic planar groups. These include generic structures like benzene rings, methyl groups, *etc*., and predicates to determine connectivity amongst such groups. There are 14 predicates in this group; and

M4. Three-dimensional structure. These include the positions of individual atoms, and constraints on distances between atom-pairs. There are 2 predicates in this group.

Carcinogenesis. We consider background information in the sets C0 and C0–C3, descriptions of which reproduced below, once again from Srinivasan (2001b):

C0. Molecular description at the atomic level. This is similar to M0 above and is comprised of 5 predicates;

C1. Toxicity properties identified by experts as being related to carcinogenic activity, and arithmetic constraints. These are an interpretation of the descriptions in Ashby and Tennant (1991), and are contained within the definitions of 5 predicates;

C2. Short-term assays for genetic risks. These include the *Salmonella* assay, in-vivo tests for the induction of micro-nuclei in rat and mouse bone marrow *etc*. The test results are simply "positive" or "negative" depending on the response and are encoded by a single predicate definition; and

C3. Generic planar groups. These are similar to M3 above, extended to 30 predicate definitions.

We will henceforth refer to background knowledge with the definitions in M0 (respectively, C0) as B_{min} and with the definitions in M0–M4 (respectively, C0–C3) as B_{max}.

5.2.2 ALGORITHMS AND MACHINES

Experimental design and regression models for screening and the response surface are constructed by the procedures made available by the authors of Steppan et al. (1998). The ILP system used in all experiments will be Aleph (Srinivasan, 1999). The programs are executed on a IBM Thinkpad (T43p), equipped with an Intel 2 GHz Pentium processor with 1 gigabyte of random access memory.

5.3 Method

Our method for the preliminary experiments is straightforward:

> For each problem (Mutagenesis and Carcinogenesis) and each level of background knowledge (B_{min} and B_{max}):
>
> 1. Construct a model with the ILP system using default values for all parameters of the ILP system. Call this model **ILP+Default**.
> 2. Select a set of n quantitative parameters of the ILP system as being potentially relevant. Use the procedure *ScreenFrac* described in Section 4.3 to screen this set using a fractional factorial design of Resolution III or higher. Let this result in a set of relevant variables R.
> 3. Use the procedure *OptimiseRSM* in Section 4.3 to obtain values for variables in R. All other parameters of the ILP system are left at their default values. Construct a model using the ILP system with this set of values. Call this model **ILP+SO**.
> 4. Decide on l levels for each variable in R and use the procedure *OptimiseFact* in Section 4.3 to obtain values for the variables in R. All other parameters of the ILP system are left at their default values. Construct a model using the ILP system with this set of values. Call this model **ILP+SO′**.
> 5. Compare the performance of the ILP system when it produces as output each of **ILP+Default**, **ILP+SO**, and **ILP+SO′** (see the details below).

We follow the preliminary experiments with experiments on additional data sets and with an additional ILP system. The following details concerning the preliminary experiments are relevant:

1. Since the tasks considered here are binary classification tasks, the performance of the ILP system in all experiments will be taken to be the classification accuracy of the model produced by the system. By this we mean the usual measure computed from a 2×2 cross-tabulation of actual and predicted classes of instances. We would like the final performance measure to be as unbiased as possible by the experimental estimates obtained during optimisation. One way is to use a technique of "double" or nested cross-validation. That is, the final performance value is obtained using k-fold cross-validation (the "outer" cross-validation) and experimental performance values during optimisation is the average of a further ("inner") k-fold cross-validation using each of the training data sets from the outer cross-validation. This procedure is computationally expensive. We adopt a simpler alternative: we use a 10-fold cross-validation estimate for the final estimate; and for the experimental estimates we use the average of holdout ("validation" set) estimates on each of the training data sets from the outer cross-validation. Thus, the test data in each of the outer cross-validation folds are not available to the ILP system when performing parameter optimisation.

2. We have no general prescription for the selection of the initial set of n parameters (Step 2). We postpone a discussion of this limitation to Section 5.4. For our experiments we have selected four parameters: C, the maximum number of literals in any acceptable clause constructed by the ILP system; *Nodes*, the maximum number of nodes explored in any single

search conducted by the ILP system; *Minacc*, the minimum accuracy required of any acceptable clause; and *Minpos*, the minimum number of positive examples to be entailed by any acceptable clause. *C* and *Nodes* are directly concerned with the search space explored by the ILP system. *Minacc* and *Minpos* are concerned with the quality of results returned (they are equivalent to "precision" and "support" used in the data mining literature). We propose to examine a two-level fractional factorial design, using the levels shown below (the column "Default" refers to the default values for the factors assigned by the Aleph system, and ± 1 refers to the coded values of the factors):

Factor	**Levels**		
	Default	Low (-1)	High ($+1$)
C	4	4	8
Nodes	5000	5000	10000
Minacc	$+1$	0.75	0.90
Minpos	1	5	10

3. We use a Resolution IV design, that comprises of a randomised presentation of the following 8 experiments (recall the full factorial design will require $2^4 = 16$ experiments):

Expt.	*C*	*Nodes*	*Minacc*	*Minpos*	*Accuracy*
E1	-1	-1	-1	-1	...
E2	-1	-1	$+1$	$+1$	...
E3	-1	$+1$	-1	$+1$	...
E4	-1	$+1$	$+1$	-1	...
E5	$+1$	-1	-1	$+1$	...
E6	$+1$	-1	$+1$	-1	...
E7	$+1$	$+1$	-1	-1	...
E8	$+1$	$+1$	$+1$	$+1$	...

This design was obtained using the software tools for experimental design provided with Steppan et al. (1998). The "Accuracy" column is the experimental performance obtained for each task, and for each of the two sets of background knowledge in order to screen the four variables for relevance. Additional experiments, and corresponding experimental performance values, will be needed in Step 3 to obtain values of the relevant parameters using the response surface. We restrict ourselves to constructing just one first-order regression model for screening, using the stepwise regression procedure provided by the authors of Steppan et al. (1998). This model is taken to approximate the local response surface: we then proceed to change levels of factors along the normal to this surface in the manner described in Figure 8. Experiments are stopped once a maximal value for the response variable is followed by three consecutive runs that yield responses that are no higher.

4. In the event that all four parameters chosen are relevant, the step of obtaining parameter values using a multi-level full factorial design (Step 5) would require conducting l^4 experiments. We will take $l = 5$, which means that, in the worst case, no more than 625 experiments will be conducted to obtain model **ILP+SO′**. Inspired by the choices made for a so-called "Central

Composite" (or CC) design (Montgomery, 2005), we will take the (coded) levels to be $0, \pm 1$, and $\pm\sqrt{2}$.

5. Comparisons of models will be done on the basis of their final performance estimates (see (1) above) (parameter values are obtained from the experimental estimates). In the event of ties, then the model requiring fewer experiments will be preferred. That is, a model is represented by the pair (A, E) (denoting estimated accuracy and number of experiments required to identify the model). Comparisons are then based on the usual definition of a lexicographic ordering on such tuples.

Further, since it is of particular relevance to ILP practitioners, we also test for statistical differences between the accuracies of **ILP+SO** and **ILP+ Default** using results on six additional data sets used in the ILP literature, and separately, by using two different ILP systems. The relevant statistical test is the Wilcoxon signed-rank test (Siegel, 1956). This is a non-parametric test of the null hypothesis that there is no significant difference between the median performance of a pair of algorithms. The test works by ranking the absolute value of the differences observed in performance of the pair of algorithms. Ties are discarded and the ranks are then given signs depending on whether the performance of the first algorithm is higher or lower than that of the second. If the null hypothesis holds, the sum of the signed ranks should be approximately 0. The probabilities of observing the actual signed rank sum can be obtained by an exact calculation (if the number of entries is less than 10), or by using a normal approximation. We note that the comparing a pair of algorithms using the Wilcoxon test is equivalent to determining if the area under the ROC curves of the algorithms differ significantly (Hand, 1997).

5.4 Results and Discussion

We present first the results concerned with screening for relevant factors. Figure 9 show responses from the ILP system for the preliminary experiments conducted for screening using the fractional design described under "Methods". The sequence of experiments following this stage for optimising relevant parameter values using: (a) the response surface; and (b) a multi-level full factorial design are in Figures 10 and 11. Finally, a comparison of the three procedures **ILP+Default**, **ILP+SO**, and **ILP+SO′** is in Figure 12. It is this last tabulation that is of direct relevance to the experimental aims of this paper, and we note the following: (1) Although no experimentation is needed for the use of default values, the model obtained with **ILP+Default** usually has the lowest predictive accuracies (the exception is Carcinogenesis, with B_{min});[8] (2) The classification accuracy of **ILP+SO** is never lower than that of any of the other methods; (3) When the classification accuracies of **ILP+SO** and **ILP+SO′** are comparable, the number of experiments needed by the former is lower.

Taken together, these observations provide *prima facie* evidence for the conjectures made at the outset of this section, namely:

*C*1**.** Using **SO** is better than using **Default**; and

*C*2**.** Using **SO** is better than using **SO′**.

8. We recall that no adequate first-order regression model was obtained for Carcinogenesis (B_{min}), resulting in default values for all parameters. Both **ILP+SO** and **ILP+SO′** suffer because of the experiments needed for the screening stage.

Expt.	*C*	*Nodes*	*Minacc*	*Minpos*	*Acc*
E1	−1	−1	−1	−1	0.793
E2	−1	−1	+1	+1	0.644
E3	−1	+1	−1	+1	0.763
E4	−1	+1	+1	−1	0.669
E5	+1	−1	−1	+1	0.757
E6	+1	−1	+1	−1	0.728
E7	+1	+1	−1	−1	0.787
E8	+1	+1	+1	+1	0.669

(a) Mutagenesis (B_{min})
$Acc = 0.726 - 0.049\ Minacc - 0.018\ Minpos$

Expt.	*C*	*Nodes*	*Minacc*	*Minpos*	*Acc*
E1	−1	−1	−1	−1	0.911
E2	−1	−1	+1	+1	0.870
E3	−1	+1	−1	+1	0.899
E4	−1	+1	+1	−1	0.899
E5	+1	−1	−1	+1	0.899
E6	+1	−1	+1	−1	0.905
E7	+1	+1	−1	−1	0.905
E8	+1	+1	+1	+1	0.876

(b) Mutagenesis (B_{max})
$Acc = 0.896 - 0.009\ Minpos - 0.008\ Minacc$

Expt.	*C*	*Nodes*	*Minacc*	*Minpos*	*Acc*
E1	−1	−1	−1	−1	0.464
E2	−1	−1	+1	+1	0.461
E3	−1	+1	−1	+1	0.444
E4	−1	+1	+1	−1	0.447
E5	+1	−1	−1	+1	0.457
E6	+1	−1	+1	−1	0.451
E7	+1	+1	−1	−1	0.467
E8	+1	+1	+1	+1	0.461

(c) Carcinogenesis (B_{min})
No adequate model

Expt.	*C*	*Nodes*	*Minacc*	*Minpos*	*Acc*
E1	−1	−1	−1	−1	0.572
E2	−1	−1	+1	+1	0.595
E3	−1	+1	−1	+1	0.507
E4	−1	+1	+1	−1	0.576
E5	+1	−1	−1	+1	0.585
E6	+1	−1	+1	−1	0.526
E7	+1	+1	−1	−1	0.523
E8	+1	+1	+1	+1	0.546

(d) Carcinogenesis (B_{max})
$Acc = 0.554 - 0.028\ Minacc$

Figure 9: Screening results (procedure *ScreenFrac* in Section 4.3). *Acc* refers to the estimated accuracy of the model. The regression model is built using the "Autofit" option provided in Steppan et al. (1998). This essentially implements the stepwise regression procedure described in Appendix A. *Acc* refers to the experimental (validation-set) performance of the ILP system. Note that no adequate model is obtained in (c), meaning that the coefficients of all variables have values that are statistically insignificant. In this case, no further optimisation is performed, and all parameters are left at their default values.

We now turn to some broader implications of these results, enumerated in order of seriousness to current ILP practice:

1. The results suggest that default levels for factors need not yield optimal models for all problems, or even when the same problem is given different inputs (here, different background knowledge). This means that using ILP systems just based on default values for parameters—the accepted practice at present—can give misleading estimates of the best response possible from the system. This is illustrated in Figure 13, which shows estimated accuracies on other data sets reported in the literature that also use the Aleph system with default values for all parameters (these data sets have been used widely: see, for example, Landwehr et al., 2006 and Muggleton et al., 2008). Taken with our previous results for the mutagenesis and carcinogenesis data (we will only use the B_{max} results, as these are the results used in the literature), we are now able to make some statements of statistical significance. Figure 14 shows, across the 8 data sets, differences between the optimised and default models. The probability of

Expt.	Coded Values		Natural Values		*Acc*
	Minacc	*Minpos*	*Minacc*	*Minpos*	
E9	0	0	0.83	8	0.769
E10	−0.50	−0.18	0.79	7	0.793
E11	−1	−0.36	0.75	7	0.781
E12	−1.50	−0.54	0.71	6	0.781
E13	−2.00	−0.72	0.67	6	0.692

(a) Mutagenesis (B_{min})

Expt.	Coded Values		Natural Values		*Acc*
	Minpos	*Minacc*	*Minpos*	*Minacc*	
E9	0	0	8	0.83	0.899
E10	−0.50	−0.42	7	0.79	0.899
E11	−1.0	−0.84	5	0.76	0.911
E12	−1.50	−1.26	4	0.73	0.899
E13	−2.00	−1.68	3	0.70	0.893
E14	−2.50	−2.10	2	0.67	0.751

(b) Mutagenesis (B_{max})

Expt.	Coded Value	Natural Value	*Acc*
	Minacc	*Minacc*	
E9	0	0.83	0.553
E10	−0.50	0.79	0.572
E11	−1	0.75	0.595
E12	−1.50	0.71	0.582
E13	−2.00	0.67	0.592
E14	−2.50	0.63	0.598
E15	−3.00	0.60	0.605
E16	−3.50	0.56	0.609
E17	−4.00	0.52	0.539
E18	−4.50	0.49	0.539
E19	−3.50	0.45	0.539

(c) Carcinogenesis (B_{max})

Figure 10: Optimisation using the response surface (procedure *OptimiseRSM* in Section 4.3). In each case, the response surface used is the first-order regression model found by stepwise regression at the screening stage (shown in Figure 9). Parameters are varied along the path of steepest ascent of experimental performance values for the response variable. Experiments are stopped once a maximal value for the response variable is followed by three consecutive runs that yield responses that are no higher. No optimisation is performed for Carcinogenesis (B_{min}) since no adequate first-order response surface was found.

obtaining these results, under the hypothesis that the optimised and default procedures have equivalent performance (correctly, that the median difference between their accuracies is 0) is 0.02. In fact, since our research hypothesis is evidently directional (that accuracy of optimised models is higher than that of "default models"), the one-tailed probability of 0.01 is more appropriate. Some readers would perhaps prefer only to rank those instances where the optimised model was substantially higher. If we take "substantially higher" to mean "2 standard errors or more", then the optimised model is substantially higher than the default model in 6 out of the 8 cases (the two mutagenesis data sets are eliminated). The corresponding Wilcoxon probabilities are now 0.05 (two-tailed) and 0.025 (one-tailed). The statistical evidence in favour of the optimised models therefore appears to be significant, perhaps even highly so.

Expt.	Coded Values		Natural Values		*Acc*
	Minacc	*Minpos*	*Minacc*	*Minpos*	
E9	−1.41	−1.41	0.72	4	0.793
E10	−1.41	−1	0.72	5	0.793
E11	−1.41	0	0.72	8	0.769
E12	−1.41	+1	0.72	10	0.763
E13	−1.41	+1.41	0.72	12	0.769
E14	−1	−1.41	0.75	4	0.793
E15	−1	−1	0.75	5	0.793
E16	−1	0	0.75	8	0.763
E17	−1	+1	0.75	10	0.763
E18	−1	+1.41	0.75	12	0.769
E19	0	−1.41	0.82	4	0.799
E20	0	−1	0.82	5	0.793
E21	0	0	0.82	8	0.769
E22	0	+1	0.82	10	0.775
E23	0	+1.41	0.82	12	0.787
E24	+1	−1.41	0.90	4	0.746
E25	+1	−1	0.90	5	0.669
E26	+1	0	0.90	8	0.645
E27	+1	+1	0.90	10	0.645
E28	+1	+1.41	0.90	12	0.662
E29	+1.41	−1.41	0.93	4	0.698
E30	+1.41	−1	0.93	5	0.639
E31	+1.41	0	0.93	8	0.592
E32	+1.41	+1	0.93	10	0.598
E33	+1.41	+1.41	0.93	12	0.598

(a) Mutagenesis (B_{min})

Expt.	Coded Values		Natural Values		*Acc*
	Minacc	*Minpos*	*Minacc*	*Minpos*	
E9	−1.41	−1.41	0.72	4	0.883
E10	−1.41	−1	0.72	5	0.911
E11	−1.41	0	0.72	8	0.905
E12	−1.41	+1	0.72	10	0.899
E13	−1.41	+1.41	0.72	12	0.864
E14	−1	−1.41	0.75	4	0.899
E15	−1	−1	0.75	5	0.911
E16	−1	0	0.75	8	0.905
E17	−1	+1	0.75	10	0.899
E18	−1	+1.41	0.75	12	0.864
E19	0	−1.41	0.82	4	0.911
E20	0	−1	0.82	5	0.911
E21	0	0	0.82	8	0.899
E22	0	+1	0.82	10	0.899
E23	0	+1.41	0.82	12	0.864
E24	+1	−1.41	0.90	4	0.905
E25	+1	−1	0.90	5	0.899
E26	+1	0	0.90	8	0.882
E27	+1	+1	0.90	10	0.870
E28	+1	+1.41	0.90	12	0.858
E29	+1.41	−1.41	0.93	4	0.899
E30	+1.41	−1	0.93	5	0.888
E31	+1.41	0	0.93	8	0.864
E32	+1.41	+1	0.93	10	0.858
E33	+1.41	+1.41	0.93	12	0.858

(b) Mutagenesis (B_{max})

Expt.	Coded Value	Natural Value	*Acc*
	Minacc	*Minacc*	
E9	−1.41	0.72	0.586
E10	−1	0.75	0.595
E11	0	0.83	0.553
E12	+1	0.90	0.516
E13	+1.41	0.93	0.526

(c) Carcinogenesis (B_{max})

Figure 11: Optimisation by using a multi-level full factorial design (procedure *OptimiseFact* in Section 4.3). In each case, relevant factors are those obtained by screening (Figure 9). A 5-level full factorial design is then used to find the best values for these factors, using experimental performance values for the response variable.

Procedure	(Accuracy,Expts.)			
	Mutagenesis		Carcinogenesis	
	B_{min}	B_{max}	B_{min}	B_{max}
ILP+Default	$(0.755 \pm 0.031, 0)$	$(0.846 \pm 0.026, 0)$	$(0.510 \pm 0.028, 0)$	$(0.504 \pm 0.028, 0)$
ILP+SO	$(0.803 \pm 0.029, 13)$	$(0.883 \pm 0.023, 14)$	$(0.510 \pm 0.028, 8)$	$(0.591 \pm 0.027, 19)$
ILP+SO′	$(0.787 \pm 0.030, 33)$	$(0.883 \pm 0.023, 33)$	$(0.510 \pm 0.028, 8)$	$(0.579 \pm 0.027, 13)$

Figure 12: Comparison of procedures, based on their final performance, using the parameter values obtained from optimising experimental performance. The entries shown are 10-fold cross-validation estimates and the number of experiments needed to obtain the optimised value. There is no unbiased estimator of variance for the cross-validation estimates (Bengio and Grandvalet, 2004): the standard error reported is computed using the approximation in Breiman et al. (1984).

2. The screening results suggest that as inputs change, so can the relevance of factors (for example, when the background changes from B_{min} to B_{max} in Carcinogenesis, *Minacc* becomes a a relevant factor). Further evidence for this comes from the "DSSTox" data set (see Figure 15). This means that a once-off choice of relevant factors across all possible inputs can lead to sub-optimal performances from the system for some inputs.

3. Screening, as proposed here, still requires identification of an initial set of variables as factors to be varied (here, these were C, *Nodes*, *Minacc* and *Minpos*). While the set can have any number of elements (all quantitative of course, for the techniques here to be applicable), the choice of these elements remains in the hands of the practitioner using the ILP system. Some element of human expertise of this kind appears unavoidable (and indeed, is even desirable, to prevent pointless experimentation). Additional assistance in the form of including, with each ILP system, a set of potentially sensitive parameters, could be a great help.

4. Optimisation, as proposed here, requires the selection of an appropriate step-size and specification of a stopping criterion for a sequential search conducted along the gradient to the response surface. We have followed the prevalent practice in the field, namely, obtaining the step-size by a process of a binary search over the interval $[0,1]$; and using a "k-in-a-row" stopping rule (that is, stopping the search if k steps yield no improvement in response). Other techniques exist, and are described in Appendix B.

5. Even if a set of relevant factors are available for a given input, a multi-level full factorial design can be an expensive method to determine appropriate levels. Once done, performance may still be sub-optimal. The results here suggests that experimental studies that *ad hoc* discretisation followed by exhaustive combinations of the different discrete levels of relevant parameters may not yield the best results.

Finally, a controlled comparison of **Default**, **SO** and **SO′** has required us to enforce that the ILP system used is the same in all experiments. In practice, we are often interested in controlled comparisons of a different kind, namely, the performances of different ILP systems. The results here suggest equipping each ILP system with the procedure **SO** could enable a controlled comparison of best-case performances: a practice which has hitherto not been adopted by empirical

Data	**ILP+Default**	**ILP+SO**
Mut(42)	0.857 ± 0.054	0.857 ± 0.054
Alz (Amine)	0.714 ± 0.017	0.802 ± 0.015
Alz (Tox)	0.792 ± 0.014	0.872 ± 0.011
Alz (Acetyl)	0.527 ± 0.014	0.774 ± 0.011
Alz (Memory)	0.551 ± 0.020	0.674 ± 0.019
DSSTox	0.647 ± 0.020	0.731 ± 0.018

Figure 13: Estimated accuracies for the Aleph system from some additional data sets used in the literature (Muggleton et al., 2008; Landwehr et al., 2006). The data sets are used in comparative experiments ("System X versus Aleph") that use default settings for all parameters of Aleph. Accuracy estimates for such models are in the column headed "ILP+Default" (although these exact values do not concern us here, we note that differences, if any, to accuracies reported in the literature can be attributed to differences in the cross-validation splits used). The column headed "ILP+SO" are final performance estimates obtained using Aleph with the **SO** procedure described in the paper, and the method used in the preliminary experiments. Standard errors are calculated as before. The DSSTox background information differ slightly in Muggleton et al. (2008) and Landwehr et al. (2006) and the models here use the variant from Muggleton et al. (2008).

Data	**ILP+Default**	**ILP+SO**	Δ	Signed Rank
Carcin	0.504	0.591	0.089	+4
Mut (188)	0.846	0.883	0.037	+1
Mut(42)	0.857	0.857	0	–
Alz (Amine)	0.714	0.802	0.088	+5
Alz (Tox)	0.792	0.872	0.080	+2
Alz (Acetyl)	0.527	0.774	0.247	+7
Alz (Memory)	0.551	0.674	0.123	+6
DSSTox	0.647	0.731	0.084	+3

Figure 14: Absolute differences in accuracy Δ between the procedures **ILP+SO** and **ILP+Default**, and their signed ranks (eliminating ties). The Wilcoxon probability of obtserving the signed ranks under the null hypothesis that median differences are 0, is 0.02 (0.01 for a directional test).

Data	**ILP+Default**	**ILP + SO**
DSSTox (Muggleton et al., 2008)	0.647 ± 0.020	0.731 ± 0.018
DSSTox (Landwehr et al., 2006)	0.631 ± 0.020	0.631 ± 0.020

Figure 15: Estimated accuracies for the Aleph system for two variants of the "DSSTox" problem. The data sets in the two variants use slightly different background information, resulting in different accuracies for both default and optimised models. Screening results are also different in the two cases: *Minacc* and *C* are relevant in DSSTox (Muggleton et al., 2008); but none of the parameters are relevant in DSSTox (Landwehr et al., 2006).

Data	**Toplog+Default**	**Toplog+SO**	Δ	Signed Rank
Carcin	0.641	0.623	0.018	−2
Mut (188)	0.840	0.867	0.027	+3.5
Mut(42)	0.881	0.881	0	–
Alz (Amine)	0.704	0.704	0	–
Alz (Tox)	0.672	0.699	0.027	+3.5
Alz (Acetyl)	0.640	0.635	0.005	−1
Alz (Memory)	0.526	0.653	0.127	+5
DSSTox	0.618	0.618	0	–

Figure 16: Absolute differences in accuracy Δ between the procedures **Toplog+SO** and **Toplog+Default**, and their signed ranks (eliminating ties). Once again, we differences, if any, to accuracies reported in the literature can be attributed to differences in the cross-validation splits used. Although the sum of the signed ranks (+9) is in favour of Toplog+**SO**, the evidence is not statistically significant (that is $p > 0.05$)

ILP studies, but whose value is self-evident. Of course, screening and optimisation experiments would have to be conducted for each system in turn, since the factors relevant to one system (and its levels) would typically have no relation to those of any of the others. We illustrate this in Figures 16–17. The former shows results of applying the procedure **SO** to a recently proposed ILP system (Toplog) on the data sets we have considered thus far. Parameter screening and optimisation proceeds for a different set of parameters to those used for Aleph: we have used the parameters *Max_literals_in_hypothesis* (equivalent to the parameter *C* in the Aleph experiments), *Max_singletons_in_hypothesis*, *Example_inflation*, and *Minpos* (which has the same meaning as *Minpos* in the Aleph experiments). The choice of these parameters was based on their use in data files provided with the Toplog program. It is evident from Figure 16 that there is an improvement in performance after using **SO** (the overall sum of signed ranks is in favour of Toplog+**SO**) although the differences are not statistically significant. This statistical caveat notwithstanding, Figure 17 shows the perils of not comparing like-with-like. Figure 17(a) shows that having subject both Toplog and Aleph to the same procedure for screening and optimisation (that is, **SO**), we find no significant difference in their performance. On the other hand, Figure 17(b) shows that performing screening and optimisation on one (Aleph), but not the other (Toplog), can lead to misleading results (that the performance of Aleph is significantly better than Toplog).

6. Concluding Remarks

As an ILP system moves from being a prototype for demonstrating a proof-of-concept to being a tool for regular data analysis, it moves into the province of engineering. The requirements of a system in this latter world are significantly more stringent than in the former: robustness is needed, of course, as are mechanisms that facilitate ease of use, recovery from failures, and so on. It also becomes no longer adequate simply to demonstrate that a model *can* be constructed in some novel manner, requiring instead that the model constructed is as good as possible for a given set of inputs (by this we mean primarily the background knowledge and examples). Besides the obvious benefit

Data	**Toplog+SO**	**Aleph+SO**	Δ	Signed Rank
Carcin	0.623	0.591	0.032	−4
Mut (188)	0.867	0.883	0.016	+1
Mut(42)	0.881	0.857	0.024	−3
Alz (Amine)	0.704	0.802	0.070	+5
Alz (Tox)	0.699	0.872	0.173	+8
Alz (Acetyl)	0.635	0.774	0.139	+7
Alz (Memory)	0.653	0.674	0.021	+2
DSSTox	0.618	0.731	0.113	+6

(a)

Data	**Toplog+Default**	**Aleph+SO**	Δ	Signed Rank
Carcin	0.641	0.591	0.050	−3
Mut (188)	0.840	0.883	0.043	+2
Mut(42)	0.881	0.857	0.024	−1
Alz (Amine)	0.704	0.802	0.098	+4
Alz (Tox)	0.672	0.872	0.200	+8
Alz (Acetyl)	0.640	0.774	0.134	+6
Alz (Memory)	0.526	0.674	0.148	+7
DSSTox	0.618	0.731	0.113	+5

(b)

Figure 17: (a) Absolute differences in accuracy Δ between the procedures **Aleph+SO** and **Toplog+SO**, and their signed ranks (eliminating ties). Although the sum of the signed ranks is in favour of Aleph+**SO** (+22), the evidence is not statistically significant (that is $p > 0.05$). (b) Absolute differences in accuracy Δ between the procedures **Aleph+SO** and **Toplog+Default**, and their signed ranks (eliminating ties). The sum of the signed ranks is in favour of Aleph+**SO** (+28), is now statistically significant ($p = 0.05$ for a non-directional test, $p = 0.025$ for a directional test). Performing the comparison (b) instead of (a) can result in the misleading conclusion that the Aleph system performs significantly better than Toplog on these data sets.

to the modelling problem being addressed, it ensures that the performance of ILP systems can be assessed in a meaningful manner. Here, we have taken a system engineer's approach to this problem by identifying a set of critical parameters of the system, and then varying these to improve performance. The principal tools we have used are those developed under the umbrella of design and analysis of experiments. Our principal contribution here is to show how these tools can be used

to develop better models with ILP systems. To the best of our knowledge, this is the first time[9] any such formal framework has been employed for this purpose in ILP.

There are a number of ways in which the work here can be extended further. On the conceptual front, we have concentrated on the simplest forms of designed experiments (sometimes called "classical" DOE). Substantial effort has been expended in developing designs other than the fractional factorial designs used here. Response surface optimisation could also involve more complex models than the simple first-order models used here. Both options could yield better results than those obtained here. On the experimental front, our emphasis has been on a controlled study of fractional-factorial screening and response-surface optimisation, using well-studied ILP benchmarks. There are clearly many other data sets studied within ILP that could benefit from utilising the techniques proposed. We have also modelled system performance by its estimated accuracy: clearly other measures may be of interest (for example, some combination of the accuracy and complexity of models, in the MDL sense). Finally, it is evident from our results in Figure 17 that there are wider implications of the results here to the work on the comparative study of ILP systems, and to the development of ILP systems as tools for data analysis. Indeed, nothing restricts the procedures here just to ILP, and the same comments apply to many other machine learning systems. Although outside the scope of this paper, these directions are clearly of some importance, and worth pursuing.

Acknowledgments

The authors would like to thank Ravi Kothari of IBM Research—India, for initiating interest in the area of design and analysis of experiments. For much of this work, A.S. was supported by a Ramanujan Fellowship, Dept. of Science and Technology, Government of India; and was at the Dept. of Biotechnology's Centre of Excellence at the School of Information Technology, Jawaharlal Nehru University, New Delhi.

Appendix A. A Note on Linear Regression Models

In this section we provide details of regression models that are of relevance to this paper. All these details can be obtained in any textbook on statistical modelling: we reproduce them here simply for completeness.

Given a response variable y and variables $x_1, x_2, \ldots, x_k$, a regression model expresses a relationship between y and the x_i as follows:

$$y = f(x_1, x_2, \ldots, x_k) + \varepsilon$$

where f denotes a systematic functional relationship between y and the x_i, and ε denotes random variation in y that is unrelated to the x_i (usually called the *error*). Usually f is specified as some mathematical function (for example, a polynomial in the x_i) and ε by a probability density function (PDF). The PDF for ε is taken to have mean 0 and standard deviation σ: normally the distribution is also taken to be Gaussian. Thus, in a slightly lop-sided way, for a given set of values for the x_i, it is easier to think of a random value being chosen for ε and then constant $f(x_1, \ldots, x_k)$ being

9. At the time of going to press, we have become aware of a recent paper by Janssen and Fürnkranz (2010) with a motivation very similar to this paper (although not applied to ILP). The connection between the work reported there and that in this paper, especially in the context of ILP, would be worth investigating further.

added to give the final value of y. From this is evident that y will have a PDF with mean given by $E(y) = E(f(x_1,\ldots,x_k)+\varepsilon) = f(x_1,\ldots,x_k)+E(\varepsilon) = f(x_1,\ldots,x_k))$; and standard deviation σ. Thus, the regression function effectively specifies the expected, or mean value, of y, given the x_i. "Linear regression" refers to the case when the functional relationship is a linear equation of the form:

$$f(x_1,\ldots,x_k) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k.$$

Here, "linear" refers to being linear in the coefficients β_i. So, the following is also a case of linear regression:

$$f(x_1,\ldots,x_k) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \beta_{k+1} x_1^2 + \cdots + \beta_{2k} x_k^2 + \beta_{2k+1} x_1 x_2 + \cdots.$$

To differentiate between these kinds of equation, we denote the former kind which only contain terms $x_1, x_2, \ldots$ as first-order function; and equations of the latter kind which contain quadratic and interaction terms as a second-order function.

In general, assuming we knew the form of f (for example, that it was a first-order function, with errors following a Gaussian distribution with zero mean and variance σ^2), and which of the x_i were functionally related to y, we still need to be able to obtain values of the β_i from a set of observations, or data points, giving values for the relevant x_i and the corresponding values of y. Actually, the best we are able to do is obtain estimates of β_i, which we will denote here as b_i, along with some statistical statement on these estimates. The result is a regression model:

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \cdots.$$

Thus, with each data point k, we have an associated "residual" given by difference between the value y_k for that data point, and the value $\hat{y}_k$ obtained from the regression model. The usual approach for obtaining the estimates b_i is the method of least squares, that attempts to minimise the sum of squares of the residuals. The details can be found in any standard statistical textbook (for example, Walpole and Myers, 1978).

We now turn to the first of our assumptions, namely, that of the form of the function. The validity of this assumption can be tested by examining how well the model fits the observed data; and, if used for prediction, estimating how well it will predict response values on new data. The degree of model fit is obtained by examining the residuals and calculating first the statistical significance of model. This tests the null hypothesis $H_0 : b_0 = b_1 = \cdots = b_k = 0$ (that is, there is no linear relationship between y and any of the x_i). Specifically, the quantity:

$$F = \frac{SSR/k}{SSE/(N-1-k)}$$

is calculated, where where *SSE* refers to the sum of squared residuals $(\sum_{k=1}^{N}(y_k - \hat{y}_k))^2$, N being the number of data points); and *SSR* is the sum of squares of deviations of the model's response from the mean response $(\sum_{k=1}^{N}(\hat{y} - \overline{y}))^2$). F is known to follow the F-distribution with $k, N-1-k$ degrees of freedom (Walpole and Myers, 1978). So, the hypothesis H_0 can be rejected at some level of significance α, if the F-value obtained is greater than the value tabulated for $F_{\alpha,k,N-1-k}$.

Assuming the null hypothesis is rejected, a quantity that is often used to quantify the degree of fit is the the *coefficient of determination*:

$$R^2 = 1 - \frac{SSE}{SST}$$

where SST is similar to SSR, being the sum of squares of deviations of the observed response from the mean response ($\sum_{k=1}^{N} (y_k - \bar{y}))^2$). A little arithmetical manipulation will show $SSR + SSE = SST$, and therefore:

$$R^2 = \frac{SSR}{SST}.$$

Thus, R^2 is the proportion of the variation in y "explained" by the model. Clearly, $SSR \leq SST$ and therefore $0 \leq R^2 \leq 1$. In general, adding more terms to the regression model will only increase R^2 as the model tends to overfit the data. A quantity that takes overfitting into account is the "adjusted" coefficient of determination:

$$R^2_{adj} = 1 - \frac{N-1}{N-k-1}(1-R^2).$$

If there is a substantial difference between R^2 and R^2_{adj}, then the model is taken to be overfitting the data.

While R^2 or R^2_{adj} denote how well the regression model fits the observed data, it does not have anything to say on the model's performance on new data. An estimate of the predictive power of the model is obtained by performing a resampling exercise by leaving out each of the N data points, and obtaining the corresponding residual based on the model constructed with the remaining $N-1$ points. This is used to calculate a coefficient of determination for prediction R^2_{pred}. Since we will not be using regression models for prediction in this paper, we will not pursue this further here.

Assumptions about the form of the regression model tacitly include assumptions about the errors, namely that they are independent, identically distributed Gaussian variables with zero mean and variance σ^2. The validity of these assumptions are normally checked by visual tests. Graphs of the residual against the predicted response should show no specific pattern; and normal quantile-quantile plots of the residuals should be a straight line (Jain, 1991).

We turn now to the second major assumption, namely that the factors of relevance are known before obtaining the model. This requirement can now be relaxed, since we are able to also test the hypothesis that each of the coefficients b_i are individually equal to zero (the earlier test of significance simply tested that *all* of the b_i were zero: rejection of that hypothesis could still mean *some* of the b_i were zero). This test allows us to eliminate as irrelevant all those factors whose coefficients are not significantly different from zero. In fact, the test forms the basis for a "greedy" procedure that examines the stepwise addition and removal of factors. We reproduce the implementation described in Srinivasan (2001b) of this procedure in Figure 18. It is normal to start procedure with $I = \emptyset$. Although it is not guaranteed to find the most relevant subset of factors, and in the worst case, the number of subsets examined can be exponential in $|V|$ the method has been found to work well in practice. Restricted variants of the method are also popular: *forward selection* starts with $I = \emptyset$ and dispenses with the exclusion steps (Steps 6–7 in Figure 18); *backward elimination* starts with $I = V$ and dispenses with the inclusion steps (Steps 4–5 in in Figure 18). Both variants examine no more than $O(|V|^2)$ subsets.

$stepr(V, I, F_{in}, F_{out})$: Given a set of potential regressor variables V (factors in this paper); an initial subset of variables $I \subseteq V$; and minimum values of the F statistic that a variable must achieve to enter (F_{in}) or remain (F_{out}) in the regression equation, returns a subset $S \subseteq V$ identified by a stepwise variable selection procedure.

1. $i = 0$
2. $S_i = I, V_i = V \setminus I$
3. Increment i
4. Let v_{in} be the single best variable in V_{i-1} that can be included (that is, on inclusion, gives the greatest increase in the coefficient of determination)
5. If $f(v_{in}|S_{i-1}) \geq F_{in}$ then $S = S_{i-1} \cup \{v_{in}\}$; otherwise $S = S_{i-1}$
6. Let v_{out} be the single best variable in S that can be excluded (that is, on exclusion, gives the greatest increase in the coefficient of determination)
7. If $f(v_{out}|S \setminus \{v_{out}\}) \leq F_{out}$ then $S_i = S \setminus \{v_{out}\}$; otherwise $S_i = S$
8. If $S_i = S_{i-1}$ then return S_i; otherwise continue
9. $V_i = V \setminus S_i$
10. Go to Step 3

Figure 18: A stepwise variable selection procedure for multiple linear regression (reproduced from Srinivasan, 2001b). The coefficient of determination (often denoted by R^2) denotes the proportion of total variation in the dependent variable that is explained by the fitted model. Given a model formed with the set of variables X, it is possible to compute the observed change in R^2 due to the addition of some variable v. The probability that the true value of this change is 0 can be obtained from a use of the F statistic (Walpole and Myers, 1978). The function $f(v|X)$ returns the value of the F distribution under the null hypothesis that there is no change in R^2 by adding variable v to those in X. The thresholds F_{in} and F_{out} thus specify acceptable probability levels for the inclusion (and exclusion) of variables. It is evident that $F_{in} > F_{out}$ in order to avoid the same variable from repeatedly being included and excluded. A correct implementation of $svs(\ldots)$ also requires sample data and the appropriate regression function to be provided as parameters. We have ignored these here for simplicity.

Appendix B. A Note on Constructing and Optimising Response Surfaces

In this section we describe some issues that are relevant to constructing and optimising response surfaces. Specifically, we are concerned with: (1) A procedure for obtaining a fractional experimental design that is suitable for estimating the main effects using the regression procedure described just previously; (2) The search procedure along the gradient to the response surface.

B.1 Fractional Factorial Designs

We begin by assuming that we have k main effects and that the response surface is approximated by a first-order model with main effects only. That is, we are required to estimate $k+1$ coefficients in a linear model. This requires at least $k+1$ data points, and we simply reproduce a recipe described in Jain (1991) that produces a suitable two-level fractional factorial design:

1. Two-level fractional designs are obtained by dividing the full factorial design of k factors by some number 2^p ($1 \leq p < k$). It is common to refer to such a design as a 2^{k-p} design. Thus, we want to reduce the number of experiments from 2^k to some number 2^{k-p} such that $2^{k-p} \geq (k+1)$. That is, $p = \lfloor k - log(k+1) \rfloor$. Select any $k-p$ factors and construct a two-level full factorial design with these factors. Clearly, this will contain $k-p$ columns (one for each factor). Next, extend this table with columns containing all products of factors. Thus, suppose we initially had $k = 4$ factors (A, B, C, D say), and wanted to construct a 2^{4-1} factorial design (that is $p = 1$). We commence by selecting $k - p = 3$ factors (A, B, C) say, and first construct the following table (this example is from Jain, 1991):

Expt.	A	B	C	AB	AC	BC	ABC
E1	-1	-1	-1	$+1$	$+1$	$+1$	-1
E2	-1	-1	$+1$	$+1$	-1	-1	$+1$
E3	-1	$+1$	-1	-1	$+1$	-1	$+1$
E4	-1	$+1$	$+1$	-1	-1	$+1$	-1
E5	$+1$	-1	-1	-1	-1	$+1$	$+1$
E6	$+1$	-1	$+1$	-1	$+1$	-1	-1
E7	$+1$	$+1$	-1	$+1$	-1	-1	-1
E8	$+1$	$+1$	$+1$	$+1$	$+1$	$+1$	$+1$

   It should be evident that the resulting table will contain $2^{k-p} - 1$ columns.

2. From the $2^{k-p} - 1 - (k-p)$ "product" columns on the right of this table, select p columns and rename them with the p factors not selected in the step above. For example, if we select the ABC column and replace it with D:

Expt.	A	B	C	AB	AC	BC	D
E1	-1	-1	-1	$+1$	$+1$	$+1$	-1
E2	-1	-1	$+1$	$+1$	-1	-1	$+1$
E3	-1	$+1$	-1	-1	$+1$	-1	$+1$
E4	-1	$+1$	$+1$	-1	-1	$+1$	-1
E5	$+1$	-1	-1	-1	-1	$+1$	$+1$
E6	$+1$	-1	$+1$	-1	$+1$	-1	-1
E7	$+1$	$+1$	-1	$+1$	-1	-1	-1
E8	$+1$	$+1$	$+1$	$+1$	$+1$	$+1$	$+1$

This design will allow us to estimate the main effects A,B,C,D, as well as the interactions AB,AC and BC. However (by construction) it will be impossible to distinguish between the effect of D and that of ABC: the two effects are said to be *confounded* and the terms said to be *aliased*. These are not the only effects that are confounded, and it can be verified that each main effect is confounded with a three-way interaction ($A = BCD$ and so on), and that each two-way interaction is confounded with other two-way interactions ($AC = BD$ and so on). If we are only interested in estimating main effects, then, provided we can assume that three-way interaction effects are negligible, then a table containing just the four A,B,C,D columns above would be adequate. That is, the fractional design is:

Expt.	A	B	C	D
E1	-1	-1	-1	-1
E2	-1	-1	$+1$	$+1$
E3	-1	$+1$	-1	$+1$
E4	-1	$+1$	$+1$	-1
E5	$+1$	-1	-1	$+1$
E6	$+1$	-1	$+1$	-1
E7	$+1$	$+1$	-1	-1
E8	$+1$	$+1$	$+1$	$+1$

The reader will recognise this as the design used to estimate main effects in the paper. It is clear that the choice of replacing the ABC column with D was an arbitrary one (as indeed, was the choice of A,B,C in the first place): we could, for example, have elected to replace the AB column with D. Thus, there are several 2^{4-1} fractional factorial designs that could have been devised. The difference lies in the assumptions that need to made when estimating

main effects: in general, it is considered better to confound main effects with higher order interactions, as these are assumed to be smaller. That is, a design that confounds D with AB will probably yield poorer estimates of the effect of D than one that confounds D with ABC.

Some additional points are in order:

1. The column vectors in the two-level full and fractional factorial designs satisfy some properties: (a) The sum of each column is zero; (b) The sum of products of each column is zero; and (c) The sum of squares of each column is equal to the number of experiments. These properties result in some advantages in computing the main effects: see Jain (1991).

2. In a fractional design some factor combination, usually called *identity* and denoted by I, contains 1 in all rows. Such a combination is called the *generator* for the design. For example, $I = ABCD$ is the generator for the design above.

3. Two-level fractional factorial designs are categorised by their *resolution*. The resolution R of a fractional factorial design can be computed as the smallest number of factors that are confounded with the generator I. In the 2^{4-1} design above terms with I is confounded with just one factor combination ($ABCD$). Thus the resolution of the design is 4. Resolutions are normally denoted by Roman numeral subscripts. Thus, the fractional design in Figure 5 is a 2^{4-1}_{IV} design (Montgomery, 2005). In Resolution II designs, main effects are aliased with other main effects. In Resolution III designs, main effects are aliased with with two-factor interactions, and two-factor interactions may be aliased with each other. In Resolution IV designs, main effects are not aliased with each other or with two-factor interactions, but two-factor interactions may be aliased with each other. In Resolution V designs, the only aliasing that occurs is between two- and three-factor interactions, and so on.

4. Two desirable properties relating resolution and linear models with two-level factors (± 1) are those of orthogonality and rotatability. Orthogonal designs result in minimal variance when estimating coefficients, and both full factorial designs and fractional designs in which main effects are not aliased with each other (that is, Resolution III or more) are known to be orthogonal for first-order models (Montgomery, 2005). Rotatability concerns variance in prediction across the factor space. Designs that yield predictions whose variance changes symmetrically from the centre of the factor space are said to be rotatable. That is, the variance of prediction at points equidistant from the centre of the factor space should be the same. Once again, full factorial designs and fractional designs of Resolution III or more are rotatable designs for first-order models. Rotatable designs for models with higher order terms ($x_1^2, x_2^2, \ldots$) will require additional experiments (we will describe these in the following section).

5. In general, if there is a variation in response y even for fixed values of the factors, then we will need to perform several replicates of each experiment, and attempt to model the average response $\bar{y}$. Also, to ensure that there is no dependency in the response variable across experiments, we may need to run the experiments in a randomised order. We will ignore this aspect here, and assume a single replicate for each experiment. One consequence of the latter assumption is that factor levels need to be spread out widely (that is, in two-level experiments, the difference between values corresponding to -1 and $+1$ should be as large as possible), so that effect estimates are reliable (see Montgomery, 2005).

It is evident from from these points that increasing the resolution will allow the construction of models that contain more terms from the full factorial model. Thus, with Resolution III and IV designs, it will only be possible to obtain models that contain the main effects (first-order models). With a Resolution V model, a model with both main effects and two-way interactions can be obtained. Rotatable designs also provide some theoretical guarantees on the estimates, both of coefficients and the response, on these models.

B.2 Gradient Ascent

The primary device used in the paper is to seek local improvements in the response y by making small movements in the direction of the gradient to a response surface. The rationale for gradient ascent can be found in any text on optimization: we present a version here (from Bronson and Naadimuthu, 1982) for completeness. Let us suppose that the response surface is given by a scalar field f defined on points that are some subset of $\Re^k$, and whose values $f(x_1, x_2, \ldots, x_k)$ we denote using a vector notation as $f(\mathbf{X})$. We wish to determine a point $\mathbf{x}*$ for which $f(\mathbf{x}*)$ is a (local) maximum.

From the vector calculus, it is known that for any fixed point $\mathbf{x}$ and a unit vector $\mathbf{U}$, the rate of change of $f(\mathbf{X})$ at $\mathbf{x}$ in the direction of $\mathbf{U}$ is given by $\nabla f|_{\mathbf{X}=\mathbf{x}} \cdot \mathbf{U}$, where ∇f is a k-dimensional vector of partial derivatives given by $\left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_k}\right)$ and $\cdot$ denotes the inner, or scalar product of a pair of vectors. For vectors $\mathbf{a}$ and $\mathbf{b}$ the inner product $\mathbf{a} \cdot \mathbf{b}$ is given by $|\mathbf{a}||\mathbf{b}|cos\theta$, where θ is the angle between the vectors $\mathbf{a}$ and $\mathbf{b}$. With some slight abuse of notation, the rate of change of $f(\mathbf{X})$ at $\mathbf{x}$ in the direction of $\mathbf{U}$ is:

$$\nabla f|_{\mathbf{X}=\mathbf{x}} \cdot \mathbf{U} = |\nabla f||\mathbf{U}|cos\theta = |\nabla f|cos\theta.$$

The rate of change is therefore greatest when $cos\theta = 1$, or $\theta = 0$. That is, $\mathbf{U}$ is in the same direction of ∇f. Thus, of all non-unit vector displacements of size δ from the point $\mathbf{x}$, the rate of change of $f(\mathbf{x})$ will be greatest for the vector $\delta\nabla f|_{\mathbf{x}}$ (since this vector is clearly along the direction of ∇f). Further, the best value of δ will be the one that maximises $f(\mathbf{x} + \delta\nabla f|_{\mathbf{x}})$.

B.2.1 SEARCH ALONG THE GRADIENT

In order to use the differential calculus to obtain a value of δ that maximises $f(\mathbf{x} + \delta\nabla f|_{\mathbf{x}})$ in any interval, the function has to be known analytically and the resulting equation for stationary points $f'(\mathbf{x} + \delta\nabla f|_{\mathbf{x}}) = 0$ should be solvable algebraicly. In our case, we do not know the functional form of f: the first-order response surface is simply a local approximation to f that ceases to be appropriate after some value of δ. We therefore have to adopt some form of search for an appropriate value of δ. The simplest of these—and widely used in response surface methods (Neddermeijer et al., 2000)—is the enumerative search we have used in the paper, along with a "k-in-a-row" stopping rule (that is, the search terminates when k steps yield no improvement). Improved versions have been suggested in the literature. The enumerative search could be improved by using better sequential search techniques (for example, a three-point interval search, or a Fibonacci search). In fact, this search itself can be posed as an optimisation problem. In Fu (1994) data from experiments performed along the gradient are used to construct a higher order polynomial function of response values in terms of δ. For example, with 3 data points along ∇f obtained from step sizes of $\delta = \delta_1, \delta_2, \delta_3$, and corresponding response values $y = y_1, y_2, y_3$ it will be possible to obtain least-squares estimates for the

Expt.	δ	*Acc*
E9	0.0	0.769
E10	−0.5	0.793
E12	−1.5	0.781
E13	−2.0	0.692

$Acc = 0.763 - 0.117\,\delta - 0.075\,\delta^2$

$\delta^* = -0.78$

Figure 19: Data from steps of the gradient ascent used to estimate a polynomial regression model relating response (*Acc*) to step-size (δ). The data shown here are from Figure 10(a). The "optimal" value δ^* is obtained using standard techniques from the differential calculus applied to this model.

α_i in $y = \alpha_0 + \alpha_1\delta + \alpha_2\delta^2$. The optimal value for δ can then be easily estimated from this function, as $\delta^* = \frac{-a_1}{2a_2}$ (where a_1 and a_2 are the least-squares estimates of α_1 and α_2). We illustrate this in Figure 19 below, that uses data points from the gradient ascent steps in Figure 10. The procedure, although not perfect, is reasonably good: the step size estimate (−0.78) results in an actual response value of 0.787 (the regression model predicts 0.809).

Other techniques have been proposed as improvements on gradient search, which we do not elaborate further here. We refer the reader to Safizadeh and Signorile (1994) for descriptions and pointers to these.

References

J. Ashby and R.W. Tennant. Definitive relationships among chemical structure, carcinogenicity and mutagenicity for 301 chemicals tested by the U.S. NTP. *Mutation Research*, 257:229–306, 1991.

M. Baz, B. Hunsaker, J.P. Brooks, and A. Gosavi. Automatic tuning of optimization software parameters. Technical Report 2007-7, University of Pittsburgh, Dept. of Industrial Engineering, 2007.

Y. Bengio. Gradient based optimisation of hyperparameters. *Neural Computation*, 12(8):1889–1900, 2000.

Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research*, 5:1089–1105, 2004.

G.E.P. Box and K.B. Wilson. On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society, Series B (Methodological)*, 13(1):1–45, 1951.

I. Bratko and S.H. Muggleton. Applications of inductive logic programming. *Communications of the ACM*, 38(11):65–70, 1995.

L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984.

R. Bronson and G. Naadimuthu, editors. *Schaum's Outline of Theory and Problems of Operations Research*. McGraw Hill, New York, 1982. (2nd Edition).

A.K. Debnath, R.L Lopez de Compadre, G. Debnath, A.J. Schusterman, and C. Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, 34(2):786 – 797, 1991.

L. DeRaedt and M. Bruynooghe. Interactive concept-learning and constructive induction by analogy. *Machine Learning*, 8(2):107–150, 1992.

S. Dzeroski. Relational data mining applications: An overview. In S. Dzeroski and N. Lavrac, editors, *Relational Data Mining*, pages 339–360. Springer, Berlin, 2001.

M.C. Fu. Optimization via simulation. *Annals of Operations Research*, 53:199–248, 1994.

L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, 2007.

D. J. Hand. *Construction and Assessment of Classification Rules*. Wiley, Chichester, 1997.

J.C. Helton, J.D. Johnson, C.J. Sallaberry, and C.B. Storlie. Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering & System Safety*, 91(10–11): 1175–1209, 2006.

R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley, New York, 1991.

Frederik Janssen and Johannes Fürnkranz. On the quest for optimal rule learning heuristics. *Mach. Learn.*, 78:343–379, March 2010. ISSN 0885-6125.

S.S. Keerthi, V. Sindhwani, and O. Chapelle. An efficient method for gradient-based adaptation of hyperparameters in svm models. In *NIPS*, pages 673–680, 2006.

R.D. King and A. Srinivasan. Prediction of rodent carcinogenicity bioassays from molecular structure using inductive logic programming. *Environmental Health Perspectives*, 104(5):1031–1040, 1996.

R.D. King, S.H. Muggleton, A. Srinivasan, and M.J.E. Sternberg. Structure-activity relationships derived by machine learning: The use of atoms and their bond connectivities to predict mutagenicity by inductive logic programming. *Proc. of the National Academy of Sciences*, 93: 438–442, 1996.

R. Kohavi and G.H. John. Automatic parameter selection by minimizing estimated error. In A. Prieditis and S. Russell, editors, *Proceedings of the Twelfth International Conference on Machine Learning*, pages 204–212, San Francisco, CA, 1995. Morgan Kaufmann.

N. Landwehr, A. Passerini, L. De Raedt, and P. Frasconi. kfoil: Learning simple relational kernels. In *AAAI*, pages 389–396, 2006.

D.C. Montgomery. *Design and Analysis of Experiments (5th Ed.)*. John Wiley, New York, 2005.

S. Muggleton. Inverse Entailment and Progol. *New Gen. Comput.*, 13:245–286, 1995.

S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19,20:629–679, 1994.

S.H. Muggleton, J.C. Almeida Santos, and A. Tamaddoni-Nezhad. Toplog: Ilp using a logic program declarative bias. In *ICLP*, pages 687–692, 2008.

H.G. Neddermeijer, G.J. van Oortmarssen, N. Piersma, and R.Dekker. A framework for response surface methodology for simulation optimization. In *Winter Simulation Conference*, pages 129–136, 2000.

M.H. Safizadeh and R. Signorile. Optimization of simulation via quasi-newton methods. *ORSA Journal on Computing*, 6(4):388–408, 1994.

S. Siegel. *Nonparametric Statistics for the Behavioural Sciences*. McGraw-Hill, New York, 1956.

A. Srinivasan. The aleph manual. 1999. URL `http://www.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/`.

A. Srinivasan. Four suggestions and a rule concerning the application of ilp. In Nada Lavrac and Saso Dzeroski, editors, *Relational Data Mining*, pages 365–374. Springer-Verlag, Berlin, 2001a. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/ilpkdd.ps.gz`.

A. Srinivasan. Extracting context-sensitive models in inductive logic programming. *Machine Learning*, 44:301–324, 2001b. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/relev.ps.gz`.

D.D. Steppan, J. Werner, and R.P. Yeater. *Essential Regression and Experimental Design for Chemists and Engineers*. 1998. URL `http://www.jowerner.homepage.t-online.de/download.htm`.

R.E. Walpole and R.H. Myers. *Probability and Statistics for Engineers and Scientists*. Collier Macmillan, New York, 1978. 2nd Edition.

F. Zelezny, A. Srinivasan, and C.D. Page. Lattice-search runtime distributions may be heavy-tailed. In *Proceedings of the Twelfth International Conference on Inductive Logic Programming (ILP2002)*, LNAI, Berlin, 2002. Springer. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/rrr.ps.gz`.

Journal of Machine Learning Research 12 (2011) 663-689 Submitted 5/10; Revised 2/11; Published 3/11

Efficient Structure Learning of Bayesian Networks using Constraints

Cassio P. de Campos CASSIOPC@ACM.ORG
Dalle Molle Institute for Artificial Intelligence
Galleria 2
Manno 6928, Switzerland

Qiang Ji JIQ@RPI.EDU
Dept. of Electrical, Computer & Systems Engineering
Rensselaer Polytechnic Institute
110 8th Street
Troy, NY 12180, USA

Editor: David Maxwell Chickering

Abstract

This paper addresses the problem of learning Bayesian network structures from data based on score functions that are decomposable. It describes properties that strongly reduce the time and memory costs of many known methods without losing global optimality guarantees. These properties are derived for different score criteria such as Minimum Description Length (or Bayesian Information Criterion), Akaike Information Criterion and Bayesian Dirichlet Criterion. Then a branch-and-bound algorithm is presented that integrates structural constraints with data in a way to guarantee global optimality. As an example, structural constraints are used to map the problem of structure learning in Dynamic Bayesian networks into a corresponding augmented Bayesian network. Finally, we show empirically the benefits of using the properties with state-of-the-art methods and with the new algorithm, which is able to handle larger data sets than before.

Keywords: Bayesian networks, structure learning, properties of decomposable scores, structural constraints, branch-and-bound technique

1. Introduction

A Bayesian network is a probabilistic graphical model that relies on a structured dependency among random variables to represent a joint probability distribution in a compact and efficient manner. It is composed by a directed acyclic graph (DAG) where nodes are associated to random variables and conditional probability distributions are defined for variables given their parents in the graph. Learning the graph (or structure) of these networks from data is one of the most challenging problems, even if data are complete. The problem is known to be NP-hard (Chickering et al., 2003), and best exact known methods take exponential time on the number of variables and are applicable to small settings (around 30 variables). Approximate procedures can handle larger networks, but usually they get stuck in local maxima. Nevertheless, the quality of the structure plays a crucial role in the accuracy of the model. If the dependency among variables is not properly learned, the estimated distribution may be far from the *correct* one.

In general terms, the problem is to find the best structure (DAG) according to some score function that depends on the data (Heckerman et al., 1995). There are methods based on other (local) statistical analysis (Spirtes et al., 1993), but they follow a completely different approach. The re-

©2011 Cassio P. de Campos and Qiang Ji.

search on this topic is active (Chickering, 2002; Teyssier and Koller, 2005; Tsamardinos et al., 2006; Silander and Myllymaki, 2006; Parviainen and Koivisto, 2009; de Campos et al., 2009; Jaakkola et al., 2010), mostly focused on complete data. In this case, best exact ideas (where it is guaranteed to find the global best scoring structure) are based on dynamic programming (Koivisto and Sood, 2004; Singh and Moore, 2005; Koivisto, 2006; Silander and Myllymaki, 2006; Parviainen and Koivisto, 2009), and they spend time and memory proportional to $n \cdot 2^n$, where n is the number of variables. Such complexity forbids the use of those methods to a couple of tens of variables, mainly because of the memory consumption (even though time complexity is also a clear issue). Ott and Miyano (2003) devise a faster algorithm when the complexity of the structure is limited (for instance the maximum number of parents per node and the degree of connectivity of a subjacent graph). Perrier et al. (2008) use structural constraints (creating an undirected super-structure from which the undirected subjacent graph of the optimal structure must be a subgraph) to reduce the search space, showing that such direction is promising when one wants to learn structures of large data sets. Kojima et al. (2010) extend the same ideas by using new search strategies that exploit clusters of variables and ancestral constraints. Most methods are based on improving the dynamic programming method to work over reduced search spaces. On a different front, Jaakkola et al. (2010) apply a linear programming relaxation to solve the problem, together with a branch-and-bound search. Branch-and-bound methods can be effective when good bounds and cuts are available. For example, this has happened with certain success in the Traveling Salesman Problem (Applegate et al., 2006). We have proposed an algorithm that also uses branch and bound, but employs a different technique to find bounds (de Campos et al., 2009). It has been showed that branch and bound methods can handle somewhat larger networks than the dynamic programming ideas. The method is described in detail in Section 5.

In the first part of this paper, we present structural constraints as a way to reduce the search space. We explore the use of constraints to devise methods to learn specialized versions of Bayesian networks (such as naive Bayes and Tree-augmented naive Bayes) and generalized versions, such as Dynamic Bayesian networks (DBNs). DBNs are used to model temporal processes. We describe a procedure to map the structural learning problem of a DBN into a corresponding augmented Bayesian network through the use of further constraints, so that the same exact algorithm we discuss for Bayesian networks can be employed for DBNs.

In the second part, we present some properties of the problem that bring a considerable improvement on many known methods. We build on our recent work (de Campos et al., 2009) on *Akaike Information Criterion* (AIC) and *Bayesian Information Criterion* (BIC), and present new results for the Bayesian Dirichlet (BD) criterion (Cooper and Herskovits, 1992) and some derivations under a few assumptions. We show that the search space of possible structures can be reduced drastically without losing the global optimality guarantee and that the memory requirements are very small in many practical cases.

As data sets with many variables cannot be efficiently handled (unless P=NP), a desired property of a learning method is to produce an *anytime* solution, that is, the procedure, if stopped at any moment, provides an approximate solution, while if kept running, its solution improves until a global optimum is found. We point out that the term *anytime* is used to mean that the difference between best current solution and upper bound for the global optimum constantly decreases throughout the algorithm's execution (even though we cannot guarantee whether the improvement happens because a better solution is found or because the upper bound is shrunk). We describe an anytime and exact algorithm using a branch-and-bound (B&B) approach with caches. Scores are

pre-computed during an initialization step to save computational time. Then we perform the search over the possible graphs iterating over arcs. Because of the B&B properties, the algorithm can be stopped with a best current solution and an upper bound for the global optimum, which gives a certificate to the answer and allows the user to stop the computation when she/he believes that the current solution is good enough. For example, such an algorithm can be integrated with a structural Expectation-Maximization (EM) method without the huge computational expenses of other exact methods by using the generalized EM (where finding an improving solution is enough), but still guaranteeing that a global optimum is found if run until the end. Due to this property, the only source of approximation would regard the EM method itself. It worth noting that using a B&B method is not new for structure learning (Suzuki, 1996). Still, that previous idea does not constitute a global exact algorithm, instead the search is conducted after a node ordering is fixed. Our method does not rely on a predefined ordering and finds a global optimum structure considering all possible orderings.

The paper is divided as follows. Section 2 describes the notation and introduces Bayesian networks and the structure learning problem based on score functions. Section 3 presents the structural constraints that are treated in this work, and shows examples on how they can be used to learn different types of networks. Section 4 presents important properties of the score functions that considerably reduce the memory and time costs of many methods. Section 5 details our branch-and-bound algorithm, while Section 6 shows experimental evaluations of the properties, the constraints and the exact method. Finally, Section 7 concludes the paper.

2. Bayesian Networks

A Bayesian network represents a joint probability distribution over a collection of random variables, which we assume to be categorical. It can be defined as a triple $(\mathcal{G},\mathcal{X},\mathcal{P})$, where $\mathcal{G} \doteq (V_\mathcal{G},E_\mathcal{G})$ is a directed acyclic graph (DAG) with $V_\mathcal{G}$ a collection of n nodes associated to random variables $\mathcal{X}$ (a node per variable), and $E_\mathcal{G}$ a collection of arcs; $\mathcal{P}$ is a collection of conditional mass functions $p(X_i|\Pi_i)$ (one for each instantiation of Π_i), where Π_i denotes the parents of X_i in the graph (Π_i may be empty), respecting the relations of $E_\mathcal{G}$. In a Bayesian network every variable is conditionally independent of its non-descendants given its parents (Markov condition).

We use uppercase letters such as X_i, X_j to represent variables (or nodes of the graph, which are used interchanged), and x_i to represent a generic state of X_i, which has state space $\Omega_{X_i} \doteq \{x_{i1}, x_{i2}, \ldots, x_{ir_i}\}$, where $r_i \doteq |\Omega_{X_i}| \geq 2$ is the number of (finite) categories of X_i ($|\cdot|$ is the cardinality of a set or vector, and the notation $\doteq$ is used to indicate a definition instead of a mathematical equality). Bold letters are used to emphasize sets or vectors. For example, $\mathbf{x} \in \Omega_\mathbf{X} \doteq \times_{X\in\mathbf{X}}\Omega_X$, for $\mathbf{X} \subseteq \mathcal{X}$, is an instantiation for all the variables in $\mathbf{X}$. Furthermore, $r_{\Pi_i} \doteq |\Omega_{\Pi_i}| = \prod_{X_t\in\Pi_i} r_t$ is the number of possible instantiations of the parent set Π_i of X_i, and $\theta = (\theta_{ijk})_{\forall ijk}$ is the entire vector of parameters such that the elements are $\theta_{ijk} = p(x_{ik}|\pi_{ij})$, with $i \in \{1,\ldots,n\}$, $j \in \{1,...,r_{\Pi_i}\}$, $k \in \{1,...,r_i\}$, and $\pi_{ij} \in \Omega_{\Pi_i}$.

Because of the Markov condition, the Bayesian network represents a joint probability distribution by the expression $p(\mathbf{x}) = p(x_1,\ldots,x_n) = \prod_i p(x_i|\pi_i)$, for every $\mathbf{x} \in \Omega_\mathcal{X}$, where every x_i and π_i are consistent with $\mathbf{x}$.

Given a complete data set $D = \{D_1,\ldots,D_N\}$ with N instances, where $D_u \doteq \mathbf{x}_u \in \Omega_\mathcal{X}$ is an instantiation of all the variables, the goal of structure learning is to find a DAG $\mathcal{G}$ that maximizes a given score function, that is, we look for $\mathcal{G}^* = \text{argmax}_{\mathcal{G}\in\mathcal{G}}\, s_D(\mathcal{G})$, with $\mathcal{G}$ the set of all DAGs with

nodes $\mathcal{X}$, for a given score function s_D (the dependency on data is indicated by the subscript D).[1] In this paper, we consider some well-known score functions: the Bayesian Information Criterion (BIC) (Schwarz, 1978) (which is equivalent to the *Minimum Description Length*), the Akaike Information Criterion (AIC) (Akaike, 1974), and the Bayesian Dirichlet (BD) (Cooper and Herskovits, 1992), which has as subcases BDe and BDeu (Buntine, 1991; Cooper and Herskovits, 1992; Heckerman et al., 1995). As done before in the literature, we assume parameter independence and modularity (Heckerman et al., 1995). The score functions based on BIC and AIC differ only in the weight that is given to the penalty term:

$$BIC/AIC: \quad s_D(\mathcal{G}) = \max_{\theta} L_{\mathcal{G},D}(\theta) - t(\mathcal{G}) \cdot w,$$

where $t(\mathcal{G}) = \sum_{i=1}^{n}(r_{\Pi_i} \cdot (r_i - 1))$ is the number of free parameters, $w = \frac{\log N}{2}$ for BIC and $w = 1$ for AIC, $L_{\mathcal{G},D}$ is the log-likelihood function with respect to data D and graph $\mathcal{G}$:

$$L_{\mathcal{G},D}(\theta) = \log \prod_{i=1}^{n} \prod_{j=1}^{r_{\Pi_i}} \prod_{k=1}^{r_i} \theta_{ijk}^{n_{ijk}},$$

where n_{ijk} indicates how many elements of D contain both x_{ik} and π_{ij}. Note that the values $(n_{ijk})_{\forall ijk}$ depend on the graph $\mathcal{G}$ (more specifically, they depend on the parent set Π_i of each X_i), so a more precise notation would be to use $n_{ijk}^{\Pi_i}$ instead of n_{ijk}. We avoid this heavy notation for simplicity unless necessary in the context. Moreover, we know that $\theta^* = (\theta^*_{ijk})_{\forall ijk} = (\frac{n_{ijk}}{n_{ij}})_{\forall ijk} = \text{argmax}_\theta L_{\mathcal{G},D}(\theta)$, with $n_{ij} = \sum_k n_{ijk}$.[2]

In the case of the BD criterion, the idea is to compute a score based on the posterior probability of the structure $p(\mathcal{G}|D)$. For that purpose, the following score function is used:

$$BD: \quad s_D(\mathcal{G}) = \log\left(p(\mathcal{G}) \cdot \int p(D|\mathcal{G},\theta) \cdot p(\theta|\mathcal{G}) d\theta\right),$$

where the logarithmic is often used to simplify computations, $p(\theta|\mathcal{G})$ is the prior of θ for a given graph $\mathcal{G}$, assumed to be a Dirichlet with hyper-parameters $\alpha = (\alpha_{ijk})_{\forall ijk}$ (which are assumed to be strictly positive):

$$p(\theta|\mathcal{G}) = \prod_{i=1}^{n} \prod_{j=1}^{r_{\Pi_i}} \Gamma(\alpha_{ij}) \prod_{k=1}^{r_i} \frac{\theta_{ijk}^{\alpha_{ijk}-1}}{\Gamma(\alpha_{ijk})},$$

where $\alpha_{ij} = \sum_k \alpha_{ijk}$. Hyper-parameters $(\alpha_{ijk})_{\forall ijk}$ also depend on the graph $\mathcal{G}$, and we indicate it by $\alpha_{ijk}^{\Pi_i}$ if necessary in the context. From now on, we also omit the subscript D. We assume that there is no preference for any graph, so $p(\mathcal{G})$ is uniform and vanishes in the computations. Under the assumptions, it has been shown (Cooper and Herskovits, 1992) that for multinomial distributions,

$$s(\mathcal{G}) = \log \prod_{i=1}^{n} \prod_{j=1}^{r_{\Pi_i}} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + n_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})}.$$

The BDe score (Heckerman et al., 1995) assumes that $\alpha_{ijk} = \alpha^* \cdot p(\theta_{ijk}|\mathcal{G})$, where α^* is the hyper-parameter known as the Equivalent Sample Size (ESS), and $p(\theta_{ijk}|\mathcal{G})$ is the prior probability for

1. In case of many optimal DAGs, then we assume to have no preference and argmax returns one of them.
2. If $n_{ij} = 0$, then $n_{ijk} = 0$ and we assume the fraction $\frac{n_{ijk}}{n_{ij}}$ to be equal to one.

$(x_{ik} \wedge \pi_{ij})$ given $\mathcal{G}$ (or simply given Π_i). The BDeu score (Buntine, 1991; Cooper and Herskovits, 1992) assumes further that local priors are such that α_{ijk} becomes $\frac{\alpha^*}{r_{\Pi_i} r_i}$ and α^* is the only free hyper-parameter.

An important property of all such criteria is that their functions are decomposable and can be written in terms of the local nodes of the graph, that is, $s(\mathcal{G}) = \sum_{i=1}^{n} s_i(\Pi_i)$, such that

$$BIC/AIC: \quad s_i(\Pi_i) = \max_{\theta_i} L_{\Pi_i}(\theta_i) - t_i(\Pi_i) \cdot w, \tag{1}$$

where $L_{\Pi_i}(\theta_i) = \sum_{j=1}^{r_{\Pi_i}} \sum_{k=1}^{r_i} n_{ijk} \log \theta_{ijk}$, and $t_i(\Pi_i) = r_{\Pi_i} \cdot (r_i - 1)$. And similarly,

$$BD: \quad s_i(\Pi_i) = \sum_{j=1}^{r_{\Pi_i}} \left(\log \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + n_{ij})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} \right). \tag{2}$$

In the case of BIC and AIC, Equation (1) is used to compute the global score of a graph using the local scores at each node, while Equation (2) is employed for BD, BDe and BDeu, using the respective hyper-parameters α.

3. Structural Constraints

A way to reduce the space of possible DAGs is to consider some constraints provided by experts. We work with structural constraints that specify where arcs may or may not be included. These constraints help to reduce the search space and are available in many situations. Moreover, we show examples in Sections 3.1 and 3.2 of how these constraints can be used to learn structures of different types of networks, such as naive Bayes, tree-augmented naive Bayes, and Dynamic Bayesian networks. We work with the following rules, used to build up the structural constraints:

- $indegree(X_j, k, op)$, where $op \in \{\text{lt}, \text{eq}\}$ and k an integer, means that the node X_j must have *less than* (when $op = \text{lt}$) or *equal to* (when $op = \text{eq}$) k parents.
- $arc(X_i, X_j)$ indicates that the node X_i must be a parent of X_j.
- Operators *or* ($\vee$) and *not* ($\neg$) are used to form the rules. The *and* operator is not explicitly used as we assume that each constraint is in disjunctive normal form.

The structural constraints can be imposed locally as long as they involve just a single node and its parents. In essence, parent sets of a node X_i that do violate a constraint are never processed nor stored, and this can be checked locally when one is about to compute the local score. On the other hand, constraints such as $(arc(X_1, X_2) \vee arc(X_2, X_3))$ cannot be imposed locally, as it defines a non-local condition (the arcs go to distinct variables, namely X_2 and X_3). In this work we assume that constraints are local. Besides constraints devised by an expert, one might use constraints to force the learning procedure to obtain specialized types of networks. The next two subsections describe (somewhat non-trivial) examples of use of constraints to learn different types of networks. Specialized networks tend to be easier to learn, because the search space is already reduced to the structures that satisfy the underlying constraints. Notwithstanding, the readers who are only interested in learning general Bayesian networks might want to skip the rest of this section and continue from Section 4.

3.1 Learning Naive and TAN structures

For example, the constraints $\forall_{i\neq c, j\neq c}\ \neg arc(X_i, X_j)$ and $indegree(X_c, 0, \text{eq})$ impose that only arcs from node X_c to the others are possible, and that X_c is a root node, that is, a Naive Bayes structure will be learned. A learning procedure would in fact act as a feature selection procedure by letting some variables unlinked. Note that the symbol $\forall$ just employed is not part of the language but is used for easy of expose (in fact it is necessary to write down every constraint defined by such construction). As another example, the constraints $\forall_{j\neq c}\ indegree(X_j, 3, \text{lt})$, $indegree(X_c, 0, \text{eq})$, and $\forall_{j\neq c}\ indegree(X_j, 0, \text{eq}) \vee arc(X_c, X_j)$ ensure that all nodes have X_c as parent, or no parent at all. Besides X_c, each node may have at most one other parent, and X_c is a root node. This learns the structure of a Tree-augmented Naive (TAN) classifier, also performing a kind of feature selection (some variables may end up unlinked). In fact, it learns a forest of trees, as we have not imposed that all variables must be linked. In Section 6 we present some experimental results which indicate that learning TANs is a much easier (still very important) practical situation.

We point out that learning structures of networks with the particular purpose of building a classifier can be also tackled by other score functions that consider conditional distributions (Pernkopf and Bilmes, 2005). Here we present a way to learn TANs considering the fit of the joint distribution, which can be done by constraints. Further discussions about learning classifiers is not the aim of this work.

3.2 Learning Dynamic Bayesian Networks

A more sophisticated application of structural constraints is presented in this section, where they are employed to translate the structure learning in Dynamic Bayesian Networks (DBNs) to a corresponding problem in Bayesian networks. While Bayesian networks are not directly related to time, DBNs are used to model temporal processes. Assuming Markovian and stationary properties, DBNs may be encoded in a very compact way and inferences are executed quickly. They are built over a collection of sets of random variables $\{\mathcal{X}^0, \mathcal{X}^1, \ldots, \mathcal{X}^T\}$ representing variables in different times $0, 1, \ldots, T$ (we assume that time is discrete). A Markovian property holds, which ensures that $p(\mathcal{X}^{t+1}|\mathcal{X}^0, \ldots \mathcal{X}^t) = p(\mathcal{X}^{t+1}|\mathcal{X}^t)$, for $0 \leq t < T$. Furthermore, because the process is assumed to be stationary, we have that $p(\mathcal{X}^{t+1}|\mathcal{X}^t)$ is independent of t, that is, $p(\mathcal{X}^{t+1}|\mathcal{X}^t) = p(\mathcal{X}^{t'+1}|\mathcal{X}^{t'})$ for any $0 \leq t, t' < T$. This means that a DBN is just as a collection of Bayesian networks that share the same structure and parameters (apart from the initial Bayesian network for time zero). If $X_i^t \in \mathcal{X}^t$ are the variables at time t, a DBN may have arcs between nodes X_i^t of the same time t and arcs from nodes X_i^{t-1} (previous time) to nodes X_i^t of time t. Hence, a DBN can be viewed as two-slice temporal Bayesian network, where at time zero, we have a standard Bayesian network as in Section 2, which we denote $\mathcal{B}^o$, and for slices 1 to T we have another Bayesian network (called *transitional* Bayesian network and denoted simply $\mathcal{B}$) defined over the same variables but where nodes may have parents on two consecutive slices, that is, $\mathcal{B}$ precisely defines the distributions $p(\mathcal{X}^{t+1}|\mathcal{X}^t)$, for any $0 \leq t < T$.

To learn a DBN, we assume that many temporal sequences of data are available. Thus, a complete data set $D = \{D_1, \ldots, D_N\}$ is composed of N sequences, where each D_u is composed of instances $D_u^t \doteq \mathbf{x_u^t} = \{x_{u,1}^t, \ldots, x_{u,n}^t\}$, for $t = 0, \ldots, T$ (where T is the total number of slices/frames apart from the initial one). Note that there is an implicit order among the elements of each D_u. We denote by $D^0 \doteq \{D_u^0 :\ 1 \leq u \leq N\}$ the data of the first slice, and by $D^t \doteq \{(D_u^t, D_u^{t-1}) :\ 1 \leq u \leq N\}$, with $1 \leq t \leq T$, the data of a slice t (note that the data of the slice $t-1$ is also included, be-

cause it is necessary for learning the transitions). As the conditional probability distributions for time $t > 0$ share the same parameters, we can unroll the DBN to obtain the factorization $p(\mathcal{X}^{1:T}) = \prod_i p^0(X_i^0|\Pi_i^0)\prod_{t=1}^T \prod_i p(X_i^t|\Pi_i^t)$, where $p^0(X_i^0|\Pi_i^0)$ are the local conditional distributions of $\mathcal{B}^0$, X_i^t and Π_i^t represent the corresponding variables in time t, and $p(X_i^t|\Pi_i^t)$ are the local distributions of $\mathcal{B}$.

Unfortunately learning a DBN is at least as hard as learning a Bayesian network, because the former can be viewed as a generalization of the latter. Still, we show that the same method used for Bayesian networks can be used to learn DBNs. With complete data, learning parameters of DBNs is similar to learning parameters of Bayesian networks, but we deal with counts n_{ijk} for both $\mathcal{B}^0$ and $\mathcal{B}$. The counts related to $\mathcal{B}^0$ are obtained from the first slice of each sequence, so there are N samples overall, while counts for $\mathcal{B}$ are obtained from the whole time sequences, so there are $N \cdot T$ elements to consider (supposing that each sequence has the same length T, for ease of expose). The score function of a given structure decomposes between the score function of $\mathcal{B}^0$ and the score function of $\mathcal{B}$ (because of the decomposability of score functions), so we look for graphs such that

$$(\mathcal{G}^{0*}, \mathcal{G}'^{*}) = \underset{\mathcal{G}^0, \mathcal{G}'}{\operatorname{argmax}} \left(s_{D^0}(\mathcal{G}^0) + s_{D^{1:T}}(\mathcal{G}')\right) = (\underset{\mathcal{G}^0}{\operatorname{argmax}}\, s_{D^0}(\mathcal{G}^0), \underset{\mathcal{G}'}{\operatorname{argmax}}\, s_{D^{1:T}}(\mathcal{G}')), \tag{3}$$

where G^0 is a graph over $\mathcal{X}^0$ and $\mathcal{G}'$ is a graph over variables $\mathcal{X}^t, \mathcal{X}^{t-1}$ of a generic slice t and its predecessor $t-1$. Counts are obtained from data sets with time sequences separately for the initial and the transitional Bayesian networks, and the problem reduces to the learning problem in a Bayesian network with some constraints that force the arcs to respect the DBN's stationarity and Markovian characteristics (of course, it is necessary to obtain the counts from the data in a particular way). We make use of the constraints defined in Section 3 to develop a simple transformation of the structure learning problem to a corresponding structure learning problem in an augmented Bayesian network. The steps of this procedure are as follows:

1. Learn $\mathcal{B}^0$ using the data set D^0. Note that this is already a standard Bayesian network structure learning problem, so we obtain the graph $\mathcal{G}^0$ for the first maximization of Equation (3).

2. Suppose there is a Bayesian network $\mathcal{B}' = (\mathcal{G}', \mathcal{X}', \mathcal{P}')$ with twice as many nodes as $\mathcal{B}^0$. Denote the nodes as $(X_1, \ldots, X_n, X_1', \ldots, X_n')$. Construct a new data set D' that is composed by $N \cdot T$ elements $\{D^1, \ldots, D^T\}$. Note that D' is precisely a data set over $2n$ variables, because it is formed of pairs (D_u^{t-1}, D_u^t), which are complete instantiations for the variables of $\mathcal{B}'$, containing the elements of two consecutive slices.

3. Include structural constraints as follows:

$$\forall_{1 \le i \le n}\ arc(X_i, X_i'), \tag{4}$$

$$\forall_{1 \le i \le n}\ indegree(X_i, 0, eq). \tag{5}$$

   Equation (4) forces the time relation between the same variable in consecutive time slices (in fact this constraint might be discarded if someone does not want to enforce each variable to be correlated to itself of the past slice). Equation (5) forces the variables $X_1, \ldots, X_n$ to have no parents (these are the variables that are simulating the previous slice, while the variables $\mathcal{X}'$ are simulating the current slice).

4. Learn $\mathcal{B}'$ using the data set D' with an standard Bayesian network structure learning procedure, capable of enforcing the structural constraints. Note that the parent sets of $X_1,\ldots,X_n$ are already fixed to be empty, so the output graph will maximize the scores associated only to nodes $\mathcal{X}'$: $\text{argmax}_{\mathcal{G}'} s_{D^{1:T}}(\mathcal{G}')) =$

$$\underset{\mathcal{G}'}{\text{argmax}} \left(\sum_i s_{i,D^{1:T}}(\Pi_i) + \sum_{i'} s_{i',D^{1:T}}(\Pi'_i) \right) = \underset{\mathcal{G}'}{\text{argmax}} \sum_{i'} s_{i',D^{1:T}}(\Pi'_i).$$

This holds because of the decomposability of the score function among nodes, so that the scores of the nodes $X_1,\ldots,X_n$ are fixed and can be disregarded in the maximization (they are constant).

5. Take the subgraph of $\mathcal{G}'$ corresponding to the variables $X'_1,\ldots,X'_n$ to be the graph of the transitional Bayesian network $\mathcal{B}$. This subgraph has arcs among $X'_1,\ldots,X'_n$ (which are arcs correlating variables of the same time slice) as well as arcs from the previous slice to the nodes $X'_1,\ldots,X'_n$.

Therefore, after applying this transformation, the structure learning problem in a DBN can be performed by two calls to the method that solves the problem in a Bayesian network. We point out that an expert may create her/his own constraints to be used during the learning, besides those constraints introduced by the transformation, as long as such constraints do not violate the DBN implicit constraints. This makes possible to learn DBNs together with expert's knowledge in the form of structural constraints.

4. Properties of the Score Functions

In this section we present mathematical properties that are useful when computing score functions. Local scores need to be computed many times to evaluate the candidate graphs when we look for the best graph. Because of decomposability, we can avoid to compute such functions several times by creating a cache that contains $s_i(\Pi_i)$ for each X_i and each parent set Π_i. Note that this cache may have an exponential size on n, as there are 2^{n-1} subsets of $\{X_1,\ldots,X_n\}\setminus\{X_i\}$ to be considered as parent sets. This gives a total space and time of $O(n\cdot 2^n\cdot v)$ to build the cache, where v is the worst-case asymptotic time to compute the local score function at each node.[3] Instead, we describe a collection of results that are used to obtain much smaller caches in many practical cases.

First, Lemma 1 is quite simple but very useful to discard elements from the cache of each node X_i. It holds for all score functions that we treat in this paper. It was previously stated in Teyssier and Koller (2005) and de Campos et al. (2009), among others.

Lemma 1 *Let X_i be a node of $\mathcal{G}'$, a candidate DAG for a Bayesian network where the parent set of X_i is Π'_i. Suppose $\Pi_i \subset \Pi'_i$ is such that $s_i(\Pi_i) > s_i(\Pi'_i)$ (where s is one of BIC, AIC, BD or derived criteria). Then Π'_i is not the parent set of X_i in an optimal DAG $\mathcal{G}^*$.*

Proof This fact comes straightforward from the decomposability of the score functions. Take a graph $\mathcal{G}$ that differs from $\mathcal{G}'$ only on the parent set of X_i, where it has Π_i instead of Π'_i. Note that $\mathcal{G}$

3. Note that the time to compute a single local score might be large depending on the number of parents but still asymptotically bounded by the data set size.

is also a DAG (as $\mathcal{G}$ is a subgraph of $\mathcal{G}'$ built from the removal of some arcs, which cannot create cycles) and $s(\mathcal{G}) = \sum_{j \neq i} s_j(\Pi'_j) + s_i(\Pi_i) > \sum_{j \neq i} s_j(\Pi'_j) + s_i(\Pi'_i) = s(\mathcal{G}')$. Any DAG $\mathcal{G}'$ with parent set Π'_i for X_i has a subgraph $\mathcal{G}$ with a better score than that of $\mathcal{G}'$, and thus Π'_i is not the optimal parent configuration for X_i in $\mathcal{G}^*$. ■

Unfortunately Lemma 1 does not tell us anything about supersets of Π'_i, that is, we still need to compute scores for all the possible parent sets and later verify which of them can be removed. This would still leave us with $n \cdot 2^n \cdot v$ asymptotic time and space requirements (although the space would be reduced after applying the lemma). The next two subsections present results to avoid all such computations. BIC and AIC are treated separately from BD and derivatives (reasons for that will become clear in the derivations).

4.1 BIC and AIC Score Properties

Next theorems handle the issue of having to compute scores for all possible parent sets, when one is using BIC or AIC criteria. BD scores are dealt later on.

Theorem 2 *Using BIC or AIC as score function, suppose that X_i, Π_i are such that $r_{\Pi_i} > \frac{N}{w} \frac{\log r_i}{r_i - 1}$. If Π'_i is a proper superset of Π_i, then Π'_i is not the parent set of X_i in an optimal structure.*

Proof [4] We know that Π'_i contains at least one additional node, that is, $\Pi'_i \supseteq \Pi_i \cup \{X_e\}$ and $X_e \notin \Pi_i$. Because $\Pi_i \subset \Pi'_i$, $L_i(\Pi'_i)$ is certainly greater than or equal to $L_i(\Pi_i)$, and $t_i(\Pi'_i)$ will certainly be greater than the corresponding value $t_i(\Pi_i)$ in $\mathcal{G}$. The difference in the scores is $s_i(\Pi'_i) - s_i(\Pi_i)$, which equals to (see the explanations after the formulas):

$$\begin{aligned}
&\max_{\theta'_i} L_i(\Pi'_i) - t_i(\Pi'_i) - (\max_{\theta_i} L_i(\Pi_i) - t_i(\Pi_i)) \leq \\
&\qquad - \max_{\theta_i} L_i(\Pi_i) - t_i(\Pi'_i) + t_i(\Pi_i) = \\
&\sum_{j=1}^{r_{\Pi_i}} n_{ij} \left(- \sum_{i=1}^{r_i} \frac{n_{ijk}}{n_{ij}} \log \frac{n_{ijk}}{n_{ij}} \right) - t_i(\Pi'_i) + t_i(\Pi_i) \leq \\
&\qquad \sum_{j=1}^{r_{\Pi_i}} n_{ij} H(\theta_{ij}) - t_i(\Pi'_i) + t_i(\Pi_i) \leq \\
&\qquad \sum_{j=1}^{r_{\Pi_i}} n_{ij} \log r_i - r_{\Pi_i} \cdot (r_e - 1) \cdot (r_i - 1) \cdot w \leq \\
&\sum_{j=1}^{r_{\Pi_i}} n_{ij} \log r_i - r_{\Pi_i} \cdot (r_i - 1) \cdot w = N \log r_i - r_{\Pi_i} \cdot (r_i - 1) \cdot w.
\end{aligned}$$

The first step uses the fact that $L_i(\Pi'_i)$ is negative, so we drop it, the second step uses the fact that $\theta^*_{ijk} = \frac{n_{ijk}}{n_{ij}}$, with $n_{ij} = \sum_{i=1}^{r_i} n_{ijk}$, the third step uses the definition of entropy $H(\cdot)$ of a discrete distribution, and the fourth step uses the fact that the entropy of a discrete distribution is less than the log of its number of categories. Finally, the last equation is negative if $r_{\Pi_i} \cdot (r_i - 1) \cdot w > N \log r_i$, which

4. Another similar proof appears in Bouckaert (1994), but it leads directly to the conclusion of Corollary 3. The intermediate result is algorithmically important.

is exactly the hypothesis of the theorem. Hence $s_i(\Pi_i') < s_i(\Pi_i)$, and Lemma 1 guarantees that Π_i' cannot be the parent set of X_i in an optimal structure. ■

Corollary 3 *Using BIC or AIC as criterion, the optimal graph $\mathcal{G}$ has at most $O(\log N)$ parents per node.*

Proof Assuming $N > 4$, we have $\frac{\log r_i}{w(r_i-1)} < 1$ (because w is either 1 or $\frac{\log N}{2}$). Take a variable X_i and a parent set Π_i with exactly $\lceil \log_2 N \rceil$ elements. Because every variable has at least two states, we know that $r_{\Pi_i} \geq 2^{|\Pi_i|} \geq N > \frac{N}{w} \frac{\log r_i}{r_i - 1}$, and by Theorem 2 we know that no proper superset of Π_i can be an optimal parent set. ■

Theorem 2 and Corollary 3 ensures that the cache stores at most $O(\sum_{t=0}^{\lceil \log_2 N \rceil} \binom{n-1}{t})$ elements for each variable (all combinations up to $\lceil \log_2 N \rceil$ parents). Next lemma does not help us to improve the theoretical size bound that is achieved by Corollary 3, but it is quite useful in practice because it is applicable even in cases where Theorem 2 is not, implying that fewer parent sets need to be inspected.

Theorem 4 *Let BIC or AIC be the score criterion and let X_i be a node with $\Pi_i \subset \Pi_i'$ two possible parent sets such that $t_i(\Pi_i') + s_i(\Pi_i) > 0$. Then Π_i' and all supersets $\Pi_i'' \supset \Pi_i'$ are not optimal parent configurations for X_i.*

Proof We have that $t_i(\Pi_i') + s_i(\Pi_i) > 0 \Rightarrow -t_i(\Pi_i') - s_i(\Pi_i) < 0$, and because $L_i(\cdot)$ is a negative function, it implies

$$\Rightarrow (L_i(\Pi_i') - t_i(\Pi_i')) - s_i(\Pi_i) < 0 \Rightarrow s_i(\Pi_i') < s_i(\Pi_i).$$

Using Lemma 1, we have that Π_i' is not the optimal parent set for X_i. The result also follows for any $\Pi_i'' \supset \Pi_i$, as we know that $t_i(\Pi_i'') > t_i(\Pi_i')$ and the same argument suffices. ■

Theorem 4 provides a bound to discard parent sets without even inspecting them. The idea is to verify the assumptions of Theorem 4 every time the score of a parent set Π_i of X_i is about to be computed by taking the best score of any subset and testing it against the theorem. Only subsets that have been checked against the structural constraints can be used, that is, a subset with high score but that violates constraints cannot be used as the "certificate" to discard its supersets (in fact, it is not a valid parent set at first). This ensures that the results are valid even in the presence of constraints. Whenever the theorem can be applied, Π_i is discard and all its supersets are not even inspected. This result allows us to stop computing scores earlier than the worst-case, reducing the number of computations to build and store the cache. Π_i is also checked against Lemma 1 (which is stronger in the sense that instead of a bounding function, the actual scores are directly compared). However Lemma 1 cannot help us to avoid analyzing the supersets of Π_i.

4.2 BD Score Properties

First note that the BD scores can be rewritten as:

$$s_i(\Pi_i) = \sum_{j \in J_i} \left(\log \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + n_{ij})} + \sum_{k \in K_{ij}} \log \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} \right),$$

where $J_i \doteq J_i^{\Pi_i} \doteq \{1 \leq j \leq r_{\Pi_i} :\ n_{ij} \neq 0\}$, because $n_{ij} = 0$ implies that all terms cancel each other. In the same manner, $n_{ijk} = 0$ implies that the terms of the internal summation cancel out, so let $K_{ij} \doteq K_{ij}^{\Pi_i} \doteq \{1 \leq k \leq r_i :\ n_{ijk} \neq 0\}$ be the indices of the categories of X_i such that $n_{ijk} \neq 0$. Let $K_i^{\Pi_i} \doteq \cup_j K_{ij}^{\Pi_i}$ be a vector with all indices corresponding to non-zero counts for Π_i (note that the symbol $\cup$ must be seen as a concatenation of vectors, as we allow $K_i^{\Pi_i}$ to have repetitions). The counts n_{ijk} (and consequently $n_{ij} = \sum_k n_{ijk}$) are completely defined if we know the parent set Π_i. Rewrite the score as follows:

$$s_i(\Pi_i) = \sum_{j \in J_i} \left(f(K_{ij}, (\alpha_{ijk})_{\forall k}) + g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) \right),$$

with

$$f(K_{ij}, (\alpha_{ijk})_{\forall k}) = \log\Gamma(\alpha_{ij}) - \sum_{k \in K_{ij}} \log\Gamma(\alpha_{ijk}),$$
$$g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) = -\log\Gamma(\alpha_{ij} + n_{ij}) + \sum_{k \in K_{ij}} \log\Gamma(\alpha_{ijk} + n_{ijk}).$$

We do not need K_{ij} as argument of $g(\cdot)$ because the set of non-zero n_{ijk} is known from the counts $(n_{ijk})_{\forall k}$ that are already available as arguments of $g(\cdot)$. To achieve the desired theorem that will be able to reduce the computational time to build the cache, some intermediate results are necessary.

Lemma 5 *Let Π_i be the parent set of X_i, $(\alpha_{ijk})_{\forall ijk} > 0$ be the hyper-parameters, and integers $(n_{ijk})_{\forall ijk} \geq 0$ be counts obtained from data. We have that $g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) \leq -\log\Gamma(v) \approx 0.1214$ if $n_{ij} \geq 1$, where $v = \operatorname{argmax}_{x>0} -\log\Gamma(x) \approx 1.4616$. Furthermore, $g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) \leq -\log\alpha_{ij} + \log\alpha_{ijk} - f(K_{ij}, (\alpha_{ijk})_{\forall k})$ if $|K_{ij}| = 1$.*

Proof We use the relation $\Gamma(x + \sum_k a_k) \geq \Gamma(x+1)\prod_k \Gamma(a_k)$, for $x \geq 0, \forall_k a_k \geq 1$ and $\sum_k a_k \geq 1$ (note that it is valid even if there is a single element in the summation). This relation comes from the Beta function inequality:

$$\frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)} \leq \frac{x+y}{xy} \implies \Gamma(x+1)\Gamma(y+1) \leq \Gamma(x+y+1),$$

where $x, y > 0$. Applying the transformation $y + 1 = \sum_t a_t$ (which is possible because $\sum_t a_t > 1$ and thus $y > 0$), we obtain:

$$\Gamma(x + \sum_t a_t) \geq \Gamma(x+1)\Gamma(\sum_t a_t) \geq \Gamma(x+1)\prod_t \Gamma(a_t),$$

(the last step is due to $a_t \geq 1$ for all t, so the same relation of the Beta function can be overall applied, because $\Gamma(x+1)\Gamma(y+1) \leq \Gamma(x+y+1) \leq \Gamma(x+1+y+1)$).

With the relation just devised in hands, we have

$$\frac{\Gamma(\alpha_{ij} + n_{ij})}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} = \frac{\Gamma(\sum_{1 \leq k \leq r_i} (\alpha_{ijk} + n_{ijk}))}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} =$$

$$= \frac{\Gamma(\sum_{k \notin K_{ij}} \alpha_{ijk} + \sum_{k \in K_{ij}} (\alpha_{ijk} + n_{ijk}))}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} \geq \Gamma(1 + \sum_{k \notin K_{ij}} \alpha_{ijk}),$$

obtained by renaming $x = \sum_{k \notin K_{ij}} \alpha_{ijk}$ and $a_k = \alpha_{ijk} + n_{ijk}$ (we have that $\sum_{k \in K_{ij}} (\alpha_{ijk} + n_{ijk}) \geq n_{ij} \geq 1$ and each $a_k \geq 1$). Thus

$$g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) = -\log \frac{\Gamma(\alpha_{ij} + n_{ij})}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} \leq -\log \Gamma(1 + \sum_{k \notin K_{ij}} \alpha_{ijk}).$$

Because $v = \text{argmax}_{x>0} - \log \Gamma(x)$, we have $-\log \Gamma(1 + \sum_{k \notin K_{ij}} \alpha_{ijk}) \leq -\log \Gamma(v)$.

Now, the second part of the lemma. If $|K_{ij}| = 1$, then let $K_{ij} = \{k\}$. We know that $n_{ij} \geq 1$ and thus

$$g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) = -\log \frac{\Gamma(\alpha_{ij} + n_{ij})}{\Gamma(\alpha_{ijk} + n_{ij})} = -\log \left(\frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ijk})} \prod_{t=0}^{n_{ij}-1} \frac{(\alpha_{ij} + t)}{(\alpha_{ijk} + t)} \right) =$$

$$= -f(K_{ij}, (\alpha_{ijk})_{\forall k}) - \log \frac{\alpha_{ij}}{\alpha_{ijk}} - \sum_{t=1}^{n_{ij}-1} \log \frac{(\alpha_{ij} + t)}{(\alpha_{ijk} + t)} \leq -\log \alpha_{ij} + \log \alpha_{ijk} - f(K_{ij}, (\alpha_{ijk})_{\forall k}),$$

because $\frac{(\alpha_{ij}+t)}{(\alpha_{ijk}+t)} \geq 1$ for every t. ■

Lemma 6 *Let Π_i be the parent set of X_i, $(\alpha_{ijk})_{\forall ijk} > 0$ be the hyper-parameters, and integers $(n_{ijk})_{\forall ijk} \geq 0$ be counts obtained from data. We have that $g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) \leq 0$ if $n_{ij} \geq 2$.*

Proof If $n_{ij} \geq 2$, we use the relation $\Gamma(x + \sum_k a_k) \geq \Gamma(x+2) \prod_k \Gamma(a_k)$, for $x \geq 0$, $\forall_k a_k \geq 1$ and $\sum_k a_k \geq 2$. This inequality is obtained in the same way as in Lemma 5, but using a tighter Beta function bound:

$$\mathcal{B}(x,y) \leq \frac{x+y}{xy} \left(\frac{(x+1)(y+1)}{x+y+1} \right)^{-1} \implies \Gamma(x+2)\Gamma(y+2) \leq \Gamma(x+y+2),$$

and the relation follows by using $y + 2 = \sum_t a_t$ and the same derivation as before. Now,

$$\frac{\Gamma(\alpha_{ij} + n_{ij})}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} = \frac{\Gamma(\sum_{1 \leq k \leq r_i} (\alpha_{ijk} + n_{ijk}))}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} =$$

$$= \frac{\Gamma(\sum_{k \notin K_{ij}} \alpha_{ijk} + \sum_{k \in K_{ij}} (\alpha_{ijk} + n_{ijk}))}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} \geq \Gamma(2 + \sum_{k \notin K_{ij}} \alpha_{ijk}),$$

obtained by renaming $x = \sum_{k \notin K_{ij}} \alpha_{ijk}$ and $a_k = \alpha_{ijk} + n_{ijk}$, as we know that $\sum_{k \in K_{ij}} (\alpha_{ijk} + n_{ijk}) \geq n_{ij} \geq 2$ and each $a_k \geq 1$. Finally,

$$g((n_{ijk})_{\forall k}, (\alpha_{ijk})_{\forall k}) = -\log \frac{\Gamma(\alpha_{ij} + n_{ij})}{\prod_{k \in K_{ij}} \Gamma(\alpha_{ijk} + n_{ijk})} \leq -\log \Gamma(2 + \sum_{k \notin K_{ij}} \alpha_{ijk}) \leq 0,$$

because $\Gamma(2 + \sum_{k \notin K_{ij}} \alpha_{ijk}) \geq 1$. ■

Lemma 7 *Given a BD score and two parent sets Π_i^0 and Π_i for a node X_i such that $\Pi_i^0 \subset \Pi_i$, if*

$$s_i(\Pi_i^0) > \sum_{\substack{j \in J_i^{\Pi_i}: \\ |K_{ij}^{\Pi_i}| \geq 2}} f(K_{ij}^{\Pi_i}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) + \sum_{\substack{j \in J_i^{\Pi_i}: \\ |K_{ij}^{\Pi_i}| = 1}} \log \frac{\alpha_{ijk'}^{\Pi_i}}{\alpha_{ij}^{\Pi_i}},$$

then Π_i is not the optimal parent set for X_i.

Proof Using the results of Lemmas 5 and 6,

$$\begin{aligned}
s_i(\Pi_i) &= \sum_{j \in J_i} \left(f(K_{ij}^{\Pi_i}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) + g((n_{ijk}^{\Pi_i})_{\forall k}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) \right) \\
&\leq \sum_{j \in J_i:\ |K_{ij}^{\Pi_i}| \geq 2} \left(f(K_{ij}^{\Pi_i}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) + g((n_{ijk}^{\Pi_i})_{\forall k}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) \right) + \\
&\quad + \sum_{j \in J_i^{\Pi_i}:\ |K_{ij}^{\Pi_i}| = 1} \left(-\log \alpha_{ij}^{\Pi_i} + \log \alpha_{ijk'}^{\Pi_i} \right) \\
&\leq \sum_{j \in J_i^{\Pi_i}: |K_{ij}^{\Pi_i}| \geq 2} f(K_{ij}^{\Pi_i}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) + \sum_{j \in J_i^{\Pi_i}: |K_{ij}^{\Pi_i}| = 1} \log \frac{\alpha_{ijk'}^{\Pi_i}}{\alpha_{ij}^{\Pi_i}},
\end{aligned}$$

which by the assumption of this lemma, is less than $s_i(\Pi_i^0)$. Thus, we conclude that the parent set Π_i^0 has better score than Π_i, and the desired result follows from Lemma 1. ■

Lemma 8 *Given the BDeu score, $(\alpha_{ijk})_{\forall ijk} > 0$, and integers $(n_{ijk})_{\forall ijk} \geq 0$ such that $\alpha_{ij} \leq 0.8349$ and $|K_{ij}| \geq 2$ for a given j, then $f(K_{ij}, (\alpha_{ijk})_{\forall k}) \leq -|K_{ij}| \cdot \log r_i$.*

Proof Using $\alpha_{ijk} \leq \alpha_{ij} \leq 0.8349$ (for all k), we have

$$\begin{aligned}
f(K_{ij}, (\alpha_{ijk})_{\forall k}) &= \log \Gamma(\alpha_{ij}) - |K_{ij}| \log \Gamma(\frac{\alpha_{ij}}{r_i}) \\
&= \log \Gamma(\alpha_{ij}) - |K_{ij}| \log \Gamma(\frac{\alpha_{ij}}{r_i} + 1) + |K_{ij}| \log \frac{\alpha_{ij}}{r_i} \\
&= \log \Gamma(\alpha_{ij}) - |K_{ij}| \log \frac{\Gamma(\frac{\alpha_{ij}}{r_i} + 1)}{\alpha_{ij}} - |K_{ij}| \log r_i \\
&= |K_{ij}| \log \frac{\Gamma(\alpha_{ij})^{1/|K_{ij}|} \alpha_{ij}}{\Gamma(\frac{\alpha_{ij}}{r_i} + 1)} - |K_{ij}| \log r_i.
\end{aligned}$$

Now, $\Gamma(\alpha_{ij})^{1/|K_{ij}|} \alpha_{ij} \leq \Gamma(\frac{\alpha_{ij}}{r_i} + 1)$, because $r_i \geq 2$, $|K_{ij}| \geq 2$ and $\alpha_{ij} \leq 0.8349$ (this number can be computed by numerically solving the inequality for $r_i = |K_{ij}| = 2$). We point out that 0.8349 is a bound for α_{ij} that ensures this last inequality to hold when $r_i = |K_{ij}| = 2$, which is the worst-case scenario (greater values of r_i and $|K_{ij}|$ make the left-hand side decrease and the right-hand side increase). Because r_i of each node is known, tighter bounds might be possible according to the node. ■

Theorem 9 *Given the BDeu score and two parent sets Π_i^0 and Π_i for a node X_i such that $\Pi_i^0 \subset \Pi_i$ and $\alpha_{ij}^{\Pi_i} \leq 0.8349$ for every j, if $s_i(\Pi_i^0) > -|K_i^{\Pi_i}| \log r_i$ then neither Π_i nor any superset $\Pi_i' \supset \Pi_i$ are optimal parent sets for X_i.*

Proof We have that

$$s_i(\Pi_i^0) > -|K_i^{\Pi_i}| \log r_i = \sum_{j \in J_i^{\Pi_i}:\ |K_{ij}^{\Pi_i}| \geq 2} -|K_{ij}^{\Pi_i}| \log r_i + \sum_{j \in J_i^{\Pi_i}:\ |K_{ij}^{\Pi_i}| = 1} -\log r_i,$$

which by Lemma 8 is greater than or equal to

$$\sum_{j \in J_i^{\Pi_i}:\ |K_{ij}^{\Pi_i}| \geq 2} f(K_{ij}^{\Pi_i}, (\alpha_{ijk}^{\Pi_i})_{\forall k}) + \sum_{j \in J_i^{\Pi_i}:\ |K_{ij}^{\Pi_i}| = 1} -\log r_i.$$

Now, Lemma 7 suffices to show that Π_i is not a optimal parent set, because $-\log r_i = \log \frac{\alpha_{ijk}^{\Pi_i}}{\alpha_{ij}^{\Pi_i}}$ for any k. To show the result for any superset $\Pi_i' \supset \Pi_i$, we just have to note that $|K_i^{\Pi_i'}| \geq |K_i^{\Pi_i}|$ (because the overall number of non-zero counts can only increase when we include more parents), and $\alpha_{ij'}^{\Pi_i'}$ (for all j') are all less than 0.8349 (because the αs can only decrease when more parents are included), thus we can apply the very same reasoning to all supersets. ■

Theorem 9 provides a bound to discard parent sets without even inspecting them because of the non-increasing monotonicity of the employed bounding function when we increase the number of parents. As done for the BIC and AIC criteria, the idea is to check the validity of Theorem 9 every time the score of a parent set Π_i of X_i is about to be computed by taking the best score of any subset and testing it against the theorem (of course using only subsets that satisfy the structural constraints). Whenever possible, we discard Π_i and do not even look into all its supersets. Note that the assertion $\alpha_{ij} \leq 0.8349$ required by the theorem is not too restrictive, because as parent sets grow, as ESS is divided by larger numbers (it is an exponential decrease of the αs). Hence, the values α_{ij} become quickly below such a threshold. Furthermore, Π_i is also checked against Lemma 1 (although it does not help with the supersets). As we see later in the experiments, the practical size of the cache after the application of the properties is small even for considerably large networks, and both Lemma 1 and Theorem 9 help reducing the cache size, while Theorem 9 also help to reduce computations. Finally, we point out that Singh and Moore (2005) have already worked on bounds to reduce the number of parent sets that need to be inspected, but Theorem 9 provides a much tighter bound than their previous result, where the cut happens only after all $|K_{ij}^{\Pi_i}|$ go below two (or using their terminology, when *configurations are pure*).

5. Constrained B&B Algorithm

In this section we describe the branch-and-bound (B&B) algorithm used to find the best structure of the Bayesian network and comment on its complexity and correctness. The algorithm uses a B&B search where each case to be solved is a relaxation of a DAG, that is, the cases may contain cycles. At each step, a graph is picked up from a priority queue, and it is verified if it is a DAG. In such case, it is a feasible structure for the network and we compare its score against the best score so

far (which is updated if needed). Otherwise, there must be a directed cycle in the graph, which is then broken into subcases by forcing some arcs to be absent/present. Each subcase is put in the queue to be processed (these subcases cover all possible subgraphs related to the original case, that is, they cover all possible ways to break the cycle). The procedure stops when the queue is empty. Note that every time we break a cycle, the subcases that are created are independent, that is, their sets of graphs are disjoint. We obtain this fact by properly breaking the cycles to avoid overlapping among subcases (more details below). This is the same idea as in the inclusion-exclusion principle of combinatorics employed over the set of arcs that formed the cycle and ensures that we never process the same graph twice, and also ensures that all subgraphs are covered.

The initialization of the algorithm is as follows:

- $C : (X_i, \Pi_i) \rightarrow \mathcal{R}$ is the cache with the scores for all the variables and their possible parent configurations. This is constructed using a queue and analyzing parent sets according to the properties of Section 4, which saves (in practice) a large amount of space and time. All the structural constraints are considered in this construction so that only valid parent sets are stored.

- $\mathcal{G}$ is the graph created by taking the best parent configuration for each node without checking for acyclicity (so it is not necessarily a DAG), and s is the score of $\mathcal{G}$. This graph is used as an upper bound for the best possible graph, as it is clearly obtained from a relaxation of the problem (the relaxation comes from allowing cycles).

- $\mathcal{H}$ is an initially empty matrix containing, for each possible arc between nodes, a mark stating that the arc must be present, or is prohibited, or is free (may be present or not). This matrix controls the search of the B&B procedure. Each branch of the search has a $\mathcal{H}$ that specifies the graphs that still must be searched within that branch.

- Q is a priority queue of triples $(\mathcal{G}, \mathcal{H}, s)$, ordered by s (initially it contains a single triple with $\mathcal{G}$, $\mathcal{H}$ and s as mentioned. The order is such that the top of the queue contains always the triple of greatest s, while the bottom has the triple of smallest s.

- $(\mathcal{G}_{best}, s_{best})$ keeps at any moment the best DAG and score found so far. The value of s_{best} could be set to $-\infty$, but this best solution can also be initialized using any inner approximation method. For instance, we use a procedure that guesses an ordering for the variable, then computes the global best solution for that ordering, and finally runs a hill climbing over the resulting structure. All these procedures are very fast (given the small size of the pre-computed cache that we obtain in the previous steps). A good initial solution may significantly reduce the search of the B&B procedure, because it may give a lower bound closer to the upper bound defined by the relaxation $(\mathcal{G}, \mathcal{H}, s)$.

- *iter*, initialized with zero, keeps track of the iteration number. *bottom* is a user parameter that controls how frequent elements will be picked from the bottom of the queue instead of the usual removal from the top. For example, a value of 1 means to pick always from the bottom, a value of 2 alternates elements from the top and the bottom evenly, and a large value makes the algorithm picks always from the top.

The main loop of the B&B search is as follows:

- While Q is not empty, do

  1. Increment $iter$. If $\frac{iter}{bottom}$ is not an integer, then remove the top of Q and put into $(\mathcal{G}_{cur}, \mathcal{H}_{cur}, s_{cur})$. Otherwise remove the bottom of Q into $(\mathcal{G}_{cur}, \mathcal{H}_{cur}, s_{cur})$. If $s_{cur} \leq s_{best}$ (worse than an already known solution), then discard the current element and start the loop again.
  2. If $\mathcal{G}_{cur}$ is a DAG, then update $(\mathcal{G}_{best}, s_{best})$ with $(\mathcal{G}_{cur}, s_{cur})$, discard the current element and start the loop again (if $\mathcal{G}_{cur}$ came from the top of Q, then the algorithm stops—no other graph in the queue can be better than $\mathcal{G}_{cur}$).
  3. Take a cycle of $\mathcal{G}_{cur}$ (one must exist, otherwise we would have not reached this step), namely $v = (X_{a_1} \rightarrow X_{a_2} \rightarrow \ldots \rightarrow X_{a_{q+1}})$, with $a_1 = a_{q+1}$.
  4. For $y = 1, \ldots, q$, do
     (a) Mark on $\mathcal{H}_{cur}$ that the arc $X_{a_y} \rightarrow X_{a_{y+1}}$ is prohibited. This implies that the branch we are going to create will not have this cycle again.
     (b) Recompute $(\mathcal{G}, s)$ from $(\mathcal{G}_{cur}, s_{cur})$ such that the new parent set of $X_{a_{y+1}}$ in $\mathcal{G}$ complies with this new $\mathcal{H}_{cur}$. This is done by searching in the cache $C(X_{a_{y+1}}, \Pi_{a_{y+1}})$ for the best parent set. If there is a parent set in the cache that satisfies $\mathcal{H}_{cur}$, then
        – Include the triple $(\mathcal{G}, \mathcal{H}_{cur}, s)$ into Q.[5]
     (c) Mark on $\mathcal{H}_{cur}$ that the arc $X_{a_y} \rightarrow X_{a_{y+1}}$ must be present and that the sibling arc $X_{a_{y+1}} \rightarrow X_{a_y}$ is prohibited, and continue the loop of step 4. (*Step 4c forces the branches that we create to be disjoint among each other.*)

There are two considerations to show the correctness of the method. First, we need to guarantee that all the search space is considered, even though we do not explicitly search through all of it. Second, we must ensure that the same part of the search space is not processed more than once, so we do not lose time and know that the algorithm will finish with a best global graph. The search is conducted over all possible graphs (not necessarily DAGs). The queue Q contains the subspaces (of all possible graphs) to be analyzed. A triple $(\mathcal{G}, \mathcal{H}, s)$ indicates, through $\mathcal{H}$, which is this subspace. $\mathcal{H}$ is a matrix containing an indicator for each possible arc. It says if an arc is allowed (meaning it might or might not be present), prohibited (it cannot be present), or demanded (it must be present) in the current subspace of graphs. Thus, $\mathcal{H}$ completely defines the subspaces. $\mathcal{G}$ and s are respectively the best graph inside $\mathcal{H}$ (note that $\mathcal{G}$ might have cycles) and its score value (which is an upper bound for the best DAG in this subspace).

In the initialization step, Q begins with a triple where $\mathcal{H}$ indicates that every arc is allowed,[6] so all possible graphs are within the subspace of the initial $\mathcal{H}$. In this moment, the main loop starts and the only element of Q is put into $(\mathcal{G}_{cur}, \mathcal{H}_{cur}, s_{cur})$ and s_{cur} is compared against the best known score. Note that as $\mathcal{G}_{cur}$ is the graph with the greatest score that respects $\mathcal{H}_{cur}$, any other graph within the subspace defined by $\mathcal{H}_{cur}$ will have worse score. Therefore, if s_{cur} is less than the best known score, all this branch represented by $\mathcal{H}_{cur}$ may be discarded (this is the *bound* step). Certainly no graph within that subspace will be worth checking, because their scores are less than s_{cur}.

5. One may check the acyclicity of the graph before including the triple in the queue. We analyze this possibility later on.
6. In fact, the implementation may set $\mathcal{H}$ with possible known restrictions of arcs, that is, those that are known to be demanded or prohibited by structural constraints may be included in the initial $\mathcal{H}$.

If $\mathcal{G}_{cur}$ has score greater than s_{best}, then the graph $\mathcal{G}_{cur}$ is checked for cycles, as it may or may not be acyclic (all we know is that $\mathcal{G}_{cur}$ is a relaxed solution within the subspace $\mathcal{H}_{cur}$). If it is acyclic, then $\mathcal{G}_{cur}$ is the best graph so far. Moreover, if the acyclic $\mathcal{G}_{cur}$ was extracted from the top of Q, then the algorithm may stop, as all the other elements in the queue have lower score (this is guaranteed by the priority of the queue). Otherwise we restart the loop, as we cannot find a better graph within this subspace (the acyclic $\mathcal{G}_{cur}$ is already the best one by definition). On the other hand, if $\mathcal{G}_{cur}$ is cyclic, then we need to divide the space $\mathcal{H}_{cur}$ into smaller subcases with the aim of removing the cycles of $\mathcal{G}_{cur}$ (this is the *branch* step). Two characteristics must be kept by the branch step: (i) $\mathcal{H}_{cur}$ must be fully represented in the subcases (so we do not miss any graph), and (ii) the subcases must be disjoint (so we do not process the same graph more than once). A possible way to achieve these two requirements is as follows: let the cycle $v = (X_{a_1} \to X_{a_2} \to \ldots \to X_{a_{q+1}})$ be the one detected in $\mathcal{G}_{cur}$. We create q subcases such that

- The first subcase does not contain $X_{a_1} \to X_{a_2}$ (but may contain the other arcs of that cycle, that is, we do not prohibit the others).
- The second case certainly contains $X_{a_1} \to X_{a_2}$, but $X_{a_2} \to X_{a_3}$ is prohibited (so they are disjoint because of the difference in the presence of the first arc).
- (And so on such that) The y-th case certainly contains $X_{a_{y'}} \to X_{a_{y'+1}}$ for all $y' < y$ and prohibits $X_{a_y} \to X_{a_{y+1}}$. This is done until the last element of the cycle.

This is the same idea as the inclusion-exclusion principle, but applied here to the arcs of the cycle. It ensures that we never process the same graph twice, and also that we cover all the graphs, as by the union of the mentioned sets we obtain the original $\mathcal{H}$. Because of that, the algorithm runs at most $\prod_i |C(X_i)|$ steps, where $|C(X_i)|$ is the size of the cache for X_i (there are not more ways to combine parent sets than that number). In practice, we expect the *bound* step to be effective in dropping parts of the search space in order to reduce the total time cost.

The B&B algorithm as described alternately picks elements from the top and from the bottom of the queue (the percentage of elements from the bottom is controlled by the user parameter *bottom*). In terms of covering all search space, we have to ensure that all elements of the queue are processed, no matter the order we pick them, and that is enough to the correctness of the algorithm. However, there is an important difference between elements from the top and the bottom: top elements improve the upper bound for the global score, because we know that the global score is less than or equal to the highest score in the queue. Still, the elements from the top cannot improve the lower bound, as lower bounds are made of valid DAGs, and the first found DAG from the top is already the global optimal solution (by the priority of the queue). In order to update also the lower bound, elements from the bottom can be used, as they have low score with (usually) small subspaces, making easier to find valid DAGs. In fact, we know that an element from the bottom, if not a DAG, will generate new elements of the queue whose subspaces have upper bound score less than that of the originating elements, which certainly put them again in the bottom of the queue. This means that processing elements from the bottom is similar to perform a depth-first search, which is likely to find valid DAGs. Hence, we guarantee to have both lower and upper bounds converging to the optimal solution.

In the experiments of Section 6, we have chosen the parameter *bottom* such that one in three iterations picks an element from the bottom of the queue. This choice has not been tuned and has been taken with the aim of increasing the chance of finding valid DAGs. Note that every element

from the top will certainly decrease the upper bound, while the elements from the bottom may or may not increase the lower bound. There is no obvious choice here: if we use fewer elements from the bottom, then we improve the upper bound faster, but we possibly have a worse lower bound, which implies in less chance of *bounding* regions of the search space (which would help to improve the upper bound in a faster way as well); on the other hand, if we use many elements from the bottom, then we increase the chance (even if there is no guarantee) of improving the lower bound, but we spend less time improving the upper bound, which ultimately has to be tightened until it meets the lower bound. In other words, if the current best solution is already very good (in the sense of being optimal or almost optimal—note that we do not know it when the method is running), then it is useless to pick elements from the bottom. Therefore, a possible (heuristic) approach is to adaptively select the percentage of elements to pick from the bottom: in the very beginning of the algorithm, more elements are picked from the bottom. As time passes, as the upper bound gets closer to the best current solution (it also becomes less likely to find better solutions because the chance that the current solution is already good gets higher with time), so the percentage of elements picked from the bottom should keep reducing until it reaches zero (or almost zero). Currently we have not implemented any strategy to modify the percentage of elements that are picked from top and bottom of the queue.

Two other ideas are worth mentioning regarding the B&B algorithm: (i) if we periodically perform local searches within subspaces using distinct starting points, the lower bound can be improved (still this has its own computational cost, so it must be selectively done); (ii) if we do check for acyclicity in the step 4b before inserting the triple into the queue, then it is possible to update the current best solution earlier, and the algorithm still works. In this case, step 2 is unnecessary because DAGs will never be inserted into the queue (given that we check if the initial graph is not already a DAG before starting the main loop). Still, we need to find the cycle to be used in step 3, so to save computations we need to spend memory to store the cycle (previously found in step 4b) together with the triples of the queue. Hence, this idea trades some computational time (or memory usage) by a speed-up in finding some DAGs to improve the lower bound. Note that, in most cases, the graph that is checked in step 4b will not be a DAG anyway. While this modification benefits the improvement of the lower bound by spending some additional computation/memory, some preliminary experiments have not shown any significant gain. However, this is still to be better analyzed, as it may vary depending on implementation details.

The B&B can be stopped at any time and the current best solution as well as an upper bound for the global best score are available. This stopping criterion might be based on number of steps, time and/or memory consumption, percentage of error (difference between upper and lower bounds). This is an important property of this method. For example, if we are just looking for an improving solution, we may include in the loop an *if* to check if the current best solution is already better than some threshold, which would save computational time. Still, if we run it until the end, we are ensured to have a global optimum solution.

The algorithm can also be easily parallelized. We can split the content of the priority queue into many different tasks. No shared memory needs to exist among tasks if each one has its own version of the cache. The only data structure that needs consideration is the queue, which from time to time must be balanced between tasks. With a message-passing idea that avoids using *locks*, the gain of parallelization is linear in the number of tasks.

Some particular cases of the algorithm are worth mentioning. If we fix an ordering for the variables such that all the arcs must link a node towards another non-precedent in the ordering (this

is a common idea in many approximate methods), the proposed algorithm does not perform any branch, as the ordering implies acyclicity, and so the initial solution is already the best (only for that ordering—recall that the number of possible orderings is exponential in n). The performance would be proportional to the time to create the cache. Another important case is when one limits the maximum number of parents of a node. This is relevant for hard problems with many variables, as it would imply in a bound on the cache size.

	ESS	adult	breast	car	letter	lung	mush	nurse	wdbc	zoo
	0.1	6.2	0.0	0.1	3.7	1699.6	7.5	0.9	221.2	0.4
Memory	1	6.2	0.0	0.1	3.7	1150.1	5.9	0.8	204.6	0.4
(in MB)	10	6.3	0.0	0.1	3.8	812.3	5.4	0.7	206.2	0.3
	BIC	1.8	0.0	0.0	2.3	0.3	0.5	0.4	5.3	0.1
	0.1	89.3	0.0	0.0	429.4	2056	357.9	0.7	2891	1.7
Time	1	91.6	0.0	0.0	440.4	1398	278.7	0.7	2692	1.7
(in sec.)	10	91.6	0.0	0.0	438.1	1098	268.9	0.7	2763	1.7
	BIC	67.4	0.0	0.1	859.6	1.3	72.1	1.4	351	0.3
	0.1	$2^{17.4}$	$2^{10.5}$	$2^{8.8}$	$2^{20.1}$	$2^{30.8}$	$2^{24.0}$	$2^{11.2}$	$2^{27.9}$	$2^{19.8}$
Number	1	$2^{17.4}$	$2^{10.5}$	$2^{8.8}$	$2^{20.1}$	$2^{30.2}$	$2^{23.6}$	$2^{11.2}$	$2^{27.8}$	$2^{19.7}$
of Steps	10	$2^{17.4}$	$2^{10.4}$	$2^{8.8}$	$2^{20.1}$	$2^{29.8}$	$2^{23.5}$	$2^{11.2}$	$2^{27.9}$	$2^{19.6}$
	BIC	$2^{14.8}$	$2^{7.3}$	$2^{8.4}$	$2^{19.0}$	$2^{15.4}$	$2^{17.1}$	$2^{10.9}$	$2^{20.7}$	$2^{13.1}$
Worst-case		$2^{17.9}$	$2^{12.3}$	$2^{8.8}$	$2^{20.1}$	$2^{31.1}$	$2^{26.5}$	$2^{11.2}$	$2^{28.4}$	$2^{20.1}$

Table 1: Memory, time and number of steps (local score evaluations) used to build the cache. Results for BIC and BDeu with ESS varying from 0.1 to 10 are presented.

6. Experiments

We perform experiments to show the benefits of the reduced cache and search space. Later we show some examples of the use of constraints.[7] First, we use data sets available at the UCI repository (Asuncion and Newman, 2007). Lines with missing data are removed and continuous variables are discretized over the mean into binary variables. The data sets are: *adult* (15 variables and 30162 instances), *breast* (10 variables and 683 instances), *car* (7 variables and 1728 instances) *letter* (17 variables and 20000 instances), *lung* (57 variables and 27 instances), mushroom (23 variables and 1868 instances, denoted by *mush*), nursery (9 variables and 12960 instances, denoted by *nurse*), Wisconsin Diagnostic Breast Cancer (31 variables and 569 instances, denoted by *wdbc*), *zoo* (17 variables and 101 instances). The number of categories per variables varies from 2 to dozens in some cases (we refer to UCI for further details).

Table 1 presents the used memory in MB (first block), the time in seconds (second block) and number of steps in local score evaluations (third block) for the cache construction, using the properties of Section 4. Each column presents the results for a distinct data set. In different lines we show results for BDeu with ESS equals to 0.1, 1, 10, and for BIC. The line *worst-case* presents the number of steps to build the cache without using Theorems 4 (for BIC/AIC) and 9 (for BDeu), which are the theorems that allow the algorithm to avoid computing every subset of parents. As we see through the log-scale in which they are presented, the reduction in number of steps has not been

7. The software is available online in the web address `http://www.ecse.rpi.edu/~cvrl/structlearning.html`.

exponential, but still saves a good amount of computations (roughly half of the work). In the case of the BIC score, the reduction is more significant. In terms of memory, the usage clearly increases with the number of variables in the network (lung has 57 and wdbc has 31 variables).

	ESS	adult	breast	car	letter	lung	mush	nurse	wdbc	zoo
Max.	0.1	2.1(4)	1.0(1)	0.7(1)	4.5(5)	0.1(2)	4.1(5)	1.2(3)	1.3(2)	1.4(3)
Number	1	2.4(4)	1.0(1)	1.0(2)	5.2(6)	0.4(2)	4.4(7)	1.7(3)	1.7(3)	1.9(4)
of Parents	10	3.3(5)	1.0(1)	1.9(2)	5.9(6)	3.0(4)	4.8(8)	2.1(3)	3.1(4)	3.4(4)
	BIC	2.8(5)	1.0(1)	1.3(2)	6.3(7)	2.1(3)	4.1(4)	1.8(3)	2.7(3)	2.8(3)
Worst-case		14.0	9.0	6.0	16.0	6.0*	22.0	8.0	8.0*	16.0
Final Size	0.1	$2^{4.2}$	$2^{1.5}$	$2^{1.1}$	$2^{8.2}$	$2^{0.2}$	$2^{8.5}$	$2^{1.9}$	$2^{3.6}$	$2^{3.3}$
of the	1	$2^{4.8}$	$2^{1.9}$	$2^{1.6}$	$2^{9.0}$	$2^{0.8}$	$2^{8.9}$	$2^{2.4}$	$2^{4.9}$	$2^{4.4}$
Cache	10	$2^{6.3}$	$2^{3.3}$	$2^{3.0}$	$2^{10.5}$	$2^{10.7}$	$2^{9.8}$	$2^{3.5}$	$2^{12.1}$	$2^{8.9}$
	BIC	$2^{9.3}$	$2^{4.7}$	$2^{4.5}$	$2^{15.3}$	$2^{11.5}$	$2^{13.0}$	$2^{5.6}$	$2^{12.9}$	$2^{10.9}$
Worst-case		$2^{17.9}$	$2^{12.3}$	$2^{8.8}$	$2^{20.1}$	$2^{31.1*}$	$2^{26.5}$	$2^{11.2}$	$2^{28.4*}$	$2^{20.1}$
Implied	0.1	$2^{54.1}$	$2^{13.3}$	$2^{6.3}$	$2^{129.0}$	$2^{8.2}$	$2^{175.7}$	$2^{11.6}$	$2^{90.3}$	$2^{39.3}$
Search	1	$2^{62.1}$	$2^{17.1}$	$2^{8.3}$	$2^{144.8}$	$2^{33.1}$	$2^{186.0}$	$2^{15.4}$	$2^{132.7}$	$2^{60.3}$
Space	10	$2^{91.6}$	$2^{33.2}$	$2^{20.6}$	$2^{176.1}$	$2^{612.0}$	$2^{221.8}$	$2^{27.3}$	$2^{375.1}$	$2^{150.7}$
(approx.)	BIC	2^{71}	2^{23}	2^{10}	2^{188}	2^{330}	2^{180}	2^{17}	2^{216}	2^{111}
Worst-case		2^{210}	2^{90}	2^{42}	2^{272}	2^{1441*}	2^{506}	2^{72}	2^{727*}	2^{272}

Table 2: Final cache characteristics: maximum number of parents (average by node; between parenthesis is presented the actual maximum number), actual cache size, and (approximate) search space implied by the cache. Worst-cases are presented for comparison (those marked with a star are computed using the constraint on the number of parents that was applied to *lung* and *wdbc*). Results of BIC and BDeu with ESS from 0.1 to 10 are presented.

The benefits of the application of these results imply in performance gain for many algorithms in the literature to learn Bayesian network structures, as long as they only need to work over the (already precomputed) small cache. In Table 2 we present the final cache characteristics, where we find the most attractive results, for instance, the small cache sizes when compared to the worst case. The first block contains the maximum number of parents per node (averaged over the nodes, and the actual maximum between parenthesis). The worst-case is the total number of nodes in the data set minus one, apart from *lung* (where we have set a limit of at most six parents) and *wdbc* (with at most eight parents). The second block shows the cache size for each data set and distinct values of ESS. We also show the results of the BIC score and the worst-case values for comparison. We see that the actual cache size is smaller (in orders of magnitude) than the worst-case situation. It is also possible to analyze the search space reduction implied by these results by looking the implications to the search space of structure learning. We must point out that by search space we mean all the possible combinations of parent sets for all the nodes. Eventually some of these combinations are not DAGs, but are still being counted. However, there are two considerations: (i) the precise counting problem is harder to solve (in order to give the exact search space size), and (ii) many structure learning algorithms run over more than only DAGs, because they need to look at the graphs (and thus combinations of parents) to decide if they are acyclic or not. In these cases, the actual search space is not simply the set of possible DAGs, even though the final solution will be a DAG. Still, some algorithms might do a better job by using other ideas of searching for

the best structure instead of looking to possible DAGs, which might imply in a smaller worst-case complexity (for instance, the dynamic programming method runs over subsets of variables, which are in number 2^n).

		B&B			DP		OS		HC	
	network	Score	gap	time	score	time	score	time	score	time
BIC	adult	-286902.8	5.5%	150.3	0.0%	0.77	0.1%	0.17	0.5%	0.30
	breast	-8254.8	0.0%	0.01	0.0%	0.01	0.0%	0.01	0.0%	0.00
	car	-13100.5	0.0%	0.01	0.0%	0.01	0.0%	0.01	0.2%	0.00
	letter	-173716.2	8.1%	574.1	-0.6%	22.8	1.0%	0.75	3.7%	0.30
	lung	-1146.9	2.5%	907.1	*Fail*	*Fail*	1.0%	0.13	0.7%	0.05
	mushroom	-12834.9	15.3%	239.8	*Fail*	*Fail*	1.0%	0.12	4.8%	0.05
	nursery	-126283.2	0.0%	0.04	0.0%	0.04	0.0%	0.04	0.03%	0.06
	wdbc	-3053.1	13.6%	333.5	*Fail*	*Fail*	0.8%	0.13	0.9%	0.02
	zoo	-773.4	0.0%	5.2	0.0%	3.5	1.0%	0.03	0.6%	0.00
ESS=0.1	adult	-288591.2	0.0%	92.1	0.0%	0.75	0.1%	0.21	0.3%	0.32
	breast	-8635.1	0.0%	0.02	0.0%	0.01	0.0%	0.01	0.0%	0.00
	car	-13295.0	0.0%	0.01	0.0%	0.00	0.0%	0.00	0.1%	0.01
	letter	-181941.5	5.7%	375.75	-0.1%	7.6	0.1%	0.27	2.1%	0.27
	lung	-1731.9	0.0%	0.22	*Fail*	*Fail*	0.0%	0.11	0.0%	0.05
	mushroom	-12564.2	14.7%	382.4	*Fail*	*Fail*	0.2%	0.15	5.3%	0.05
	nursery	-126660.4	0.0%	0.06	0.0%	0.04	0.0%	0.04	0.1%	0.06
	wdbc	-3558.6	4.4%	494.1	*Fail*	*Fail*	1.4%	0.05	1.3%	0.01
	zoo	-1024.5	0.0%	0.09	0.0%	3.1	0.8%	0.01	1.0%	0.00
ESS=1	adult	-286695.2	4.5%	203.0	0.0%	0.76	0.1%	0.22	0.3%	0.34
	breast	-8254.3	0.0%	0.02	0.0%	0.01	0.0%	0.01	0.0%	0.00
	car	-13145.3	0.0%	0.01	0.0%	0.00	0.0%	0.00	0.05%	0.00
	letter	-178635.2	6.7%	520.2	-0.7%	9.9	0.0%	0.34	2.1%	0.27
	lung	-1249.7	0.0%	0.61	*Fail*	*Fail*	0.1%	0.12	0.1%	0.05
	mushroom	-12097.0	16.7%	381.5	*Fail*	*Fail*	0.2%	0.19	4.2%	0.05
	nursery	-126212.7	0.0%	0.06	0.0%	0.04	0.0%	0.04	0.1%	0.05
	wdbc	-3175.9	11.2%	471.1	*Fail*	*Fail*	0.7%	0.06	1.0%	0.02
	zoo	-794.1	0.0%	1.4	0.0%	3.4	1.1%	0.02	8.7%	0.00
ESS=10	adult	-285014.5	11.8%	213.8	-0.1%	0.88	0.04%	0.24	0.5%	0.33
	breast	-8130.2	0.0%	0.04	0.0%	0.01	0.0%	0.00	0.3%	0.00
	car	-13038.6	0.0%	0.03	0.0%	0.00	0.0%	0.00	0.03%	0.00
	letter	-174111.8	8.7%	1250	-0.4%	22.3	0.1%	0.84	1.8%	0.32
	lung	-957.2	11.7%	2118	*Fail*	*Fail*	3.3%	1.38	2.3%	0.1
	mushroom	-11924.0	22.7%	587.8	*Fail*	*Fail*	0.1%	0.43	2.4%	0.07
	nursery	-125846.5	0.0%	0.14	0.0%	0.04	0.0%	0.04	0.1%	0.06
	wdbc	-2986.2	22.2%	1938	*Fail*	*Fail*	0.6%	2.8	1.4%	0.23
	zoo	-697.2	13.2%	367.7	-0.3%	5.0	1.4%	0.1	0.9%	0.00

Table 3: Comparison of scores among B&B, DP, OS and HC. *Fail* means that it could not solve the problem within 10 million steps or because of memory limit (4GB). DP, OS and HC scores are in percentage w.r.t. the score of B&B (positive means worse than B&B and negative means better). Each entry with a 0.0% means that the result, in that instance, was exactly equal to the B&B result (in terms of the score). Times are given in seconds.

An expected but important point to emphasize is the correlation of the prior with the time and memory to build the cache. It would be expected that, as larger ESS (and thus the prior towards the uniform) as slower and more memory consuming is the method. That is because smoothing the different parent sets by the stronger prior makes harder to see large differences in scores, and consequently the properties that would reduce the cache size are less effective. However, this is not quite evident from the results, where the relation between ESS and time/memory is not clear. Yet it must be noted that the two largest data sets in terms of number of variables (*lung* and *wdbc*) were impossible to be processed without setting up other limits such as maximum number of parents or maximum number of free parameters in the node (we have not used any limit for the other data sets). We used an upper limit of six parents per node for *lung* and eight for *wdbc*. This situation deserves further study so as to clarify whether it is possible to run these computations on large data sets and large ESS. It might be necessary to find tighter bounds if at all possible, that is, stronger results than Theorem 9 to discard unnecessary score evaluations earlier in the computations. Nevertheless, the main goal of this present work is not to study the impact of ESS on learning, but to present properties that improve the performance of learning methods.

In Table 3, we show results of four distinct algorithms: the B&B described in Section 5, the dynamic programming (DP) idea of Silander and Myllymaki (2006), the hill-climbing (HC) method starting with an empty structure, and an algorithm that picks variable orderings randomly and then find the best structure satisfying that ordering, that is, DAGs where arcs respect the ordering of the variables (there is no arc connecting a node to its predecessors in the ordering). This algorithm (named OS) is similar to K2 algorithm with random orderings, but it is always better because a global optimum is found for each ordering (we use one million of orderings). Note that OS performs better than HC in almost all test cases. We have chosen to analyze the BIC scores (given that the properties have provided greater reduction in the search space in this case) and BDeu with ESS equals to 0.1, 1 and 10. It is clear from the results of ESS equals to 10 that the B&B procedure struggles with very large search spaces, and the same might happen for even larger ESS.

The scores obtained by each algorithm (in percentage against the value obtained by B&B) and the corresponding time are shown in Table 3 (excluding the cache construction). A limit of ten million steps is given to each method (steps here are considered as the number of queries to the cache). It is also presented the reduced space where B&B performs its search, as well as the maximum gap of the solution. This gap is obtained by the relaxed version of the problem. We can guarantee that the global optimal solution is within this gap (even though the solution found by the B&B may already be the best, as it happens, for example, in the first line of the table). With the reduced cache presented here, finding the best structure for a given ordering is very fast, so it is possible to run OS over millions of orderings in a short period of time. Some additional comments are worth. DP could not solve *wdbc* or *lung* even without the limit in number of steps, because it has exhausted 16GB of memory. Hence, we cannot expect to obtain answers in larger cases. However, it is clear that (in a worst case sense) the number of steps of DP is smaller than that of B&B, and this behavior can be seen in data sets with small number of variables. Nevertheless, B&B eventually bounds some regions without processing them, provides an upper bound at each iteration, and does not suffer from memory exhaustion as DP. It is true that B&B also uses memory increasingly if there are not good bounds, but this case can be tackled by (automatically) switching between the described B&B and a full depth-first search.[8] This makes the method applicable even to very large settings. When

8. Our implementation is able to stop the B&B and to switch to a full depth-first search, but this behavior was not necessary in the experiments because the memory requirements were not too intense.

n is large (more than 35), DP will not finish in reasonable time, and hence will not provide any solution, while B&B still gives an approximation and a bound to the global optimum. About OS, if we sample even more orderings, then its results improve and the global optimum is found also for the *adult* data set. Still, OS provides no guarantee or estimation about how far is the global optimum (here we know that the optimum has been achieved because of the solution of the exact methods). It is worth noting that both DP and OS are also benefited by the smaller cache. Although we are discussing only four algorithms, performance gain from the application of the properties in other algorithms is expected as well.

network	time(s)	cache size	space
adult	0.26	114	2^{39}
car	0.01	14	$2^{6.2}$
letter	0.32	233	2^{61}
lung	0.26	136	2^{51}
mushroom	0.71	398	2^{88}
nursery	0.06	26	2^{12}
wdbc	361.64	361	2^{99}
zoo	8.4	1697	2^{111}

Table 4: B&B procedure learning TANs using BIC. Time (in seconds) to find the global optimum, cache size (number of stored scores) and (reduced) space for the search.

The last part of this section is dedicated to some test cases with constraints. Table 4 shows the results when we employ constraints to force the final network to be a Tree-augmented Naive Bayes. Here the class variable is isolated in the data set and constraints are included as described in Section 3. Note that the cache size, the search space and consequently the time to solve the problems have substantially decreased. Finally, Table 5 has results for random data sets with predefined number of nodes and instances using the BIC score. A randomly created Bayesian network with at most $3n$ arcs (where n is the number of nodes) is used to sample the data. Because of that, we are able to generate random structural constraints that are certainly valid for this *true* Bayesian network (approximately n constraints for each case). The table contains the total time to run the problem and the size of the cache, together with the results when using constraints. Note that the code was run in parallel with a number of tasks equals to n, otherwise an increase by a factor of n must be applied to the results in the table. Each line contains the mean and standard deviation of ten executions (using random generated networks) for time and cache size with and without constraints (using the same data sets in order to compare them). We can see that the gain is recurrent in all cases. The B&B method was able to find a global optimal solution in all but the cases with one hundred nodes, where it has achieved an approximate solution with error always less than 0.1% (this amounts to 40% of the test cases with 100 nodes). We point out that the other exact method we have analyzed based on dynamic programming cannot deal with such large networks because of both memory and time costs. There is an increase in computational time from 30 to 100 nodes, but even more from 100 to 500 instances (considering the data sets with 70 and 100 nodes). This happens because the properties that reduce the cache size and search space are much more effective under small-sized data sets. However, we are not considering the improvement in accuracy when using constraints, but just the computational gain. It is not trivial to measure the quality of a learned structure, because the target of the methods is the underlying probability distribution, and distinct structures may lead

to good results in fitting such distribution. For instance, comparing number of matching arcs has only meaning if one is interested in the structure by itself, and not in the fitness of the underlying distribution. This topic deserves attention, but it would bring us far from the focus of this study.

	unconstrained				constrained			
nodes(n)/	time(sec)		cache size		time(sec)		cache size	
instances	mean	std.dev.	mean	std.dev.	mean	std.dev.	mean	std.dev.
30/100	0.07	0.02	49.6	9.1	0.04	0.01	44.3	8.98
30/500	3.70	1.18	75.6	16.6	2.33	0.73	61.4	17.7
50/100	0.31	0.08	77.9	9.6	0.20	0.04	66.1	6.71
50/500	37.1	10.8	102.5	23.0	23.2	6.86	83.0	17.7
70/100	1.91	0.82	127.5	18.1	0.97	0.32	108.3	13.6
70/500	293.3	99.5	137.3	22.2	176.3	62.6	111.8	14.5
100/100	85.0	29.3	253.4	27.7	4.44	1.06	199.5	21.1
100/500	2205.6	534.4	204.6	32.1	1414.8	419.2	168.0	21.3

Table 5: Results on ten data sets per line generated from random networks. Both mean and standard deviation of time to solve (with an upper limit of 20 million steps) and size of the cache (in number of scores) are presented for the *normal* unconstrained case and for the constrained cases (over the same data sets).

7. Conclusions

This paper describes novel properties of decomposable score functions to learn Bayesian network structure from data. Such properties allow the construction of a cache with all possible local scores of nodes and their parents without large memory consumption, which can later be used by searching algorithms. For instance, memory consumption was a bottleneck for some algorithms in the literature, see, for example, Parviainen and Koivisto (2009). This implies in a considerable reduction of the search space of graphs without losing the global optimal structure, that is, it is ensured that the overall best graph remains in the reduced space. In fact the reduced memory and search space potentially benefits many structure learning methods in the literature, and we have conducted experiments with some of them.

An algorithm based on a branch-and-bound technique is described, which integrates structural constraints with data. The procedure guarantees global optimality with respect the score function. It is an anytime procedure in the sense that the error of the current solution is constantly reduced either by finding a better solution or by reducing the upper bound for the global optimum. If stopped early, the method provides the current solution and its maximum error. This can be useful if one wants to integrate it with an expectation-maximization (EM) method to treat incomplete data sets, and such characteristic is usually not present in other exact structure learning methods. In the EM method, the global structure learning procedure ensures that the maximization step is never trapped by a local solution, and the anytime property allows the use of a generalized EM to reduce considerably the computational cost.

Because of the properties and the characteristics of the B&B method, it is more efficient than dynamic programming state-of-the-art exact methods for large domains. We show through experiments with randomly generated data and public data sets that problems with up to 70 nodes can

be exactly processed in reasonable time, and problems with 100 nodes are handled within a small worst-case error. Dynamic programming methods are able to treat less than 35 variables. Described ideas may also help to improve other approximate methods and may have interesting practical applications. We show through experiments with public data sets that requirements of memory are small, as well as the resulting reduced search space. Of course we do not expect to exactly solve problems for considerably large networks, still the paper makes a relevant step towards solving larger instances. We can summarize the comparison with the dynamic programming idea as follows: if the problem has few variables, dynamic programming is probably the fastest method (the branch-and-bound method will also be reasonably fast); if the problem has medium size, the branch-and-bound method might solve it exactly (dynamic programming will mostly fail to answer); finally, if the problem is large, the branch-and-bound method will eventually give an approximation (and its worst-case error), while the standard dynamic programming idea will fail.

There is certainly much further to be done. One important question is whether the bounds of the theorems in Section 4 (more specifically Theorem 9) can be improved or not. We are actively working on this question. Furthermore, the experimental analysis can be extended to further clarify the understanding of the problem, for instance how the ESS affects the results. It is clear that, for considerably large domains, none of the exact methods are going to suffice by themselves. Besides developing ideas and algorithms for dealing with large domains, the comparison of structures and what define them to be good is an important topic. For example, accuracy of the generated networks can be evaluated with real data. On the other hand, it does not ensure that we are finding the true links of the underlying structure, but a somehow similar graph that produces a close joint distribution. For that, one could use generated data and compare the structures against the one data were generated from it. A study on how the properties may help fast approximate methods is also a desired goal.

Acknowledgments

This work is supported in part by the grant W911NF-06-1-0331 from the U.S. Army Research Office, and by the *Computational Life Sciences (CLS)* program phase II, canton Ticino, Switzerland.

References

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton Univ. Press, 2006.

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

R. Bouckaert. Properties of Bayesian belief network learning algorithms. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, UAI'94, pages 102–109, San Francisco, CA, 1994. Morgan Kaufmann.

W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the 8th Annual Conference on Uncertainty in Artificial Intelligence*, UAI'92, pages 52–60, San Francisco, CA, 1991. Morgan Kaufmann.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

D. M. Chickering, C. Meek, and D. Heckerman. Large-sample learning of Bayesian networks is np-hard. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, UAI'03, pages 124–13, San Francisco, CA, 2003. Morgan Kaufmann.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

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'09, pages 113–120, Montreal, 2009. Omnipress.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

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'10, pages 358–365, 2010.

M. Koivisto. Advances in exact Bayesian structure discovery in Bayesian networks. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, UAI'06, pages 241–248. AUAI Press, 2006.

M. Koivisto and K. Sood. Exact Bayesian Structure Discovery in Bayesian Networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

K. Kojima, E. Perrier, S. Imoto, and S. Miyano. Optimal search on clustered structural constraint for learning Bayesian network structure. *Journal of Machine Learning Research*, 11:285–310, 2010.

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'09, pages 436–443, Arlington, Virginia, United States, 2009. AUAI Press.

F. Pernkopf and J. Bilmes. Discriminative versus generative parameter and structure learning of Bayesian network classifiers. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML'05, pages 657–664, New York, NY, USA, 2005. ACM.

E. Perrier, S. Imoto, and S. Miyano. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research*, 9:2251–2286, 2008.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

T. Silander and P. Myllymaki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, UAI'06, pages 445–452, Arlington, Virginia, 2006. AUAI Press.

A. P. Singh and A. W. Moore. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University, 2005. CMU-CALD-05-106.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search*. Springer-Verlag, New York, 1993.

J. Suzuki. Learning Bayesian belief networks based on the minimum description length principle: An efficient algorithm using the B&B technique. In *Proceedings of the 13th International Conference on Machine Learning*, ICML'96, pages 462–470, 1996.

M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, UAI'05, pages 584–590, 2005.

I. Tsamardinos, L. E. Brown, and C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

Journal of Machine Learning Research 12 (2011) 691-730 Submitted 2/10; Revised 11/10; Published 3/11

Inverse Reinforcement Learning in Partially Observable Environments

Jaedeug Choi JDCHOI@AI.KAIST.AC.KR
Kee-Eung Kim KEKIM@CS.KAIST.AC.KR
Department of Computer Science
Korea Advanced Institute of Science and Technology
Daejeon 305-701, Korea

Editor: Shie Mannor

Abstract

Inverse reinforcement learning (IRL) is the problem of recovering the underlying reward function from the behavior of an expert. Most of the existing IRL algorithms assume that the environment is modeled as a Markov decision process (MDP), although it is desirable to handle partially observable settings in order to handle more realistic scenarios. In this paper, we present IRL algorithms for partially observable environments that can be modeled as a partially observable Markov decision process (POMDP). We deal with two cases according to the representation of the given expert's behavior, namely the case in which the expert's policy is explicitly given, and the case in which the expert's trajectories are available instead. The IRL in POMDPs poses a greater challenge than in MDPs since it is not only ill-posed due to the nature of IRL, but also computationally intractable due to the hardness in solving POMDPs. To overcome these obstacles, we present algorithms that exploit some of the classical results from the POMDP literature. Experimental results on several benchmark POMDP domains show that our work is useful for partially observable settings.

Keywords: inverse reinforcement learning, partially observable Markov decision process, inverse optimization, linear programming, quadratically constrained programming

1. Introduction

Inverse reinforcement learning (IRL) was first proposed by Russell (1998) as follows:

Given (1) measurements of an agent's behavior over time, in a variety of circumstances, (2) measurements of the sensory inputs to the agent, (3) a model of the physical environment (including the agent's body).

Determine the reward function that the agent is optimizing.

The significance of IRL has emerged from the connection between reinforcement learning (RL) and other research areas such as neurophysiology (Montague and Berns, 2002; Cohen and Ranganath, 2007), behavioral neuroscience (Lee et al., 2004; Niv, 2009) and economics (Erev and Roth, 1998; Borgers and Sarin, 2000; Hopkins, 2007). In these research areas, the reward function is generally assumed to be fixed and known, but it is often non-trivial to come up with an appropriate reward function for each problem. Hence, a progress in IRL can have a significant impact on many research areas.

©2011 Jaedeug Choi and Kee-Eung Kim.

IRL is a natural way to examine animal and human behaviors. If the decision maker is assumed to follow the principle of rationality (Newell, 1982), its behavior could be understood by the reward function that the decision maker internally optimizes. In addition, we can exploit the computed reward function to generate an agent that imitates the decision maker's behavior. This will be a useful approach to build an intelligent agent. Another advantage of IRL is that the solution of IRL problems, that is, the reward function, is one of the most transferable representations of the agent's behavior. Although it is not easy to transfer the control policy of the agent to other problems that have a similar structure with the original problem, the reward function could be applied since it compactly represents the agent's objectives and preferences.

In the last decade, a number of studies on IRL have been reported. However, most of the previous IRL algorithms (Ng and Russell, 2000; Abbeel and Ng, 2004; Ramachandran and Amir, 2007; Neu and Szepesvari, 2007; Syed and Schapire, 2008; Ziebart et al., 2008) assume that the agent acts in an environment that can be modeled as a Markov decision process (MDP). Although the MDP assumption provides a good starting point for developing IRL algorithms, the implication is that the agent has access to the true global state of the environment. The assumption of an omniscient agent is often too strong in practice. Even though the agent is assumed to be an expert in the given environment, the agent may be (and often is) making optimal behaviors with a limited sensory capability. Hence, to relax the strong assumption and widen the applicability of IRL to more realistic scenarios, the IRL algorithms should be extended to partially observable environments, which can be modeled as partially observable Markov decision processes.

A partially observable Markov decision process (POMDP) (Sondik, 1971; Monahan, 1982; Kaelbling et al., 1998) is a general mathematical framework for single-agent planning under uncertainty about the effect of actions and the true state of the environment. Recently, many approximate techniques have been developed to compute an optimal control policy for large POMDPs. Thus, POMDPs have increasingly received a significant amount of attention in diverse research areas such as robot navigation (Spaan and Vlassis, 2004; Smith, 2007), dialogue management (Williams and Young, 2007), assisted daily living (Hoey et al., 2007), cognitive radio (Zhao et al., 2007) and network intrusion detection (Lane and Brodley, 2003). However, in order to address real-world problems using POMDPs, first, a model of the environment and the reward function should be obtained. The parameters for the model of an environment, such as transition probabilities and observation probabilities, can be computed relatively easily by counting the events if the true state can be accessed, but determining the reward function is non-trivial. In practice, the reward function is repeatedly hand-tuned by domain experts until a satisfactory policy is acquired. This usually entails a labor intensive process. For example, when developing a spoken dialogue management system, POMDP is a popular framework for computing the dialogue strategy, since we can compute an optimal POMDP policy that is robust to speech recognition error and maintains multiple hypotheses of the user's intention (Williams and Young, 2007). In this domain, transition probabilities and observation probabilities can be calculated from the dialogue corpus collected from a wizard-of-oz study. However, there is no straightforward way to compute the reward function, which should represent the balance among the reward of a successful dialogue, the penalty of an unsuccessful dialogue, and the cost of information gathering. It is manually adjusted until a satisfying dialogue policy is obtained. Therefore, a systematic method is desired to determine the reward function.

In this paper, we describe IRL algorithms for partially observable environments extending our previous results in Choi and Kim (2009). Specifically, we assume that the environment is modeled as a POMDP and try to compute the reward function given that the agent follows an optimal policy.

The algorithm is mainly motivated by the classical IRL algorithm by Ng and Russell (2000) and we adapt the algorithm to be robust for large problems by using the methods suggested by Abbeel and Ng (2004). We believe that some of the more recently proposed IRL algorithms (Ramachandran and Amir, 2007; Neu and Szepesvari, 2007; Syed and Schapire, 2008; Ziebart et al., 2008) also could be extended to handle partially observable environments. The aim of this paper is to present a general framework for dealing with partially observable environments, the computational challenges involved in doing so, and some approximation techniques for coping with the challenges. Also, we believe that our work will prove useful for many problems that could be modeled as POMDPs.

The remainder of the paper is structured as follows: Section 2 reviews some definitions and notations of MDP and POMDP. Section 3 presents an overview of the IRL algorithms by Ng and Russell (2000) and Abbeel and Ng (2004). Section 4 gives a formal definition of IRL for partially observable environments, and discusses the fundamental difficulties of IRL and the barriers of extending IRL to partially observable environments. In Section 5, we focus on the problem of IRL with the explicitly given expert's policy. We present the optimality conditions of the reward function and the optimization problems with the computational challenges and some approximation techniques. Section 6 deals with more practical cases where the trajectories of the expert's actions and observations are given. We present algorithms that iteratively find the reward function, comparing the expert's policy and other policies found by the algorithm. Section 7 shows the experimental results of our algorithms in several POMDP domains. Section 8 briefly reviews related work on IRL. Finally, Section 9 discusses some directions for future work.

2. Preliminaries

Before we present the IRL algorithms, we briefly review some definitions and notations of MDP and POMDP to formally describe the completely observable environment and the partially observable environment.

2.1 Markov Decision Process

A Markov decision process (MDP) provides a mathematical framework for modeling a sequential decision making problem under uncertainty about the effect of an agent's action in an environment where the current state depends only on the previous state and action, namely, the Markov property. An MDP is defined as a tuple $\langle S, A, T, R, \gamma \rangle$:

- S is the finite set of states.
- A is the finite set of actions.
- $T : S \times A \to \Pi(S)$ is the state transition function, where $T(s,a,s')$ denotes the probability $P(s'|s,a)$ of reaching state s' from state s by taking action a.
- $R : S \times A \to \mathbb{R}$ is the reward function, where $R(s,a)$ denotes the immediate reward of executing action a in state s, whose absolute value is bounded by R_{max}.
- $\gamma \in [0,1)$ is the discount factor.

A policy in MDP is defined as a mapping $\pi : S \to A$, where $\pi(s) = a$ denotes that action a is always executed in state s following the policy π. The value function of policy π at state s is the

expected discounted return of starting in state s and executing the policy. The value function can be computed as:

$$V^{\pi}(s) = R(s,\pi(s)) + \gamma \sum_{s' \in S} T(s,\pi(s),s') V^{\pi}(s'). \tag{1}$$

Given an MDP, the agent's objective is to find an optimal policy π^* that maximizes the value for all the states, which should satisfy the Bellman optimality equation:

$$V^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^*(s') \right].$$

It is often useful to express the above equation in terms of Q-function: π is an optimal policy if and only if

$$\pi(s) \in \operatorname*{argmax}_{a \in A} Q^{\pi}(s,a),$$

where

$$Q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^{\pi}(s'), \tag{2}$$

which is the expected discounted return of executing action a in state s and then following the policy π.

2.2 Partially Observable Markov Decision Process

A partially observable Markov decision process (POMDP) is a general framework for modeling the sequential interaction between an agent and a partially observable environment where the agent cannot completely perceive the underlying state but must infer the state based on the given noisy observation. A POMDP is defined as a tuple $\langle S, A, Z, T, O, R, b_0, \gamma \rangle$:

- S, A, T, R and γ are defined in the same manner as in MDPs.
- Z is the finite set of observations.
- $O : S \times A \to \Pi(Z)$ is the observation function, where $O(s,a,z)$ denotes the probability $P(z|s,a)$ of perceiving observation z when taking action a and arriving in state s.
- b_0 is the initial state distribution, where $b_0(s)$ denotes the probability of starting in state s.

Since the true state is hidden, the agent has to act based on the history of executed actions and perceived observations. Denoting the set of all possible histories at the t-th time step as $H_t = (A \times Z)^t$, a policy in POMDP is defined as a mapping from histories to actions $\pi : H_t \to A$. However, since the number of possible histories grows exponentially with the number of time steps, many POMDP algorithms use the concept of *belief*. Formally, the belief b is the probability distribution over the current states, where $b(s)$ denotes the probability that the state is s at the current time step, and Δ denotes a $|S|-1$ dimensional *belief simplex*. The belief update for the next time step can be computed from the belief at the current time step: Given the action a at the current time step and the observation z at the next time step, the updated belief b_z^a for the next time step is obtained by

$$b_z^a(s') = P(s'|b,a,z) = \frac{O(s',a,z) \sum_s T(s,a,s') b(s)}{P(z|b,a)}, \tag{3}$$

where the normalizing factor $P(z|b,a) = \sum_{s'} O(s',a,z) \sum_s T(s,a,s')b(s)$. Hence, the belief serves as a sufficient statistic for fully summarizing histories, and the policy can be equivalently defined as a mapping $\pi : \Delta \to A$, where $\pi(b) = a$ specifies action a to be selected at the current belief b by the policy π. Using beliefs, we can view POMDPs as belief-state MDPs, and the value function of an optimal policy satisfies the Bellman equation:

$$V^*(b) = \max_a \Big[\sum_s b(s)R(s,a) + \gamma \sum_{s',z} T(s,a,s')O(s',a,z)V^*(b^a_z) \Big]. \tag{4}$$

Alternatively, a policy in POMDP can be represented as a finite state controller (FSC). An FSC policy is defined by a directed graph $\langle \mathcal{N}, \mathcal{E} \rangle$, where each node $n \in \mathcal{N}$ is associated with an action $a \in A$ and has an outgoing edge $e_z \in \mathcal{E}$ per observation $z \in Z$. The policy can be regarded as $\pi = \langle \psi, \eta \rangle$ where ψ is the *action strategy* associating each node n with an action $\psi(n) \in A$, and η is the *observation strategy* associating each node n and observation z with a successor node $\eta(n,z) \in \mathcal{N}$.

Given an FSC policy $\pi = \langle \psi, \eta \rangle$, the value function V^π is the expected discounted return of executing π and is defined over the joint space of FSC nodes and POMDP states. It can be computed by solving a system of linear equations:

$$V^\pi(n,s) = R(s,a) + \gamma \sum_{n',s'} T^{a,os}(\langle n,s \rangle, \langle n',s' \rangle) V^\pi(n',s'), \tag{5}$$

where

$$T^{a,os}(\langle n,s \rangle, \langle n',s' \rangle) = T(s,a,s') \sum_{\substack{z \in Z \text{ s.t.} \\ os(z)=n'}} O(s',a,z), \tag{6}$$

with $a = \psi(n)$ and $os(z) = \eta(n,z)$. The value at node n for belief b is calculated by

$$V^\pi(n,b) = \sum_s b(s) V^\pi(n,s), \tag{7}$$

and the starting node for the initial belief b_0 is chosen by $n_0 = \text{argmax}_n V^\pi(n,b_0)$. We can also define Q-function for an FSC policy π:

$$Q^\pi(\langle n,s \rangle, \langle a,os \rangle) = R(s,a) + \gamma \sum_{n',s'} T^{a,os}(\langle n,s \rangle, \langle n',s' \rangle) V^\pi(n',s'),$$

which is the expected discounted return of choosing action a at node n and moving to node $os(z)$ upon observation z, and then following policy π. Also, Q-function for node n at belief b is computed by

$$Q^\pi(\langle n,b \rangle, \langle a,os \rangle) = \sum_s b(s) Q^\pi(\langle n,s \rangle, \langle a,os \rangle).$$

With an FSC policy π, we can sort the reachable beliefs into nodes, such that B_n denotes the set of beliefs that are reachable from the initial belief b_0 and the starting node n_0 when the current node is n. Note that $|B_n| \geq 1$ for every node n.

3. IRL in Completely Observable Markovian Environments

The MDP framework provides a good starting point for developing IRL algorithms in completely observable Markovian environments and most of the previous IRL algorithms address the problems in the MDP framework. In this section, we overview the IRL algorithms proposed by Ng and Russell (2000) and Abbeel and Ng (2004) as background to our work.

The IRL problem in completely observable Markovian environments is denoted with IRL for MDP\R, which is formally stated as follows: Given an MDP\R $\langle S, A, T, \gamma \rangle$ and an expert's policy π_E, find the reward function R that makes π_E an optimal policy for the given MDP. The problem can be categorized into two cases: The first case is when an expert's policy is explicitly given and the second case is when an expert's policy is implicitly given by its trajectories.

3.1 IRL for MDP\R from Policies

Let us assume that an expert's policy π_E is explicitly given. Ng and Russell (2000) present a necessary and sufficient condition for the reward function R of an MDP to guarantee the optimality of π_E:

$$Q^{\pi_E}(s, \pi_E(s)) \geq Q^{\pi_E}(s, a), \quad \forall s \in S, \forall a \in A, \tag{8}$$

which states that deviating from the expert's policy should not yield a higher value. From the condition, they suggest the following:

Theorem 1 [Ng and Russell, 2000] *Let an MDP\R $\langle S, A, T, \gamma \rangle$ be given. Then the policy π is optimal if and only if the reward function R satisfies*

$$R^{\pi} - R^{a} + \gamma (T^{\pi} - T^{a})(I - \gamma T^{\pi})^{-1} R^{\pi} \succeq 0, \quad \forall a \in A, \tag{9}$$

where the matrix notations and the matrix operator are defined as follows:

- *T^{π} is a $|S| \times |S|$ matrix with (s, s') element being $T(s, \pi(s), s')$.*
- *T^{a} is a $|S| \times |S|$ matrix with (s, s') element being $T(s, a, s')$.*
- *R^{π} is a $|S|$ vector with s-th element being $R(s, \pi(s))$.*
- *R^{a} is a $|S|$ vector with s-th element being $R(s, a)$.*
- *V^{π} is a $|S|$ vector with s-th element being $V^{\pi}(s)$.*
- *$X \succeq Y \Leftrightarrow X(i) \geq Y(i)$, for all i, if the length of X is the same as that of Y.*

Proof Equation (1) can be rewritten as $V^{\pi} = R^{\pi} + \gamma T^{\pi} V^{\pi}$. Thus,

$$V^{\pi} = (I - \gamma T^{\pi})^{-1} R^{\pi}. \tag{10}$$

By the definition of an optimal policy and Equation (2), π is optimal if and only if

$$\begin{aligned}
\pi(s) &\in \operatorname*{argmax}_{a \in A} Q^{\pi}(s, a), \quad \forall s \in S \\
&= \operatorname*{argmax}_{a \in A} (R(s, a) + \gamma \sum_{s'} T(s, a, s') V^{\pi}(s')), \quad \forall s \in S \\
\Leftrightarrow R(s, \pi(s)) &+ \gamma \sum_{s'} T(s, \pi(s), s') V^{\pi}(s') \\
&\geq R(s, a) + \gamma \sum_{s'} T(s, a, s') V^{\pi}(s'), \quad \forall s \in S, \forall a \in A.
\end{aligned}$$

$$\begin{aligned}
\text{maximize}_R \quad & \sum_s \sum_{a \in A \setminus \pi_E(s)} \left[Q^{\pi_E}(s, \pi_E(s)) - Q^{\pi_E}(s,a) \right] - \lambda \|R\|_1 \\
\text{subject to} \quad & R^{\pi_E} - R^a + \gamma (T^{\pi_E} - T^a)(I - \gamma T^{\pi_E})^{-1} R^{\pi_E} \succeq 0, \quad \forall a \in A \\
& |R(s,a)| \leq R_{max}, \quad \forall s \in S, \forall a \in A
\end{aligned}$$

Table 1: Optimization problem of IRL for MDP\R from the expert's policy.

By rephrasing with the matrix notations and substituting with Equation (10),

$$\begin{aligned}
& R^\pi + \gamma T^\pi V^\pi \succeq R^a + \gamma T^a V^\pi, \quad \forall a \in A \\
\Leftrightarrow\ & R^\pi + \gamma T^\pi (I - \gamma T^\pi)^{-1} R^\pi \succeq R^a + \gamma T^a (I - \gamma T^\pi)^{-1} R^\pi, \quad \forall a \in A \\
\Leftrightarrow\ & R^\pi - R^a + \gamma (T^\pi - T^a)(I - \gamma T^\pi)^{-1} R^\pi \succeq 0, \quad \forall a \in A.
\end{aligned}$$

■

Equation (9) bounds the feasible space of the reward functions that guarantee the optimality of the expert's policy, and there exist infinitely many reward functions that satisfy Equation (9). As a degenerate case, $R = 0$ is always a solution. Thus, given the expert's policy π_E, which is assumed to be optimal, the reward function is found by solving the optimization problem in Table 1, where λ is an adjustable weight for the penalty of having too many non-zero entries in the reward function. The objective is to maximize the sum of the margins[1] between the expert's policy and all other policies that deviate a single step from the expert's policy, in the hope that the expert's policy is optimal while favoring sparseness in the reward function.

3.2 IRL for MDP\R from Sampled Trajectories

In some cases, we have to assume that the expert's policy is not explicitly given but instead the trajectories of the expert's policy in the state and action spaces are available.[2] The m-th trajectory of the expert's policy is defined as the H-step state and action sequences $\{s_0^m, s_1^m, \cdots, s_{H-1}^m\}$ and $\{a_0^m, a_1^m, \cdots, a_{H-1}^m\}$.

In order to address problems with large state spaces, Ng and Russell (2000) use a linear approximation for the reward function, and we also assume that the reward function is linearly parameterized as

$$R(s,a) = \alpha_1 \phi_1(s,a) + \alpha_2 \phi_2(s,a) + \cdots + \alpha_d \phi_d(s,a) = \alpha^T \phi(s,a), \tag{11}$$

where known basis functions $\phi : S \times A \to [0,1]^d$ and the weight vector $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_d]^T \in \mathbb{R}^d$. We also assume without loss of generality that $\alpha \in [-1,1]^d$.

1. We found it more successful to use the sum-of-margins approach than the minimum-of-margins approach in the original paper, since the latter may fail when there are multiple optimal policies.
2. Although only the trajectories of states and actions are available, the transition function T is assumed to be known in MDP\R.

Algorithm 1 IRL for MDP\R from the sampled trajectories using LP.

Input: MDP\R $\langle S, A, T, \gamma \rangle$, basis functions ϕ, M trajectories

1: Choose a random initial policy π_1 and set $\Pi = \{\pi_1\}$.
2: **for** $k = 1$ to $MaxIter$ **do**
3: Find $\hat{\alpha}$ by solving the linear program:

$$\text{maximize}_{\hat{\alpha}} \quad \sum_{\pi \in \Pi} p\left(\hat{V}^{\pi_E}(s_0) - \hat{V}^{\pi}(s_0)\right)$$
$$\text{subject to} \quad |\hat{\alpha}_i| \leq 1 \quad i = 1, 2, \cdots, d$$

4: Compute an optimal policy π_{k+1} for the MDP with $\hat{R} = \hat{\alpha}^T \phi$.
5: **if** $\hat{V}^{\pi_E}(s_0) - V^{\pi_{k+1}}(s_0) \leq \varepsilon$ **then**
6: **return** $\hat{R}$
7: **else**
8: $\Pi = \Pi \cup \{\pi_{k+1}\}$
9: **end if**
10: **end for**
11: **return** $\hat{R}$

Output: the reward function $\hat{R}$

Then, from the given M trajectories, the value of π_E for the starting state s_0 is estimated by the average empirical return for an estimated reward function $\hat{R} = \hat{\alpha}^T \phi$:

$$\hat{V}^{\pi_E}(s_0) = \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{H-1} \gamma^t \hat{R}(s_t^m, a_t^m) = \frac{1}{M} \hat{\alpha}^T \sum_{m=1}^{M} \sum_{t=0}^{H-1} \gamma^t \phi(s_t^m, a_t^m).$$

The algorithm is presented in Algorithm 1. It starts with the set of policies Π initialized by a *base case* random policy π_1. Ideally, the true reward function R should yield $V^{\pi_E}(s_0) \geq V^{\pi}(s_0)$ for $\forall \pi \in \Pi$ since the expert's policy π_E is assumed to be an optimal policy with respect to R. The values of other policies with a candidate reward function $\hat{R}$ are either estimated by sampling trajectories or are exactly computed by solving the Bellman equation, Equation (1). The algorithm iteratively tries to find a better reward function $\hat{R}$, given the set of policies Π found by the algorithm $\Pi = \{\pi_1, \ldots, \pi_k\}$ up to iteration k, by solving the optimization problem in line 3, where $p(x)$ is a function that favors $x > 0$.[3] The algorithm then computes a new policy π_{k+1} that maximizes the value function under the new reward function, and adds π_{k+1} to Π. The algorithm continues until it has found a satisfactory reward function.

The above algorithm was extended for the apprenticeship learning in the MDP framework by Abbeel and Ng (2004). The goal of apprenticeship learning is to learn a policy from an expert's demonstrations without a reward function, so it does not compute the exact reward function that the expert is optimizing but rather the policy whose performance is close to that of the expert's policy on the unknown reward function. This is worth reviewing, as we adapt this algorithm to address the IRL problems in partially observable environments.

We assume that there are some known basis functions ϕ and the reward function is linearly parameterized with the weight vector α as in Equation (11). Also, assume $\|\alpha\|_1 \leq 1$ to bound R_{max} by 1. The value of a policy π can be written using the feature expectation $\mu(\pi)$ for the reward

3. Ng and Russell (2000) chose $p(x) = x$ if $x \geq 0$, and $p(x) = 2x$ if $x < 0$ in order to favor $x > 0$ but more penalize $x < 0$. The coefficient of 2 was heuristically chosen.

Algorithm 2 Apprenticeship learning using QCP.

Input: MDP\R $\langle S, A, T, \gamma \rangle$, basis functions ϕ, M trajectories

1: Choose a random initial weight α and set $\Pi = \emptyset$.
2: **repeat**
3: Compute an optimal policy π for the MDP with $R = \alpha^T \phi$.
4: $\Pi = \Pi \cup \{\pi\}$
5: Solve the following optimization problem:

$$\begin{aligned} &\text{maximize}_{t,\alpha} \quad t \\ &\text{subject to} \quad \alpha^T \mu_E \geq \alpha^T \mu(\pi) + t, \quad \forall \pi \in \Pi \\ &\qquad\qquad \|\alpha\|_2 \leq 1 \end{aligned}$$

6: **until** $t \leq \varepsilon$

Output: the reward function R

Algorithm 3 Apprenticeship learning using the projection method.

Input: MDP\R $\langle S, A, T, \gamma \rangle$, basis functions ϕ, M trajectories

1: Choose a random initial policy π_0.
2: Set $\mu_0 = \mu_0$ and $i = 1$.
3: **repeat**
4: Set $\alpha = \mu_E - \mu_{i-1}$.
5: Compute an optimal policy π_i for the MDP with $R = \alpha^T \phi$
6: Compute an orthogonal projection of μ_E onto the line through μ_{i-1} and μ_i.

$$\mu_i = \mu_{i-1} + \frac{(\mu_i - \mu_{i-1})^T (\mu_E - \mu_{i-1})}{(\mu_i - \mu_{i-1})^T (\mu_i - \mu_{i-1})} (\mu_i - \mu_{i-1})$$

7: Set $t = \|\mu_E - \mu_i\|_2$, and $i = i + 1$.
8: **until** $t \leq \varepsilon$

Output: the reward function R

function $R = \alpha^T \phi$ as follows :

$$V^{\pi}(s_0) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t))|s_0\Big] = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \alpha^T \phi(s_t, \pi(s_t))|s_0\Big]$$
$$= \alpha^T \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, \pi(s_t))|s_0\Big] = \alpha^T \mu(\pi),$$

where $\mu(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, \pi(s_t))|s_0]$. Since the expert's policy is not explicitly given, the feature expectation of the expert's policy cannot be exactly computed. Thus, we empirically estimate the expert's feature expectation $\mu_E = \mu(\pi_E)$ from the given expert's M trajectories of the visited states $\{s_0^m, s_1^m, \cdots, s_{H-1}^m\}$ and the executed actions $\{a_0^m, a_1^m, \cdots, a_{H-1}^m\}$ by

$$\hat{\mu}_E = \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{H-1} \gamma^t \phi(s_t^m, a_t^m).$$

Abbeel and Ng (2004) propose apprenticeship learning algorithms for finding a policy whose value is similar to that of the expert's policy based on the idea that the difference of the values between the obtained policy π and the expert's policy π_E is bounded by the difference between their feature expectations. Formally, this is written as follows:

$$\begin{aligned} |V^{\pi_E}(s_0) - V^{\pi}(s_0)| &= |\alpha^T \mu(\pi_E) - \alpha^T \mu(\pi)| \\ &\leq \|\alpha\|_2 \|\mu_E - \mu(\pi)\|_2 \\ &\leq \|\mu_E - \mu(\pi)\|_2 \end{aligned} \tag{12}$$

since $\|\alpha\|_1$ is assumed to be bounded by 1. The algorithm is presented in Algorithm 2. The optimization problem in line 5 can be considered as the IRL step that tries to find the reward function that the expert is optimizing. It is similar to the optimization problem in Algorithm 1, except that, the optimization problem cannot be modeled as a linear programming (LP) problem but rather as a quadratically constrained programming (QCP) problem because of L_2 norm constraint on α. Algorithm 3 is an approximation algorithm using the projection method instead of QCP, where μ_i denotes $\mu(\pi_i)$ for all i. Both algorithms terminate when $t \leq \varepsilon$. It is proved that both algorithms take a finite number of iterations to terminate (Abbeel and Ng, 2004).

4. IRL in Partially Observable Environments

We denote the problem of IRL in partially observable environments as IRL for POMDP\R and the objective is to determine the reward function that the expert is optimizing. Formally, IRL for POMDP\R is defined as follows: Given a POMDP\R $\langle S, A, Z, T, O, b_0, \gamma \rangle$ and an expert's policy π_E, find the reward function R that makes π_E an optimal policy for the given POMDP. Hence, the reward function found by IRL for POMDP\R should guarantee the optimality of the expert's policy for the given POMDP.

IRL for POMDP\R mainly suffers from two sources: First, IRL is fundamentally ill-posed, and second, computational intractability arises in IRL for POMDP\R in contrast with IRL for MDP\R. We describe these challenges below.

An IRL problem is an ill-posed problem, which is a mathematical problem that is not well-posed. The three conditions of a well-posed problem are existence, uniqueness, and stability of the

solution (Hadamard, 1902). IRL violates the condition of the uniqueness. An IRL problem may have an infinite number of solutions since there may be an infinite number of reward functions that guarantee the optimality of the given expert's policy. A degenerate one is the solution of every IRL problem since $R = 0$ yields every policy optimal. Also, given an optimal policy for a reward function, we can find some other reward function that yields the same optimal policy without any modification to the environment by the technique of reward shaping (Ng et al., 1999).

As suggested by Ng and Russell (2000), we can guarantee the optimality of the expert's policy by comparing the value of the expert's policy with that of all possible policies. However, there are an infinite number of policies in a finite POMDP, since a policy in a POMDP is defined as a mapping from a continuous belief space to a finite state space or represented by an FSC policy that might have an infinite number of nodes. In contrast, there are a finite number of policies in a finite MDP, since a policy in an MDP is defined as a mapping from a finite state space to a finite action space. In addition, in order to compare two policies in a POMDP, the values of those policies should be compared for all beliefs, because the value function is defined on a belief space. This intractability of IRL for POMDP\R originates from the same cause as the difficulty of solving a POMDP. The optimal policy of a POMDP is the solution of a belief-state MDP using the concept of belief. It is then difficult to solve an MDP with a continuous state space, since a policy and its value function are respectively defined as a mapping from the continuous state space to the finite action space and the real numbers.

In the following sections, we address the problem of IRL for POMDP\R, considering two cases as in the approaches to IRL for MDP\R. The first case is when the expert's policy is explicitly represented in the form of an FSC. The second case is when the expert's policy is implicitly given by the trajectories of the expert's executed actions and the corresponding observations. Although the second case has more wide applicability than the first case, the first case can be applied to some practical problems. For example, when building dialogue management systems, we may already have a dialogue policy engineered by human experts, but we still do not know the reward function that produces the expert's policy. We propose several methods for the problems of IRL for POMDP\R in these two cases. For the first case, we formulate the problem with constraints for the reward functions that guarantee the optimality of the expert's policy. To address the intractability of IRL for POMDP\R, we derive conditions involving a small number of policies and exploiting the result of the classical POMDP research. For the second case, we propose iterative algorithms of IRL for POMDP\R. The motivation for this approach is from Ng and Russell (2000). We also extend the algorithms proposed by Abbeel and Ng (2004) to partially observable environments.

5. IRL for POMDP\R from FSC Policies

In this section, we present IRL algorithms for POMDP\R when the expert's policy is explicitly given. We assume that the expert's policy is represented in the form of an FSC, since the FSC is one of the most natural ways to specify a policy in POMDPs.

We propose three conditions for the reward function to guarantee the optimality of the expert's policy based on comparing Q-functions and using the generalized Howard's policy improvement theorem (Howard, 1960) and the witness theorem (Kaelbling et al., 1998). We then complete the optimization problems to determine a desired reward function.

5.1 Q-function Based Approach

We could derive a simple and naive condition for the optimality of the expert's policy by comparing the value of the expert's policy with those of all other policies. Given an expert's policy π_E defined by a directed graph $\langle \mathcal{N}, \mathcal{E} \rangle$,

$$V^{\pi_E}(n,b) \geq V^{\pi'}(n',b), \quad \forall b \in \Delta_n, \forall n' \in \mathcal{N}', \tag{13}$$

for all nodes $n \in \mathcal{N}$ and all other policies π' defined by a directed graph $\langle \mathcal{N}', \mathcal{E}' \rangle$, where Δ_n denotes the set of all the beliefs where node n is optimal. Since V^{π_E} and $V^{\pi'}$ are linear in terms of the reward function R by Equations (5) and (7), the above inequality yields the set of linear constraints that defines the feasible region of the reward functions that guarantees the expert's policy to be optimal.

However, enumerating all the constraints is clearly infeasible because we have to take into account all other policies π' including those with an infinite number of nodes, as well as all the infinitely many beliefs in Δ_n. In other words, Equation (13) yields infinitely many linear constraints. Hence, we propose a simple heuristic for choosing a finite subset of constraints that hopefully yields a tight specification of the feasible region for the true reward function. First, among the infinitely many policies, we only consider polices that are *slightly* modified from the expert's policy since they are *similar* to the expert's policy yet must be suboptimal. We select as the similar policies those deviate one step from the expert's action and observation strategies, analogous to Equation (8). For each node $n \in \mathcal{N}$, there are $|A||\mathcal{N}|^{|Z|}$ ways to deviate from the expert's action and observation strategies, hence we consider a total of $|\mathcal{N}||A||\mathcal{N}|^{|Z|}$ policies that deviate one step from the experts' policy. Second, instead of considering all possible beliefs in Δ_n, we only consider the finitely sampled beliefs reachable by the expert's policy. The motivation for using the sampled beliefs comes from the fact that only the set of beliefs reachable under the optimal policy is important for solving POMDPs, and it is also widely used in most of the recent approximate POMDP solvers (Spaan and Vlassis, 2005; Smith and Simmons, 2005; Pineau et al., 2006; Ji et al., 2007; Kurniawati et al., 2008).

The above heuristic yields the following finite set of linear constraints: Given an expert's policy $\pi_E = \langle \psi, \eta \rangle$,

$$Q^{\pi_E}(\langle n,b \rangle, \langle \psi(n), \eta(n,\cdot) \rangle) \geq Q^{\pi_E}(\langle n,b \rangle, \langle a, os \rangle), \quad \forall b \in B_n, \forall a \in A, \forall os \in \mathcal{N}^Z, \tag{14}$$

for every node n in π_E, where $B_n \subseteq \Delta_n$ denotes the set of sampled beliefs that are visited at node n when following the expert's policy π_E from the initial belief b_0. The above condition states that any policy that deviates one step from the expert's action and observation strategies should not have a higher value than the expert's policy does. Note that the condition is a necessary though not a sufficient one, since we do not use the set of all other policies but use the set of $|\mathcal{N}||A||\mathcal{N}|^{|Z|}$ policies that have the same (or possibly fewer) number of nodes as the expert's policy, nor do we use the set of all beliefs in Δ_n but use the set of sampled beliefs.

We use a simple example illustrating the approach. Consider a POMDP with two actions and two observations, and the expert's policy π_E is the FSC represented by solid lines in Figure 1. The nodes are labeled with actions (a_0 and a_1) and the edges are labeled with observations (z_0 and z_1). In order to find the region of the reward functions that yields π_E as optimal, we build one-step deviating policies as mentioned above. The policies $\pi'_0, \pi'_1, \cdots, \pi'_7$ in the figure are the one-step deviating policies for node n_0 of π_E. Note that π'_i visits node n'_i instead of the original node n_0 and then exactly follows π_E. We then enumerate the constraints in Equation (14), comparing the value of π_E to that

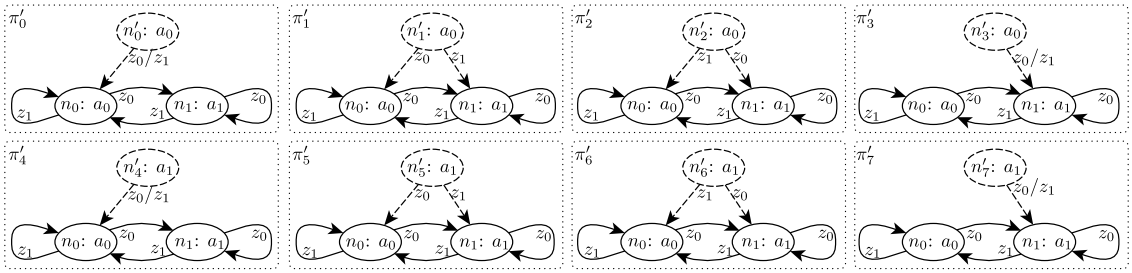


Figure 1: Set of the policies that deviate one step from node n_0 of the expert's policy.

of each one-step deviating policy. Specifically, the value at node n_0 of π_E is constrained to be not less than the value at node n'_i of π'_i, since deviating from the expert's policy should be suboptimal. To build the complete set of constraints in Equation (14), we additionally generate one-step deviating policies for node n_1 of π_E in a similar manner. We thus have $|\mathcal{N}||A||\mathcal{N}|^{|Z|} = 2 \times 2 \times 2^2 = 16$ policies that deviate one step from π_E.

5.2 Dynamic Programming (DP) Update Based Approach

A more systematic approach to defining the set of policies to be compared with the expert's policy is to use the set of FSC policies that arise during the DP update of the expert's policy. Given the expert's policy π_E, the DP update generates $|A||\mathcal{N}|^{|Z|}$ new nodes for all possible action and observation strategies, and these nodes can potentially be a new starting node. The expert's policy should be optimal if the value is not improved for any belief by the dynamic programming update. This idea comes from the generalized Howard's policy improvement theorem (Howard, 1960):

Theorem 2 [Hansen, 1998] *If an FSC policy is not optimal, the DP update transforms it into an FSC policy with a value function that is as good or better for every belief state and better for some belief state.*

The complete proof of the generalized policy improvement theorem can be found in Hansen (1998) but we give the full proof of the theorem for the convenience of the readers. First, we should prove the following lemma.

Lemma 1 *Given an FSC policy $\pi = \langle \psi, \eta \rangle$ and a node n_{new}, which is not included in π, the value function of node $n_{new} \in \mathcal{N}_{new}$ with the action strategy of selecting action a and observation strategy os is computed by*

$$V^{new}(n_{new}, s) = R(s,a) + \gamma \sum_{n',s'} T^{a,os}(\langle n_{new}, s \rangle, \langle n', s' \rangle) V^{\pi}(n', s'), \tag{15}$$

where V^{π} is calculated from Equation (5) and $T^{a,os}$ is defined in Equation (6). For some node n in π, if $V^{new}(n_{new}, s) \geq V^{\pi}(n, s)$, for $\forall s \in S$, the value of the original policy π will not be greater than that of the policy transformed by discarding node n and redirecting all the incoming edges of node n to node n_{new}.

Proof We build a new policy π_k that follows the original policy π, but executes the action and observation strategies of n_{new} for the first k times that node n is visited. The lemma is proved by the induction on the number of times k.

For the base step $k = 1$, the new policy π_1 executes the action and observation strategies of n_{new} only for the first time that node n is visited, and follows π for the rest of the time steps. Then, for any belief state b,

$$\begin{aligned} V^{\pi_1}(n,b) &= \sum_s b(s) V^{\pi_1}(n,s) = \sum_s b(s) V^{new}(n_{new},s) \\ &\geq \sum_s b(s) V^{\pi}(n,s) = V^{\pi}(n,b) \end{aligned}$$

since $V^{\pi_1}(n,s) = V^{new}(n_{new},s)$ for all $s \in S$ by the construction.

For the inductive step, we abuse notation to denote $R^{\pi_k}(s_t,a_t)$ as the reward at the t-th time step by following the policy π_k and starting from belief b and node n. Then, for any belief state b,

$$\begin{aligned} V^{\pi'_k}(n,b) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R^{\pi_k}(s_t,a_t)\big|b\Big] \\ &= \mathbb{E}\Big[\sum_{t=0}^{T_k-1} \gamma^t R^{\pi_k}(s_t,a_t) + \sum_{t=T_k}^{\infty} \gamma^t R^{\pi_k}(s_t,a_t)\big|b\Big] \\ &= \mathbb{E}\Big[\sum_{t=0}^{T_k-1} \gamma^t R^{\pi_k}(s_t,a_t)\big|b\Big] + \mathbb{E}\Big[\sum_{t=T_k}^{\infty} \gamma^t R^{\pi_k}(s_t,a_t)\big|b\Big] \\ &= \mathbb{E}\Big[\sum_{t=0}^{T_k-1} \gamma^t R^{\pi_{k-1}}(s_t,a_t)\big|b\Big] + \gamma^{T_k}\mathbb{E}\big[V^{\pi_k}(n,b_{T_k})|b\big] \\ &\geq \mathbb{E}\Big[\sum_{t=0}^{T_k-1} \gamma^t R^{\pi_{k-1}}(s_t,a_t)\big|b\Big] + \gamma^{T_k}\mathbb{E}\big[V^{\pi}(n,b_{T_k})|b\big] \\ &= V^{\pi_{k-1}}(n,b) \end{aligned}$$

where T_k represents the k-th time that node n is visited. The first equality holds by the definition of the value function. The fourth equality holds by the construction of π_{k-1} and π_k and the definition of the value function. The fifth inequality holds by $V^{\pi_k}(n,b_{T_k}) = V^{new}(n_{new},b_{T_k}) \geq V^{\pi}(n,b_{T_k})$, since π_k executes the action and observation strategies of n_{new} at b_{T_k} and executes those of n for the rest of the time. Hence, by induction, it follows that the value of the transformed policy cannot be decreased by replacing n with n_{new}. ■

Using the above lemma, we can prove Theorem 2.

Proof (of Theorem 2) The policy iteration algorithm (Hansen, 1998) transforms the policy by replacing the nodes with new nodes generated by the DP update using the following rules: (1) If there is an old node whose action and observation strategies are the same as those of a new node, the old node is unchanged. (2) If the value at an old node is less than the value at a new node, for any state, the old node is discarded and all the incoming edges of the old node are redirected to the new node. (3) The rest of new nodes are added to the original policy.

Since the value is not decreased by leaving the policy unchanged or adding a node to the policy, the first and the third transformation rules cannot decrease the value. Also, by the above lemma, the second transformation rule cannot decrease the value. Thus, the value of the transformed policy using the DP update does not decrease.

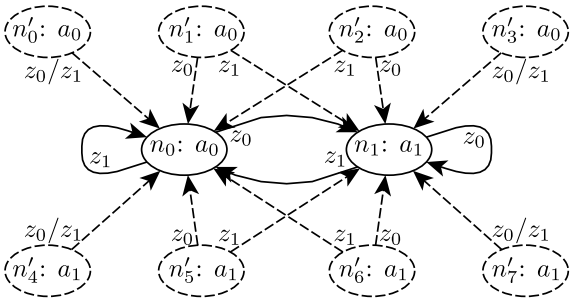


Figure 2: Set of the newly generated nodes by the DP update.

Also, if every node generated by the DP update is a duplicate of a node in the original policy, the optimality equation, Equation (4) is satisfied and the original policy is optimal. Thus, if the policy is not optimal, the DP update must generate some non-duplicate nodes that change the policy and improve the values for some belief state. ■

We should proceed with caution however in the sense that the DP update does not generate all the necessary nodes to guarantee the optimality of the expert's policy for every belief: The nodes in the expert's policy are only those reachable from the starting node n_0, which yields the maximum value at the initial belief b_0. Nodes that yield the maximum value at some other beliefs (i.e., useful) but are not reachable from n_0 are not present in the expert's policy. To guarantee the optimality of the expert's policy for every belief, we need to generate those non-existent but useful nodes. However, since there is no way to recover them, we only use nodes in the expert's policy and consider only the reachable beliefs by the expert's policy.

Let $\mathcal{N}_{new}$ be the set of nodes newly generated when transforming the expert's policy by the DP update, then $|\mathcal{N}_{new}| = |A||\mathcal{N}|^{|Z|}$. The value function of node $n_{new} \in \mathcal{N}_{new}$ is computed by Equation (15). The value function of policy π_E should satisfy

$$V^{\pi_E}(n,b) \geq V^{new}(n_{new},b), \quad \forall b \in B_n, \forall n_{new} \in \mathcal{N}_{new} \tag{16}$$

for every node $n \in \mathcal{N}$ if the expert's policy π_E is optimal. Note that V^{new} as well as V^{π_E} are linear in terms of the reward function R.

To illustrate the approach, we reuse the example in Section 5.1. Figure 2 shows π_E in solid lines and the set $\mathcal{N}_{new}$ of nodes generated by the DP update in dashed lines. We have $|A||\mathcal{N}|^{|Z|} = 2 \times 2^2 = 8$ nodes generated by the DP update, thus $\mathcal{N}_{new} = \{n'_0, n'_1, \cdots, n'_7\}$ is the complete set of nodes with all possible action and observation strategies. We then enumerate the constraints in Equation (16), making the value at each node of π_E no less than the values at the nodes in $\mathcal{N}_{new}$. Since the number of the newly generated nodes by the DP update is smaller than that of the policies generated by the Q-function based approach in Section 5.1, the computational complexity is significantly reduced.

5.3 Witness Theorem Based Approach

A more computationally efficient way to generate the set $\mathcal{N}_{new}$ of new nodes is to use the witness theorem (Kaelbling et al., 1998). We will exploit the witness theorem to find a set of useful nodes

that yield the feasible region for the true reward function as the witness algorithm incrementally generates new policy trees that improve the current policy trees. Here, we say that a node is useful if it has greater value than any other nodes at some beliefs. Formally speaking, given an FSC policy π, we define a set $B(n,U)$ of beliefs where the value function of node n dominates those of all other nodes in the set U:

$$B(n,U) = \{b | V^{new}(n,b) > V^{new}(n',b), \text{ for } \forall n' \in U \backslash \{n\}, \forall b \in \Delta\}$$

where $V^{new}(n,b) = \sum_s b(s) V^{new}(n,s)$ and $V^{new}(n,s)$ is computed by Equation (15). Node n is useful if $B(n,U) \neq \emptyset$, and U is a set of useful nodes if $B(n,U) \neq \emptyset$ for all $n \in U$. We re-state the witness theorem in terms of FSC policies as the following:

Theorem 3 [Kaelbling et al., 1998] *An FSC policy π is given as a directed graph $\langle \mathcal{N}, \mathcal{E} \rangle$. Let $\tilde{U}_a$ be a nonempty set of useful nodes with the action strategy of choosing action a, and U_a be the complete set of useful nodes with the action strategy of choosing action a. Then, $\tilde{U}_a \neq U_a$ if and only if there is some node $\tilde{n} \in \tilde{U}_a$, observation z^*, and node $n' \in \mathcal{N}$ for which there is a belief b such that*

$$V^{new}(n_{new}, b) \geq V^{\pi}(n,b)$$

for all $n \in \tilde{U}_a$, where n_{new} is a node that agrees with $\tilde{n}$ in its action and all its successor nodes except for observation z^*, for which $\eta(n_{new}, z^*) = n'$.

Proof The *if* direction of the statement is satisfied because b is a witness point for the existence of a useful node missing from $\tilde{U}_a$.

The *only if* direction can be rephrased as: If $\tilde{U}_a \neq U_a$ then there is a node $\tilde{n} \in \tilde{U}_a$, a belief state b, and a new node n_{new} that has a larger value than any other nodes $n \in \tilde{U}_a$ at b.

Choose some node $n^* \in U_a - \tilde{U}_a$. Since n^* is useful, there must be a belief b such that $V^{new}(n^*,b) > V^{new}(n',b)$ for all node $n' \in \tilde{U}_a$. Let $\tilde{n} = \text{argmax}_{n'' \in \tilde{U}_a} V^{new}(n'',b)$. Then, by the construction,

$$V^{new}(n^*,b) > V^{new}(\tilde{n},b). \tag{17}$$

Note that action a is always executed at n^* and $\tilde{n}$, since we consider only the nodes with the action strategy of choosing action a in the theorem.

Assume that for every observation z,

$$\begin{aligned} &\sum_s b(s) \sum_{s'} T(s,a,s') O(s',a,z) V^{\pi}(\eta(n^*,z),s') \\ &\quad \leq \sum_s b(s) \sum_{s'} T(s,a,s') O(s',a,z) V^{\pi}(\eta(\tilde{n},z),s'). \end{aligned}$$

Then

$$\begin{aligned} V^{new}(n^*,b) &= \sum_s b(s) \left[R(s,a) + \gamma \sum_{s'} T(s,a,s') \sum_z O(s',a,z) V^{\pi}(\eta(n^*,z),s') \right] \\ &\leq \sum_s b(s) \left[R(s,a) + \gamma \sum_{s'} T(s,a,s') \sum_z O(s',a,z) V^{\pi}(\eta(\tilde{n},z),s') \right] \\ &= V^{new}(\tilde{n},b) \end{aligned}$$

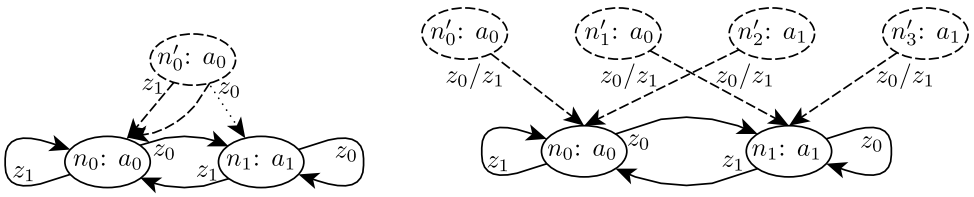


Figure 3: Set of the newly generated nodes by the witness theorem.

which contradicts (17). Thus, there must be some observation z^* such that

$$\sum_s b(s) \sum_{s'} T(s,a,s') O(s',a,z^*) V^\pi(\eta(n^*,z^*),s') > \sum_s b(s) \sum_{s'} T(s,a,s') O(s',a,z^*) V^\pi(\eta(\tilde{n},z^*),s').$$

Now, if $\tilde{n}$ and n^* differ in only one successor node, then the proof is complete with n^*, which can serve as the n_{new} in the theorem. If n and n^* differ in more than one successor node, we will identify another node that can act as n_{new}. Define n_{new} to be identical to $\tilde{n}$ except for observation z^*, for which $\eta(n_{new},z^*) = \eta(n^*,z^*)$. From this, it follows that

$$\begin{aligned} V^{new}(n_{new},b) &= \sum_s b(s) \big[R(s,a) + \gamma \sum_{s'} T(s,a,s') \sum_z O(s',a,z) V^\pi(\eta(n_{new},z),s')\big] \\ &> \sum_s b(s) \big[R(s,a) + \gamma \sum_{s'} T(s,a,s') \sum_z O(s',a,z) V^\pi(\eta(\tilde{n},z),s')\big] \\ &= V^{new}(\tilde{n},b) \geq V^{new}(n,b) \end{aligned}$$

for all $n \in \tilde{U}_a$. Therefore, the nodes $\tilde{n}$ and n_{new}, the observation z^*, $n' = \eta(n^*,z^*)$, and the belief state b satisfy the conditions of the theorem. ■

The witness theorem tells us that if a policy π is optimal, then the value of n_{new} generated by changing the successor node of each single observation should not increase for any possible beliefs. This leads us to a smaller set of inequality constraints compared to Equation (16), by defining $\mathcal{N}_{new}$ in a different way.

Let $\mathcal{N}_a = \{n \in \mathcal{N} | \psi(n) = a\}$ and $A_{-\mathcal{N}} = \{a \in A | \mathcal{N}_a = \emptyset\}$. For each action $a \notin A_{-\mathcal{N}}$, we generate new nodes by the witness theorem: For each node $\tilde{n} \in \mathcal{N}_a$, $z^* \in Z$, and $n' \in \mathcal{N}$, we make n_{new} such that $\psi(n_{new}) = \psi(\tilde{n}) = a$ and $\eta(n_{new},z) = \eta(\tilde{n},z)$ for all $z \in Z$ except for z^*, for which $\eta(n_{new},z^*) = n'$. The maximum number of newly generated nodes by the witness theorem is $\sum_a |\mathcal{N}_a||\mathcal{N}||Z| \leq |\mathcal{N}|^2|Z|$. Then, for each action $a \in A_{-\mathcal{N}}$, we use the DP update to generate $|A_{-\mathcal{N}}||\mathcal{N}|^{|Z|}$ additional nodes. The number of newly generated nodes $|\mathcal{N}_{new}|$ is no more than $|\mathcal{N}|^2|Z| + |A_{-\mathcal{N}}||\mathcal{N}|^{|Z|}$. Note that this number is often much less than $|A||\mathcal{N}|^{|Z|}$, the number of newly generated nodes by DP update, since the number of actions $|A_{-\mathcal{N}}|$ that is not executed at all by the expert's policy is typically much fewer than $|A|$.

We again reuse the example in Section 5.1 to illustrate the approach. We build the set $\mathcal{N}_{new}$ of new nodes using the witness theorem. The left panel of Figure 3 shows the construction of

$$\begin{aligned}
\text{maximize}_R \quad & \sum_{n\in\mathcal{N}}\sum_{b\in B_n}\sum_{\substack{a\in A\setminus\psi(n)\\ os\in\mathcal{N}^Z\setminus\eta(n,\cdot)}}\Big[V^\pi(\langle n,b\rangle)-Q^\pi(\langle n,b\rangle,\langle a,os\rangle)\Big]-\lambda\|R\|_1\\
\text{subject to} \quad & Q^\pi(\langle n,b\rangle,\langle\psi(n),\eta(n,\cdot)\rangle)\geq Q^\pi(\langle n,b\rangle,\langle a,os\rangle),\\
& \qquad\qquad \forall b\in B_n,\forall a\in A,\forall os\in\mathcal{N}^Z,\forall n\in\mathcal{N}\\
& |R(s,a)|\leq R_{max},\quad \forall s,\forall a
\end{aligned}$$

Table 2: Optimization problem using Q-function based optimality constraint.

$$\begin{aligned}
\text{maximize}_R \quad & \sum_{n\in\mathcal{N}}\sum_{b\in B_n}\sum_{n_{new}\in\mathcal{N}_{new}}\Big[V^\pi(n,b)-V^{new}(n_{new},b)\Big]-\lambda\|R\|_1\\
\text{subject to} \quad & V^\pi(n,b)\geq V^{new}(n_{new},b),\quad \forall b\in B_n,\forall n_{new}\in\mathcal{N}_{new},\forall n\in\mathcal{N}\\
& |R(s,a)|\leq R_{max},\quad \forall s,\forall a
\end{aligned}$$

Table 3: Optimization problem using the DP update or the witness theorem based optimality constraint.

new node n_0' from node n_0 such that $\psi(n_0')=\psi(n_0)=a_0$ and $\eta(n_0',z_1)=\eta(n_0,z_1)$. The original observation strategy of n_0 for z_0 transits to n_1 (shown in dotted line), and it is changed to n_0 (shown in dashed line). The right panel in the figure presents the complete set $\mathcal{N}_{new}$ of generated nodes using the witness theorem (shown in dashed lines). Nodes n_0' and n_1' are generated from node n_0, whereas nodes n_2' and n_3' are from node n_1. Note that $A_{-\mathcal{N}}=\emptyset$ since π_E executes all actions in the model. We thus have a total of 4 generated nodes, which is smaller than those generated by either the Q-function based or the DP update based approach.

5.4 Optimization Problem

In the previous sections, we suggested three constraints for the reward function that stem from the optimality of the expert's policy, but infinitely many reward functions can satisfy the constraints in Equations (14) and (16). We thus present constrained optimization problems with objective functions that encode our preference on the learned reward function. As in Ng and Russell (2000), we prefer a reward function that maximizes the sum of the margins between the expert's policy and other policies. At the same time, we want the reward function as sparse as possible, which can be accomplished by adjusting the penalty weight on the L_1-norm of the reward function. If we use Q-function based optimality constraint, that is, Equation (14), the value of the expert's policy is compared with those of all other policies that deviate from the expert's action and observation strategies, given in Table 2. When using the DP update or the witness theorem based optimality constraint, that is, Equation (16), the policies other than the expert's policy are captured in newly generated nodes n_{new}, hence the optimization problem now becomes the one given in Table 3. Since all the inequalities and the objective functions in the optimization problems are linear in terms

of the reward function, the desired reward function can be found efficiently by solving the linear programming problems.

When using Q-function or the DP update based approach, the number of policies compared with the expert's is exponential to the number of observations, and hence the number of constraints in the optimization problems increases exponentially. This may become intractable even for a small size expert's policy. We can address this limitation using the witness theorem based approach, since it is sufficient to consider as few as $|\mathcal{N}|^2|Z|$ nodes if the expert's policy executes all actions, which is common in many POMDP benchmark problems.

6. IRL for POMDP\R from Sampled Trajectories

In some cases, the expert's policy may not be explicitly given, but the records of the expert's trajectories may be available instead.[4] Here, we assume that the set of H-step belief trajectories is given. The m-th trajectory is denoted by $\{b_0^m, b_1^m, \ldots, b_{H-1}^m\}$, where $b_0^m = b_0$ for all $m \in \{1, 2, \cdots, M\}$. If the trajectories of the perceived observations $\{z_0^m, z_1^m, \ldots, z_{H-1}^m\}$ and the executed actions $\{a_0^m, a_1^m, \ldots, a_{H-1}^m\}$ following the expert's policy are available instead, we can reconstruct the belief trajectories by using the belief update in Equation (3).

In order to obtain an IRL algorithm for POMDP\R from the sampled belief trajectories, we linearly parameterize the reward function using the known basis functions $\phi : S \times A \to [0,1]^d$ and the weight vector $\alpha \in [-1,1]^d$ as in Equation (11): $R(s,a) = \alpha^T \phi(s,a)$. This assumption is useful for the problems with large state spaces, because with some prior knowledge about the problems, we can represent the reward function compactly using the basis functions. For example, in robot navigation problems, the basis function can be chosen to capture the features of the state space, such as which locations are considered dangerous. In the worst case when no such prior knowledge is available, the basis functions may be designed for each pair of state and action so that the number of basis functions is $|S| \times |A|$. The objective of IRL is then to determine the (unknown) parameter α of the reward function $R = \alpha^T \phi$.

In this section, we propose three trajectory-based IRL algorithms for POMDP\R. The algorithms share the same framework that iteratively repeats estimating the parameter of the reward function using an IRL algorithm and computing an optimal policy for the estimated reward function using a POMDP solver. The first algorithm finds the reward function that maximizes the margin between the values of the expert's policy and other policies for the sampled beliefs using LP. This is a simple extension to Ng and Russell (2000). The second algorithm computes the reward function that maximizes the margin between the feature expectations of the expert's policy and other policies using QCP. The last algorithm approximates the second using the projection method. The second and third algorithms are extended from the methods originally suggested for MDP environments by Abbeel and Ng (2004).

4. As in the IRL for MDP\R from sampled trajectories, we assume that the transition and observation functions are known in POMDP\R.

Algorithm 4 IRL for POMDP\R from the sampled trajectories using the MMV method.

Input: POMDP\R $\langle S, A, Z, T, O, b_0, \gamma \rangle$, basis functions ϕ, M trajectories

Choose a set B^{π_E} of all the unique beliefs in the trajectories.
Choose a random initial policy π_1 and set $\Pi = \{\pi_1\}$.
for $k = 1$ to $MaxIter$ **do**
    Find $\hat{\alpha}$ by solving the linear program:

$$\begin{aligned} &\text{maximize}_{\hat{\alpha}} \quad \sum_{\pi \in \Pi} \sum_{b \in B^{\pi_E}} p\left(\hat{V}^{\pi_E}(b) - V^{\pi}(b)\right) - \lambda \|\hat{\alpha}^T \phi\|_1 \\ &\text{subject to} \quad |\hat{\alpha}_i| \leq 1, \quad i = 1, 2, \cdots, d \end{aligned}$$

    Compute an optimal policy π_{k+1} for the POMDP with $\hat{R} = \hat{\alpha}_k^T \phi$.
    if $|\hat{V}^{\pi_E}(b) - V^{\pi_{k+1}}(b)| \leq \varepsilon, \forall b \in B^{\pi_E}$ **then**
        return $\hat{R} = \hat{\alpha}_k^T \phi$
    else
        $\Pi = \Pi \cup \{\pi_{k+1}\}$
    end if
end for
$K = \text{argmin}_{k:\pi_k \in \Pi} \max_{b \in B^{\pi_E}} |\hat{V}^{\pi_E}(b) - V^{\pi_k}(b)|$
return $\hat{R} = \hat{\alpha}_K^T \phi$

Output: the reward function $\hat{R}$

6.1 Max-Margin between Values (MMV) Method

We first evaluate the values of the expert's policy and other policies for the weight vector α of a reward function in order to compare their values. The reward for belief b is then calculated by

$$R(b,a) = \sum_{s \in S} b(s) R(s,a) = \sum_{s \in S} b(s) \alpha^T \phi(s,a) = \alpha^T \phi(b,a),$$

where $\phi(b,a) = \sum_{s \in S} b(s) \phi(s,a)$. We also compute $\hat{V}^{\pi_E}(b_0^m)$ to be the empirical return of the expert's m-th trajectory by

$$\hat{V}^{\pi_E}(b_0^m) = \sum_{t=0}^{H-1} \gamma^t R(b_t^m, a_t^m) = \sum_{t=0}^{H-1} \gamma^t \alpha^T \phi(b_t^m, a_t^m).$$

Noting that $b_0^m = b_0$ for all m, the expert's average empirical return at b_0 is given by

$$\hat{V}^{\pi_E}(b_0) = \frac{1}{M} \sum_{m=1}^{M} \hat{V}^{\pi_E}(b_0^m) = \alpha^T \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{H-1} \gamma^t \phi(b_t^m, a_t^m), \tag{18}$$

which is linear in terms of α. In a similar manner, we can compute the average empirical return of the expert's trajectories at other beliefs b_j by

$$\hat{V}^{\pi_E}(b_j) = \frac{1}{M_j} \sum_{m=1}^{M} \hat{V}^{\pi_E}(b_j) = \alpha^T \frac{1}{M_j} \sum_{m=1}^{M} \sum_{t=H_j^m}^{H-1} \gamma^{t-H_j^m} \phi(b_t^m, a_t^m), \tag{19}$$

where H_j^m is the first time that b_j is found in the m-th trajectory and M_j is the number of trajectories that contain b_j.

Given the above definitions, the rest of the derivation is fairly straightforward, and leads to a similar algorithm to that of Ng and Russell (2000). The algorithm is shown in Algorithm 4. It iteratively tries to find a reward function parameterized by α that maximizes the sum of the margins between the value $\hat{V}^{\pi_E}$ of the expert's policy and the value V^{π} of each FSC policy $\pi \in \Pi$ found so far by the algorithm at all the unique beliefs $b \in B^{\pi_E}$ in the trajectories. We could consider the initial belief b_0 alone, similar to Ng and Russell (2000) considering the initial state s_0 alone. However, we found it more effective in our experiments to include additional beliefs, since they often provide better guidance in the search of the reward function by tightening the feasible region. In order to consider the additional beliefs, we should be able to compute the value V^{π} of the intermediate policy π at belief $b \in B^{\pi_E}$, but it is not well defined. b may be unreachable under π and it is not known that we will visit b at which node of π. In our work, we use an upperbound approximation given as

$$V^{\pi}(b) \approx \max_{n} V^{\pi}(n,b), \tag{20}$$

where $V^{\pi}(n,b)$ is computed by Equation (7).

The IRL step in line 4 finds the reward function that guarantees the optimality of the expert's policy. In the optimization problem, we constrain the value of the expert's policy to be greater than that of other policies in order to ensure that the expert's policy is optimal, and maximize the sum of the margins between the expert's policy and other policies using a monotonically increasing function p.[5] In addition, we prefer the sparse reward function and the sparsity of the learned reward function can be achieved by tuning the penalty weight λ. Note that we can solve the IRL step in Algorithm 4 using LP since all the variables such as $\hat{V}^{\pi_E}$ and V^{π} are linear functions in terms of α from Equations (18), (19), and (20).

When π_{k+1} matches π_E, the differences in the value functions for all beliefs will vanish. Hence, the algorithm terminates when all the differences in the values are below the threshold ε, or the iteration number has reached the maximum number of steps *MaxIter* to terminate the algorithm in a finite number of iterations.

6.2 Max-Margin Between Feature Expectations (MMFE) Method

We can re-write the value of a FSC policy π in POMDPs using the feature expectation $\mu(\pi)$, proposed by Abbeel and Ng (2004) as follows:

$$\begin{aligned} V^{\pi}(b_0) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(b_t,a_t)|\pi,b_0\Big] = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \alpha^T \phi(b_t,a_t)|\pi,b_0\Big] \\ &= \alpha^T \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \phi(b_t,a_t)|\pi,b_0\Big] = \alpha^T \mu(\pi), \end{aligned}$$

where $\mu(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(b_t,a_t)|\pi,b_0]$, and it is assumed that $\|\alpha\|_1 \leq 1$ to bound R_{max} by 1. In order to compute the feature expectation $\mu(\pi)$ exactly, we define the occupancy distribution $occ^{\pi}(s,n)$ of the policy π that represents the relative frequency of visiting state s at node n when following the policy $\pi = \langle \psi, \eta \rangle$ and starting from belief b_0 and node n_0. It can be calculated by solving the

5. We simply choose $p(x) = x$ if $x > 0$ and $p(x) = 2x$ if $x < 0$ as in Ng and Russell (2000). This gives more penalty to violating the optimality of the expert's policy.

Algorithm 5 IRL for POMDP\R from the sampled trajectories using the MMFE method.

Input: POMDP\R $\langle S, A, Z, T, O, b_0, \gamma \rangle$, basis functions ϕ, M trajectories

Choose a random initial weight α_0.
$\Pi = \emptyset$, $\Omega = \emptyset$, and $t = \infty$.
for $k = 1$ to $MaxIter$ **do**
  Compute an optimal policy π_{k-1} for the POMDP with $R = \alpha_{k-1}^T \phi$.
  $\Pi = \Pi \cup \{\pi_{k-1}\}$ and $\Omega = \Omega \cup \{\alpha_{k-1}\}$.
  if $t \leq \varepsilon$ **then**
    break
  end if
  Solve the following optimization problem:

$$\begin{aligned} &\text{maximize}_{t,\alpha_k} \quad t \\ &\text{subject to} \quad \alpha_k^T \mu_E \geq \alpha_k^T \mu(\pi) + t, \quad \forall \pi \in \Pi \\ &\qquad\qquad \|\alpha_k\|_2 \leq 1 \end{aligned}$$

end for
$K = \text{argmin}_{k:\pi_k \in \Pi} \|\mu_E - \mu(\pi_k)\|_2$
return $R = \alpha_K^T \phi$

Output: the reward function R

following system of linear equations:

$$\begin{aligned} occ^{\pi}(s', n') = b_0(s', n')\delta_{n', n_0} + \\ \gamma \sum_{s,z,n} occ^{\pi}(s, n) T(s, \psi(n), s') O(s', \psi(n), z) \delta_{n', \eta(n,z)}, \quad \forall s' \in S, \forall n' \in \mathcal{N}, \end{aligned}$$

where $\delta_{x,y}$ denotes the Kronecker delta function, defined as $\delta_{x,y} = 1$ if $x = y$ and $\delta_{x,y} = 0$ otherwise. With the occupancy distribution, the value of the policy π can be computed by

$$V^{\pi}(b_0) = \sum_{s,n} occ^{\pi}(s, n) R(s, \psi(n)) = \sum_{s,n} occ^{\pi}(s, n) \alpha^T \phi(s, \psi(n)) = \alpha^T \mu(\pi),$$

where $\mu(\pi) = \sum_{s,n} occ^{\pi}(s, n) \phi(s, \psi(n))$. However, the feature expectation of the expert's policy π_E cannot be exactly computed, because we only have the set of trajectories on the belief space, which are recovered from the given trajectories of the actions and the observations, instead of the explicit FSC form of the expert's policy. Hence, we estimate the expert's feature expectation $\mu(\pi_E) = \mu_E$ empirically by

$$\hat{\mu}_E = \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{H-1} \gamma^t \phi(b_t^m, a_t^m).$$

From these definitions, we can derive the following inequalities, which are similar to Equation (12),

$$\begin{aligned} |V^{\pi_E}(b_0) - V^{\pi}(b_0)| &= |\alpha^T \mu_E - \alpha^T \mu(\pi)| \\ &\leq \|\alpha\|_2 \|\mu_E - \mu(\pi)\|_2 \\ &\leq \|\mu_E - \mu(\pi)\|_2. \end{aligned} \tag{21}$$

Algorithm 6 IRL for POMDP\R from the sampled trajectories using the PRJ method.

Input: POMDP\R $\langle S, A, Z, T, O, b_0, \gamma \rangle$, basis functions ϕ, M trajectories

1: Choose a random initial weight α_0.
2: Compute an optimal policy π_0 for the POMDP $R = \alpha_0^T \phi$.
3: $\Pi = \{\pi_0\}$, $\Omega = \{\alpha_0\}$, $\mu_0 = \mu_0$ and $t = \infty$.
4: **for** $k = 1$ to *MaxIter* **do**
5: $\quad \alpha_k = \mu_E - \mu_{k-1}$.
6: $\quad$ Compute an optimal policy π_k for the POMDP $R = \alpha_k^T \phi$.
7: $\quad \Pi = \Pi \cup \{\pi_k\}$ and $\Omega = \Omega \cup \{\alpha_k\}$.
8: $\quad$ **if** $t \leq \varepsilon$ **then**
9: $\quad\quad$ break
10: $\quad$ **end if**
11: $\quad$ Compute an orthogonal projection of μ_E onto the line through μ_{k-1} and μ_k

$$\mu_k = \mu_{k-1} + \frac{(\mu_k - \mu_{k-1})^T (\mu_E - \mu_{k-1})}{(\mu_k - \mu_{k-1})^T (\mu_k - \mu_{k-1})} (\mu_k - \mu_{k-1})$$

12: $\quad t = \|\mu_E - \mu_k\|_2$
13: **end for**
14: $K = \text{argmin}_{k:\pi_k \in \Pi} \|\mu_E - \mu_k\|_2$.
15: **return** $R = \alpha_K^T \phi$

Output: the reward function R

The last inequality holds since we assume $\|\alpha\|_1 \leq 1$. The above inequalities state that the difference between the expert's policy π_E and any policy π is bounded by the difference between their feature expectations, which is the same result as in Abbeel and Ng (2004). Based on Equation (21), we can easily extend Algorithm 2 to address the IRL problem for POMDP\R from the sampled trajectories. The algorithm is presented in Algorithm 5. While we can solve Algorithm 4 using LP, the algorithm requires a QCP solver, since the optimization problem in line 9 has a 2-norm constraint on α. Note that it is proved that the algorithm will terminate in a finite number of iterations in Abbeel and Ng (2004).

Abbeel and Ng (2004) construct a policy by mixing the policies found by the algorithm in order to find the policy that is as good as the given expert's policy. They choose the weight of the policies by computing the convex combination of feature expectations that minimizes the distance to the expert's feature expectation. However, this method cannot be adapted to our IRL algorithm, because there is no way to recover the reward function that provides the computed mixed policy. Thus, we return the reward function that yields the closest feature expectation to that of the expert's policy among the intermediate reward functions found by the algorithm. By Equation (21), the value of the policy that generates the closest feature expectation is assured to be similar to the value of the expert's policy and we hope that the reward function that yields the closest feature expectation will be similar to the reward function that the expert is optimizing.

6.3 Projection (PRJ) Method

In the previous section, we described the IRL algorithm for POMDP\R from the sampled trajectories using QCP. We can now address the problem using a simpler method, as Abbeel and Ng (2004) proposed. The IRL step in Algorithm 5 can be considered for finding the unit vector μ_k or-

thogonal to the maximum margin hyperplane that classifies feature expectations into two sets: One set consists of the expert's feature expectation and the other set consists of the feature expectations of the policies found by the algorithm. The unit vector μ_k can then be approximately computed by projecting the expert's feature expectation on the line between the feature expectations of the most recent policy and the previously projected point. The algorithm is shown in Algorithm 6. In the algorithm, μ_i denotes $\mu(\pi_i)$ for all i and μ denotes the point where the expert's feature expectation is projected. Similar to Algorithm 5, the algorithm returns the reward function that yields the closest feature expectation to that of the expert's policy among the intermediate reward functions found by the algorithm.

7. Experimental Results

In this section, we present the results from the experiments on some POMDP benchmark domains - *Tiger*, *1d Maze*, *5×5 Grid World*, *Heaven/Hell*, and *Rock Sample* problems. The characteristics of each problem is presented in Table 4 and brief explanations are given below.

The *Tiger* and *1d Maze* problems are classic POMDP benchmark problems (Cassandra et al., 1994). In the *Tiger* problem, an agent is standing in front of two doors. There is a reward behind one of the doors and a tiger behind the other. If the agent opens the door with the tiger, it gets a large penalty (-100). Otherwise, it receives the reward (+10). The agent initially does not know the location of the tiger. It can infer the location of the tiger by listening for the sound of the tiger with a small cost (-1) and the correct information is given with some probability (0.85). In the *1d Maze* problem, there are 4 states as presented in the first panel of Figure 4. The third state from the left is the goal state. An agent is initially set to the non-goal states with equal probabilities and can move left or right. The agent observes whether it is at the goal state or not. When the agent reaches the goal state, it is randomly moved to a non-goal state after executing any action.

The 5×5 *Grid World* problem is inspired by a problem in Ng and Russell (2000), where the states are located as shown in the second panel of Figure 4. An agent can move west, east, north or south, and their effects are assumed to be deterministic. The agent always starts from the north-west corner of the grid and the goal is at the south-east corner. After the agent reaches the goal state, the agent restarts from the start state by executing any action in the goal state. The current position cannot be observed directly but the presence of the adjacent walls can be perceived without noise. Hence, there are nine observations, eight of them corresponding to eight possible configurations of the nearby walls when on the border (N, S, W, E, NW, NE, SW, and SE), and one corresponding to no wall observation when not on the border (Null).

The *Heaven/Hell* problem (Geffner and Bonet, 1998) is a navigation problem over the states depicted in the third panel of Figure 4. The goal state is either position 4 or 6. One of these is heaven and the other is hell. When the agent reaches heaven, it receives a reward (+1). When it reaches hell, it receives a penalty (-1). It starts at position 0, and does not know the position of heaven. However, it can get the information about the position of heaven after visiting the priest at position 9. The agent always perceives its current position without any noise. After reaching heaven or hell, it is moved at the initial position.

The *Rock Sample* problem (Smith and Simmons, 2004) models a rover that moves around an area and samples rocks. The locations of the rover and the rocks are known (the rocks are marked with stars in the fourth panel of Figure 4), but the value of the rocks are unknown. If it samples a good rock, it receives a reward (+10), but if it samples a bad rock, it receives a penalty (-10). When

Problem	$\|S\|$	$\|A\|$	$\|Z\|$	γ	$\|\mathcal{N}\|$	$\|\cup_{n\in\mathcal{N}} B_n\|$
Tiger	2	3	2	0.75	5	5
1d Maze	4	2	2	0.75	3	4
5×5 *Grid World*	25	4	9	0.90	2	13
Heaven/Hell	20	4	11	0.99	18	19
Rock Sample[4,3]	129	8	2	0.95	16	22

Table 4: Characteristics of the problem domains used in the experiments. γ: The discount factor. $|\mathcal{N}|$: The number of nodes in the optimal policy. $|\cup_{n\in\mathcal{N}} B_n|$: The total number of beliefs reachable by the optimal policy.

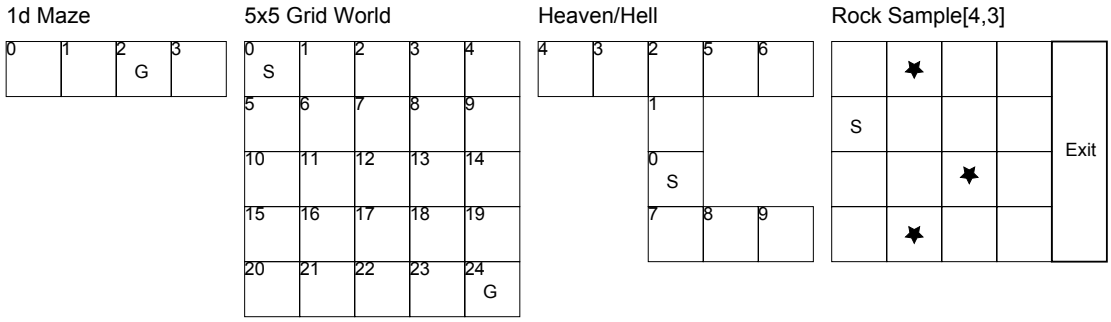


Figure 4: Maps for the *1d Maze*, 5×5 *Grid World*, *Heaven/Hell*, and *Rock Sample*[4,3] problems.

the rover tries to sample at the location without any rocks, it receives a large penalty (-100). The rover can observe the value of the rocks with a noisy long range sensor. In addition, it gets a reward (+1) if it reaches the right side of the map. When it reaches other sides of the map, it gets a large penalty (-100). The rover is immediately moved to the start position when it traverses outside of the map. The *Rock Sample* problem is instantiated as *Rock Sample*$[n,k]$, which describes that the size of the map is $n\times n$ and the number of the rocks on the map is k, and our experiment was performed on *Rock Sample*[4,3].

To evaluate the performance of the IRL algorithms, we could naively compare the true reward functions in the original problems to the reward functions found by the algorithms. However, it is not only difficult but also meaningless to simply compare the numerical values of the reward functions, since the reward function represents the relative importance of executing an action in a state. Completely different behaviors may be derived from two reward functions that have a small difference, and an identical optimal policy may be induced by two reward functions that have a large difference. For example, three reward functions in the *Tiger* problem are presented in Table 5, where R^* is the true reward function and R_1 and R_2 are two reward functions chosen for explaining the phenomenon. When the distances are measured by L_2 norm,

$$Dist(R,R^*) = \|R^* - R\|_2 = \sqrt{\sum_{s\in S, a\in A} (R^*(s,a) - R(s,a))^2},$$

the reward function R_2 is more similar to R^* than the reward function R_1. However, as shown in Figure 5, the optimal policies for R^* and R_1 are exactly the same while the optimal policy for R_2

	Listen	*Success*	*Failure*	$Dist(R,R^*)$	$V^{\pi}(b_0;R^*)$
R^*	-1	10	-100	0	1.93
R_1	-1	5.68	-100	6.10	1.93
R_2	3.05	10	-100	5.73	1.02

Table 5: Three reward functions in the *Tiger* problem. R^* is the true reward function. *Listen*: The negative cost of listening. *Success*: The reward of opening the correct door. *Failure*: The negative penalty of choosing the door with the tiger. $Dist(R,R^*)$: The distance from the true reward functions. $V^{\pi}(b_0;R^*)$: The value of the optimal policy for each reward function measured on the true reward function.

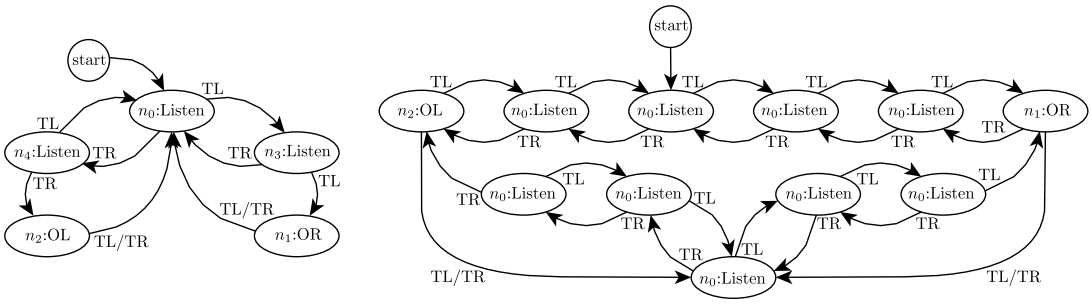


Figure 5: Optimal policies for the reward functions in Table 5. The nodes are labeled with actions (Listen, OL: Open-left, OR: Open-right). The edges are labeled with observations (TL: Tiger-left, TR: Tiger-right). *Left*: The optimal policy for R^* and R_1. *Right*: The optimal policy for R_2.

is different from that for R^*. If we still want to directly evaluate the computed reward function using a distance measure, we could apply the policy-invariant reward transformation on the true reward function and compute the minimum distance, but it is non-trivial to do so since there is an infinite number of transformations to choose from including the positive linear transformation and the potential-based shaping (Ng et al., 1999). Therefore, we compare the value functions of the optimal policies induced from the true and learned reward functions instead of directly measuring the distance between the reward functions.

The performance of the algorithms are evaluated by the differences in the values of the expert's policy and the optimal policy for the learned reward function. In the evaluations, the value of each policy is measured on the true reward function R^* and the learned reward function R_L, and we define the value $V^{\pi}(b_0;R)$ of a policy π at the initial belief b_0 measured on a reward function R as

$$V^{\pi}(b_0;R) = \sum_{s \in S} b_0(s) V^{\pi}(n_0,s;R),$$

where n_0 is the starting node of a policy π and $V^{\pi}(n_0,s;R)$ is computed by Equation (5) using the reward function R.

Problem	$D(R^*)$	$D(R_L)$	$\lvert\mathcal{N}_{new}\rvert$			Time		
			Q-IRL	D-IRL	W-IRL	Q-IRL	D-IRL	W-IRL
Tiger	0	0	375	75	39	0.07	0.04	0.03
1d Maze	0	0	54	18	12	0.02	0.02	0.02
5×5 *Grid World*	0	0	4096	2048	1044	78.06	3.00	1.54
Heaven/Hell	0	0	4.63×10^{15}	2.57×10^{14}	3260	n.a.	n.a.	6.20
Rock Sample[4,3]	13.42	0	32768	2048	634	77.99	19.20	3.77

n.a. = not applicable

Table 6: Results of IRL for POMDP\R from FSC policies. Q-IRL, D-IRL, and W-IRL respectively denote the Q-function based approach, the DP update based approach, and the witness theorem based approach. $D(R^*) = |V^{\pi_E}(b_0;R^*) - V^{\pi_L}(b_0;R^*)|$. $D(R_L) = |V^{\pi_E}(b_0;R_L) - V^{\pi_L}(b_0;R_L)|$. $|\mathcal{N}_{new}|$ denotes the number of newly generated policies. The average computation times are reported in seconds

Our algorithm requires a POMDP solver for computing the expert's policy and the intermediate optimal policies of the learned rewards. Since we assume the policy is in the form of an FSC, we use PBPI (Ji et al., 2007), which finds an optimal FSC policy approximately on the reachable beliefs. Optimization problems formulated in LP and QCP are solved using ILOG CPLEX.

The experiments are organized into two cases according to the representation of the expert's policy. In the first case, the expert's policy is explicitly given in the form of a FSC, and in the second case, the trajectories of the expert's executed actions and the corresponding observations are given instead.

7.1 Experiments on IRL from FSC Policies

The first set of experiments concerns the case in which the expert's policy is explicitly given using the FSC representation. We experimented with all three approaches in Section 5: The Q-function based approach, the DP update based approach, and the witness theorem based approach. As in the case of IRL for MDP\R, we were able to control the sparseness in the reward function by tuning the penalty weight λ. With a suitable value for λ, all three approaches yielded the same reward function.[6]

A summary of the experiments is given in Table 6. Since the *Heaven/Hell* problem has a larger number of observations than other problems and the Q-function and the DP update based approaches generate exponentially many new policies with respect to the number of observations, the optimization problems of the Q-function and the DP update based approaches were not able to handle the *Heaven/Hell* problem. Hence, the *Heaven/Hell* problem could only be solved by the witness theorem based approach. Also, the witness theorem based approach was able to solve the other problems more efficiently than the Q-function based approach and the DP update based approach.

6. With any value of λ, the reward functions computed by all the proposed optimization problems should guarantee the optimality of the expert's policy, except for the degenerated case $R = 0$ due to an overly large value of λ. However, we observed that the optimality of our solutions is often subject to numerical errors in the optimization, which is an interesting issue for future studies.

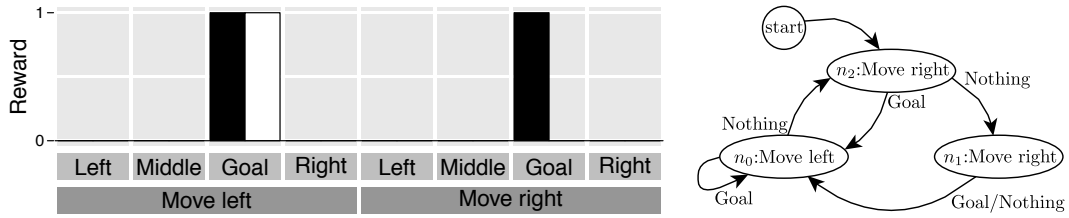


Figure 6: Comparison of the true and learned reward functions and the expert's policy in the *1d Maze* problem. *Black bars*: The true reward function. *White bars*: The learned reward function.

In Table 6, $D(R^*) = |V^{\pi_E}(b_0;R^*) - V^{\pi_L}(b_0;R^*)|$ is the difference between the values of the expert's policy π_E and the optimal policy π_L for the learned reward, which are measured on the true reward function R^*. $D(R_L) = |V^{\pi_E}(b_0;R_L) - V^{\pi_L}(b_0;R_L)|$ is the difference between the values measured on the learned reward function R_L. The differences measured on the true reward function in the *Tiger*, *1d Maze*, 5×5 *Grid World*, and *Heaven/Hell* are zero, meaning that the learned reward function generated a policy whose performance is the same as that of the expert's policy. However, our algorithms failed to find the reward that generates a policy that is optimal on the true reward in the *Rock Sample*[4,3]. Nevertheless, we can say that the learned reward function R_L satisfies the optimality of the expert's policy π_E since the policy π_L is an optimal policy on the learned reward function R_L and $|V^{\pi_E}(b_0;R_L) - V^{\pi_L}(b_0;R_L)| = 0$. Thus, the reason for our algorithms' failure in the *Rock Sample*[4,3] might be that the objective functions in the optimization problems are not well formulated to choose an appropriate reward function that yields a policy similar to the expert's, among the infinitely many reward functions in the space specified by the constraints of the optimization problems.

We further discuss the details of the results from each problem below. The learned reward functions are compared to the true reward functions for the *Tiger*, *1d Maze*, 5×5 *Grid World*, and *Heaven/Hell* problems, but the reward function in the *Rock Sample*[4,3] problem is omitted since it has too many elements to present.

In the *Tiger* problem, the true and learned reward functions are respectively represented as R^* and R_1 in Table 5. The true reward function is not sparse. Every action is associated with a non-zero reward. Since our methods favor sparse reward functions, there is some degree of difference between the true and the learned reward functions, most notably for the listen action, where our methods assign a zero reward instead of -1 as in the true reward. However, we can apply the policy-invariant reward transformation (Ng et al., 1999) on the learned reward function so that listen action yields -1 reward. R_1 is the transformed learned reward function. It is close to the true reward function and produces the optimal policy whose value is equal to the value of the expert's policy when measured on the true reward function.

For the *1d Maze* problem, the learned reward function is compared to the true reward function in the left panel of Figure 6 and the expert's policy is presented in the right panel of Figure 6. The expert's policy has three nodes: Node n_2 (the starting node) chooses to move right, and changes to node n_1 upon observing *Nothing* or to node n_0 upon observing *Goal*; node n_1 chooses to move right and always changes to node n_0; node n_0 chooses to move left, and changes to node n_2 upon observing *Nothing* or to itself upon observing *Goal*. Following the expert's policy, moving left is

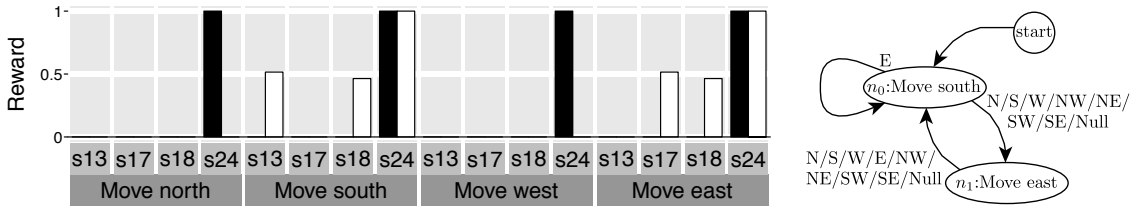


Figure 7: Comparison of the true and the learned reward functions and the expert's policy in the 5×5 *Grid World* problems. *Black bars*: The true reward. *White bars*: The learned reward.

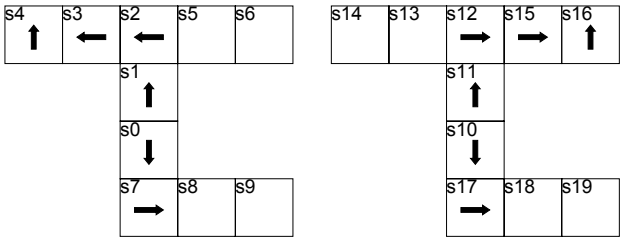


Figure 8: Learned reward function in the *Heaven/Hell* problem. *Black arrow*: $+1$ reward for moving in the direction of the arrow in each state. *Blank grid*: Zero reward for all actions in each state.

always executed after perceiving the goal state. This causes the algorithms to assign the positive reward to moving left in the goal state as the true one, but the zero reward to moving right in the goal state unlike the true one. Consequently, the algorithms find the reward function that explains the behavior of the expert's policy, and the optimal policy from the POMDP with respect to the learned reward function is the same as the expert's policy.

In the 5×5 *Grid World* problem, the expert's policy is simple as depicted in the right panel of Figure 7: The agent alternates moving south and east from the start, visiting the states in the diagonal positions (i.e., $\{s_0, s_5, s_6, s_{11}, s_{12}, s_{17}, s_{18}, s_{23}, s_{24}\}$ and $\{s_0, s_1, s_6, s_7, s_{12}, s_{13}, s_{18}, s_{19}, s_{24}\}$). The learned reward function is presented with the true reward function in the left panel of Figure 7. Our methods assign a small positive reward for moving south in states 13 and 18 and moving east in states 17 and 18. Also, the reward for moving south and east in state 24 is assigned to $+1$ for reaching the goal. The learned reward function closely reflects the behavior of the given expert' policy. Again, even though the learned reward function is different from the true one, it yields the same optimal policy.

Finally, in the *Heaven/Hell* problem, the true reward function is $+1$ for states 4 and 16 being heaven, and -1 for states 6 and 14 being hell. The learned reward is presented in Figure 8 where the agent gets a $+1$ reward when moving in the direction of the arrow in each state. The learned reward function exactly describes the behavior of the expert, which first visits the priest in states 9 and 19 starting from states 0 and 10 to acquire the position of heaven and then moves to heaven in states 4 and 16. As shown in Table 6, the learned reward function in the *Heaven/Hell* problem also yields the policy whose value is equal to that of the expert's policy.

7.2 Experiments on IRL from Sampled Trajectories

The second set of experiments involves the case when the expert's trajectories are given. We experimented on the same set of five problems with all three approaches in Section 6: the max-margin between values (MMV), the max-margin between feature expectations (MMFE), and the projection (PRJ) methods.

In this section, the reward function is assumed to be linearly parameterized with the basis functions and we prepare four sets of basis functions to examine the effect of the choice of basis functions on the performance of the algorithms:

- *Compact*: The set of basis functions that captures the necessary pairs of states and actions to present the structure of the true reward function. Let $F = \{F_0, F_1, \cdots, F_N\}$ be a partition of $S \times A$ such that $\forall (s,a) \in F_i$ have the same reward value $R(s,a)$. The *compact* basis functions for the partition F is defined such that the i-th basis function $\phi_i(s,a) = 1$ if $(s,a) \in F_i$ and $\phi_i(s,a) = 0$ otherwise.

- *Non-compact*: The set of basis functions that includes all the compact basis functions and some extra redundant basis functions. Each basis function ϕ_i is associated with some set of state-action pairs as above.

- *State-wise*: The set of basis functions that consists of the indicator functions for each state. The i-th basis function is defined as $\phi_i(s) = \delta_{s'}(s)$ if i-th state is s'.[7]

- *State-action-wise*: The set of basis functions consists of the indicator functions for each pair of state and action. The i-th basis function is defined as $\phi_i(s,a) = \delta_{(s',a')}(s,a)$ if i-th pair of state and action is (s',a').

For small problems, such as the *Tiger*, *1d Maze*, and 5×5 *Grid World* problems, we experimented with *state-action-wise* basis functions. For the two larger problems, three sets of basis functions are selected. For the *Heaven/Hell* problem, the first set consists of the *compact* set of basis functions. Table 7 shows the set F_i of pairs of states and actions for each basis function. The second set consists of the *state-wise* basis functions and the third set consists of the *state-action-wise* basis functions. For the *Rock Sample*[4,3] problem, the first set consists of the *compact* set of basis functions. The left side of Table 8 shows the set F_i of pairs of states and actions for each basis function. The second set consists of the *non-compact* set of basis functions including the redundant functions that present the rover's using its sensor (ϕ_{10}), moving on the map (ϕ_{11}), sampling at some locations without rocks (ϕ_{12}–ϕ_{15}), and sampling at the rest of the locations (ϕ_{16}). The right side of Table 8 presents the set of the pairs of states and actions for the *non-compact* basis functions. The third set consists of the *state-action-wise* basis functions.

For each experiment, we sampled 2000 belief trajectories. Each trajectory is truncated after a large finite number H of time steps. If we truncate the trajectories after $H_\varepsilon = \log_\gamma(\varepsilon(1-\gamma)/R_{max})$ time steps, the error in estimating the value would be no greater than ε. Table 9 shows the number of time steps for each problem.

As in the previous section, we compare $V^{\pi_L}(b_0; R^*)$ at each iteration, which is the value of the policy π_L from the learned reward function R_L evaluated on the true reward function R^*. The results are shown in Figure 9. All the algorithms found the reward function that generate the policy

7. Here, we use the Kronecker delta function, that is, $\delta_i(j) = 1$ if $i = j$, and $\delta_i(j) = 0$ if $i \neq j$.

F_i	States	Actions
F_0	s_4	$*$
F_1	s_{16}	$*$
F_2	s_6	$*$
F_3	s_{14}	$*$
F_4	$S \backslash \{s_4, s_6, s_{14}, s_{16}\}$	$*$

Table 7: Sets of state-action pairs for the *compact* set of basis functions in the *Heaven/Hell* problem. The states s_4 and s_{16} represent heaven and the states s_6 and s_{14} represent hell.

F_i	States	Actions
F_0	x=0	Move west
F_1	x=3	Move east
F_2	y=0	Move south
F_3	y=3	Move north
F_4	$\langle x,y \rangle = L_0, r_0 = true$	Sample
F_5	$\langle x,y \rangle = L_0, r_0 = false$	Sample
F_6	$\langle x,y \rangle = L_1, r_1 = true$	Sample
F_7	$\langle x,y \rangle = L_1, r_1 = false$	Sample
F_8	$\langle x,y \rangle = L_2, r_2 = true$	Sample
F_9	$\langle x,y \rangle = L_2, r_2 = false$	Sample
F_{10}	$\langle x,y \rangle \notin \{L_i, \forall i\}$	Sample
F_{11}	The remaining state-action pairs	

F_i	States	Actions
$F_0, \cdots, F_9$	Same as in the *compact* set	
F_{10}	$*$	Use the sensor
F_{11}	$*$	Move
F_{12}	$\langle x,y \rangle = L'_0$	Sample
F_{13}	$\langle x,y \rangle = L'_1$	Sample
F_{14}	$\langle x,y \rangle = L'_2$	Sample
F_{15}	$\langle x,y \rangle = L'_3$	Sample
F_{16}	$\langle x,y \rangle \notin \{L_i, \forall i, L'_j, \forall j\}$	Sample
F_{17}	The remaining state-action pairs	

Table 8: Sets of state-action pairs for the *compact* (*Left*) and *non-compact* set of basis functions (*Right*) in the *Rock Sample*[4,3] problem. $\langle x,y \rangle$ denotes the location of the rover. L_i is the location of i-th rock. L'_i is a randomly chosen location without rocks. r_i is the Boolean variable for representing whether the i-th rock is good or not.

close to the expert's policy in small problems, that is, the *Tiger*, *1d Maze*, and 5×5 *Grid World* problems. They also converged to the optimal value in a few iterations when using the *compact* set of basis functions in the two larger problems, that is, the *Heaven/Hell* and *Rock Sample*[4,3] problems. However, more iterations were required to converge when other sets of basis functions were used. This is due to the fact that a larger number of basis functions induces a larger search space. In the *Heaven/Hell* problem, the MMV method converged to a sub-optimal solution using the *state-wise* basis functions although the true reward function can be represented exactly using the *state-wise* basis functions. The MMV method had no such issues when using the *state-action-wise* basis functions. In the *Rock Sample*[4,3] problem, the MMV method also converged to a sub-optimal solution using the *state-action-wise* basis functions with 1024 basis functions, most of them being redundant since there are only 12 basis functions in the *compact* set. Hence, the MMV method is sensitive to the selection of basis functions, whereas the MMFE and PRJ methods robustly yield optimal solutions. Our reasoning on this phenomenon is given in the end of this subsection. Meanwhile, the value of the learned policies tends to oscillate in the beginning of the learning phase, particularly in the *Tiger* and *Rock Sample*[4,3] problems, since our methods are

Problem	# of steps	$V^{\pi_E}(b_0; R^*)$
Tiger	20	1.93
1d Maze	20	1.02
5×5 *Grid World*	50	0.70
Heaven/Hell	300	8.64
Rock Sample[4,3]	200	21.11

Table 9: Configuration for each problem and the value of the expert's policy measured on the true reward function.

Problem	ϕ	$\lvert\phi\rvert$	$V^{\pi_L}(b_0; R^*)$			Time		
			MMV	MMFE	PRJ	MMV	MMFE	PRJ
Tiger	SA	6	1.79	1.93	1.93	10.04 (72.27)	7.04 (41.56)	3.97 (96.33)
1d Maze	SA	8	1.02	1.02	1.02	0.88 (75.07)	5.18 (10.83)	0.71 (82.13)
5×5 *Grid World*	SA	100	0.70	0.70	0.70	24.10 (95.11)	20.07 (96.88)	21.49 (98.16)
Heaven/Hell	C	5	8.49	8.64	8.64	18.54 (63.02)	11.80 (79.75)	8.99 (88.66)
	S	20	5.70	8.64	8.64	375.03 (96.48)	332.97 (98.51)	937.59 (99.75)
	SA	80	8.47	8.64	8.64	443.57 (98.31)	727.87 (99.30)	826.37 (99.68)
Rock Sample[4,3]	C	11	20.84	20.05	20.38	8461.65 (99.16)	8530.18 (52.03)	10399.61 (59.86)
	NC	17	20.83	20.62	20.16	21438.83 (89.88)	10968.81 (25.05)	27808.79 (79.41)
	SA	1024	-26.42	17.83	19.05	31228.85 (72.45)	13486.41 (78.25)	16351.59 (80.57)

Table 10: Results of IRL for POMDP\R from sampled trajectories. The sets of the basis functions are denoted by C(*compact*), NC(*non-compact*), S(*state-wise*), and SA(*state-action-wise*). The average computation time for each trial is reported in seconds and the numbers in the parentheses next to the computation time are the percentages of the time taken by the POMDP solver.

not guaranteed to improve monotonically and are hence prone to yielding poor intermediate reward functions. However, these poor intermediate reward functions will effectively restrict the region of the reward functions for the final result.

We summarize $V^{\pi_L}(b_0; R^*)$ returned at the end of the algorithms and the computation time for each trials with the computation time for solving intermediate POMDPs in Table 10. As noted in the above, in most of the experiments, the algorithms eventually found the policy whose performance is the same as the expert's, which means the algorithms found the reward function that successfully recovers the expert's policy. The computation time increased when the size of basis functions and the size of the problems were increased. When the *state-action-wise* basis functions were applied for the *Rock Sample*[4,3] problem, it took about 8 hours on average for the MMV method to converge. However, the larger portion of the computation time was spent for solving intermediate POMDPs. The average percentage of the time spent for solving intermediate POMDPs was 78.83%.

The third set of experiments was conducted for examining the performance of the algorithms as the number of sampled belief trajectories varied. We experimented with the MMV, MMFE, and PRJ methods in the *Tiger* problem. Each trajectory was truncated after 20 time steps. Figure 10 presents the results where the value of policy is measured by $V^{\pi_L}(b_0; R^*)$. The MMFE method required fewer number of trajectories to attain the policy that performs close to the expert's than the MMV and PRJ methods required. The performance of the PRJ method was the worst when given

few sampled trajectories, but it improved fast as the number of trajectories increased. However, the MMV method needed many trajectories to find the near-optimal solution.

We conclude this subsection with our reasoning on why the MMFE and PRJ methods typically outperform the MMV method. The MMFE and PRJ methods directly use the differences in feature expectations (line 11 in Algorithm 5 and line 14 in Algorithm 6), whereas the MMV method uses the differences in values obtained from the *weight vectors* and feature expectations (line 12 in Algorithm 4). Using the differences in values can be problematic because it is often possible that a weight vector very different from the true one can yield a very small difference in values. Hence, it is preferable to directly use the differences in feature expectations since it still bounds the differences in values without depending on the weight vectors.

8. Related Work

In control theory, recovering a reward function from demonstrations has received significant attention, and has been referred to as the inverse optimal control (IOC) problem. It was first proposed and studied for linear systems by Kalman (1964). IRL is closely related to IOC, but the focus is on the problem of inverse optimality within the framework of RL. As already mentioned in the introduction, Russell (1998) proposed IRL as an important problem in machine learning, suggesting that it will be useful in many research areas such as studies on animal and human behaviors since the reward function reflects the objective and the preference of the decision maker. IRL is also useful for reinforcement learning since similar but different domains often share the same reward function structure albeit different dynamics. In this case, transferring the reward function learned from one domain to another domain may be useful.

Besides the task of *reward learning*, IRL has gained interest in *apprenticeship learning*, where the task is to find the policy with possibly better performance than the one demonstrated by an expert. Apprenticeship learning is useful when explicitly specifying the reward function is difficult but the expert's behaviors are available instead. Apprenticeship learning is a promising approach in robotics since it provides a framework for a robot to imitate the demonstrator without a full specification of which states are good or bad, and to what degree.

Since Russell (1998), a number of algorithms for IRL and apprenticeship learning have been proposed in the last decade. Most of the algorithms assume a completely observable setting, where the agent has capability to access the true global state of the environment often modeled as an MDP. In this section, we briefly review some of these previous works on the IRL and apprenticeship learning problem.

One of the first approaches to IRL in the MDP setting was proposed by Ng and Russell (2000), which we have covered in Section 3. They presented a sufficient and necessary condition on the reward functions which guarantees the optimality of the expert's policy, and provided some heuristics to choose a reward function since the degenerate reward functions also satisfy the optimality condition. The IRL problem was formulated as LP with the constraints corresponding to the optimality condition and the objective function corresponding to the heuristics. The algorithm was shown to produce reasonably good solutions in the experiments on some benchmark problems. We have extended this algorithm to the partially observable setting in Section 5 and Section 6.1.

Abbeel and Ng (2004) presented an apprenticeship learning algorithm based on IRL, which we have described in Section 3.2. One of the important aspects of the algorithm was to compare the feature expectations between the expert's and the learned policies rather than the estimated values.

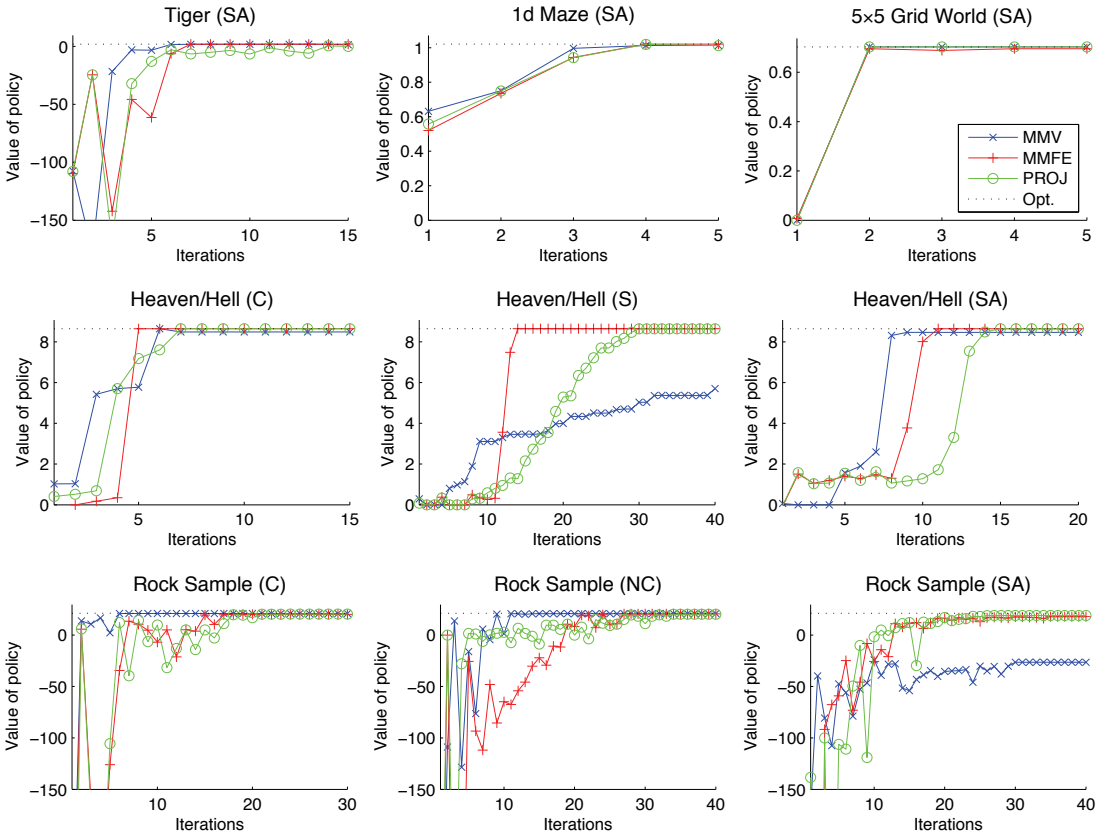


Figure 9: The value of the policies produced by the learned reward function at each iteration by the algorithms of IRL for POMDP\R from sampled trajectories. The value is measured on the true reward function for each problem. The optimal value is denoted by *Opt.* in the legend.

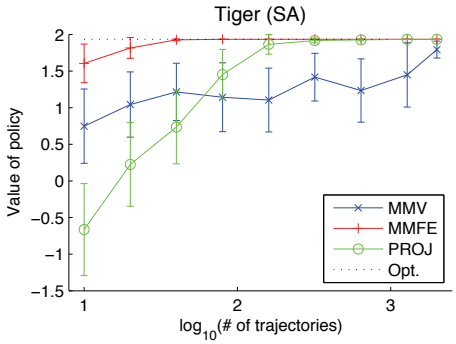


Figure 10: The value of the policies produced by the learned reward function by the algorithms of IRL for POMDP\R from varying number of sampled trajectories. Averages over 100 trials are presented with 95% confidence intervals. The x-axis represents the number of sampled trajectories on a log 10 scale.

The algorithm comes with a theoretical guarantee that the learned policy is similar to the expert's policy when evaluated on the true reward function. The algorithm was shown to successfully learn different driving styles in a simulated car driving task. This work was further extended using a number of different approaches. We have extended this algorithm to the partially observable setting in Section 6.2 and Section 6.3.

The structured max-margin optimization technique (Taskar et al., 2005) was applied to apprenticeship learning by Ratliff et al. (2006). They formulated a QP problem to find the weight vector of the reward basis functions that maximizes the margin between the expert's policy and all other policies. They also provided the maximum margin planning (MMP) algorithm based on the subgradient method, which is faster than the QP method. The MMP was shown to solve problems of practical sizes, such as route planning for outdoor mobile robots, where the QP method was not applicable.

Neu and Szepesvari (2007) proposed an algorithm for apprenticeship learning that unifies the direct and indirect methods: The direct method, using supervised learning methods, finds the policy that minimizes loss functions that penalize deviating from the expert's policy. The indirect method finds the policy using the learned reward function from IRL. Since the loss functions are defined on the policy space, the algorithm uses natural gradients to map the gradients in the policy space to those in the weight vector space of reward functions.

Whereas most of the apprenticeship learning algorithms focus on approximating the performance of the expert's policy, Syed and Schapire (2008) proposed a method called multiplicative weights for apprenticeship learning (MWAL), which tries to improve on the expert's policy. This was achieved in a game-theoretic framework using a two person zero-sum game, where the learner selects a policy that maximizes its performance relative to the expert's and the environment adversarially selects a reward function that minimizes the performance of the learned policy. The game was solved using the multiplicative weights algorithm (Freund and Schapire, 1999) for finding approximately optimal strategies in zero-sum games.

One of the difficulties in apprenticeship learning is that most proposed algorithms involve solving MDPs in each iteration. Syed et al. (2008) addressed this issue by identifying the optimization performed in the MWAL algorithm, and formulating it into an LP problem. They showed that this direct optimization approach using an off-the-shelf LP solver significantly improves the performance in terms of running time over the MWAL algorithm.

As mentioned in Section 4, IRL is an ill-posed problem since the solution of IRL is not unique. To address the non-uniqueness in the solution, the above approaches adopt some heuristics, for example, maximizing the margin between the expert's policy and other policies. We could also handle the uncertainty in the reward function using probabilistic frameworks. Ramachandran and Amir (2007) suggested a Bayesian framework for IRL and apprenticeship learning. The external knowledge about the reward function is formulated in the prior, and the posterior is computed by updating the prior using the expert's behavior data as evidence. Ziebart et al. (2008) proposed an apprenticeship learning algorithm adopting the maximum entropy principle for choosing the learned policy constrained to match feature expectations of the expert's behavior.

Recently, Neu and Szepesvari (2009) provided a unified framework for interpreting a number of incremental IRL algorithms listed above, and discussed the similarities and differences among the algorithms by defining the distance function and the update step employed in each algorithm. Each algorithm was characterized by the distance function that measures the difference between the expert's behavior data and the policy from the learned reward function, and the update step that computes new parameter values for the reward function.

The question of whether the IRL and the apprenticeship learning algorithms listed above can be extended to the partially observable setting in an efficient way remains as an important open problem.

9. Conclusion

The objective of IRL is to find the reward function that the domain expert is optimizing from the given data of her or his behavior and the model of the environment. IRL will be useful in various areas connected with reinforcement learning such as animal and human behavior studies, econometrics, and intelligent agents. However, the applicability of IRL has been limited since most of the previous approaches employed the assumption of an omniscient agent using the MDP framework.

We presented an IRL framework for dealing with partially observable environments in order to relax the assumption of an omniscient agent in the previous IRL algorithms. First, we derived the constraints of the reward function to guarantee the optimality of the expert's policy and built optimization problems to solve IRL for POMDP\R when the expert's policy is explicitly given. The results from the classical POMDP research, such as the generalized Howard's policy improvement theorem (Howard, 1960) and the witness theorem (Kaelbling et al., 1998), were exploited to reduce the computational complexity of the algorithms. Second, we proposed iterative algorithms of IRL for POMDP\R from the expert's trajectories. We proposed an algorithm that uses max-margin between values via LP, and then, in order to address larger problems robustly, we adapted the algorithms for apprenticeship learning in the MDP framework to IRL for POMDP\R. Experimental results on several POMDP benchmark domains showed that, in most cases, our algorithms robustly find solutions close to the true reward function, generating policies that acquire values close to that of the expert's policy.

We demonstrated that the classical IRL algorithm on MDP\R could be extended to POMDP\R, and we believe that more recent IRL techniques as well as some of the IRL-based apprenticeship learning techniques could be similarly extended by following our line of thought. However, there are a number of interesting issues that should be addressed in future studies.

9.1 Finding the Optimality Condition

The proposed conditions in Section 5 are not sufficient conditions of the reward function to guarantee the optimality of the expert's policy. The condition based on the comparison of Q-functions in Equation (14) should be evaluated for every possible policy that may have an infinite number of nodes. The condition using the DP update and the witness theorem in Equation (16) should be evaluated for some useful nodes that the expert's policy may not have due to their unreachability from the starting node. Also, Equations (14) and (16) should be extended to assess the value for all beliefs. Thus, it is crucial to find a sufficient condition that can be efficiently computed in order to restrict the feasible region of the reward functions tightly so that the optimization problems can find the reward function that guarantees the optimality of the given expert's policy.

9.2 Building an Effective Heuristic

Although the constraints for the reward function are not sufficient conditions, we empirically showed that $|V^{\pi_E}(b_0;R_L) - V^{\pi_L}(b_0;R_L)| = 0$, which implies that the value of the expert's policy π_E is equal to that of the optimal policy π_L produced by the learned reward R_L when the value is evaluated on

the learned reward. In other words, the expert's policy is another optimal policy for the learned reward and the learned reward still satisfies the optimality condition of the expert's policy. However, the optimal policy for the learned reward does not achieve the same value as the expert's policy when the value is evaluated on the true reward. The reason for the algorithms' failure to find the appropriate reward function may lie in the shortcomings of the heuristic for the objective functions. In this paper, we use the heuristic originally proposed by Ng and Russell (2000). It prefers the reward function that maximizes the sum of the differences between the value of the expert's policy and the other policies while forcing the reward function to be as sparse as possible. Unfortunately, this heuristic failed in some cases in our experiments. Hence, a more effective heuristic should be devised to find the reward function that provides similar behavior to the expert's policy. This can be addressed by adapting more recent IRL approaches such as the Bayesian IRL (Ramachandran and Amir, 2007) and the maximum entropy IRL (Ziebart et al., 2008) to partially observable environments. The Bayesian IRL prefers the reward function inducing the high probability of executing actions in the given behavior data, and the maximum entropy IRL prefers the reward function maximizing the entropy of the distribution over behaviors while matching the feature expectations.

9.3 Scalability

The algorithms we presented are categorized into two sets: The first is for the cases when the expert's policy is explicitly given in the FSC representations and the second is for the cases when the trajectories of the expert's executed actions and the corresponding observations are given. For the first set of the algorithms, the computational complexity is reduced based on the generalized Howard's policy improvement theorem (Howard, 1960) and the witness theorem (Kaelbling et al., 1998). The algorithms still suffer from a huge number of constraints in the optimization problem. The question is then whether it is possible to select a more compact set of constraints that define the valid region of the reward function while guaranteeing the optimality of the expert's policy, which is again related to finding the sufficient condition. For the second set of the algorithms, the scalability is more affected by the efficiency of the POMDP solver than by the number of constraints in the optimization problem. Although PBPI (Ji et al., 2007), the POMDP solver used in this paper, is known to be one of the fastest POMDP solvers which return FSC policies, it was observed in the experiments that the algorithms spent more than 95% of the time to solve the intermediate POMDP problems. Computing an optimal policy in the intermediate POMDP problem takes a much longer time than solving a usual POMDP problem, since an optimal policy of the intermediate POMDP problem is often complex due to the complex reward structure. The limitation could be handled by modifying the algorithms to address the IRL problems with other POMDP solvers, such as HSVI (Smith and Simmons, 2005), Perseus (Spaan and Vlassis, 2005), PBVI (Pineau et al., 2006), and SARSOP (Kurniawati et al., 2008), which generate the policy defined as a mapping from beliefs to actions.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant 2009-0069702, and by the Defense Acquisition Program Administration and Agency for Defense Development of Korea under contract 09-01-03-04.

References

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, pages 1–8, Banff, Alta, Canada, 2004.

Tilman Borgers and Rajiv Sarin. Naive reinforcement learning with endogenous aspirations. *International Economic Review*, 41(4):921–950, 2000.

Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI)*, pages 1023–1028, Seattle, WA, USA, 1994.

Jaedeug Choi and Kee-Eung Kim. Inverse reinforecement learning in partially observable environments. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1028–1033, Pasadena, CA, USA, 2009.

Michael X Cohen and Charan Ranganath. Reinforcement learning signals predict future decisions. *Journal of Neuroscience*, 27(2):371–378, 2007.

Ido Erev and Alvin E. Roth. Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *American Economic Review*, 88(4): 848–881, 1998.

Yoav Freund and Robert E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29(1–2):79–103, 1999.

Hector Geffner and Blai Bonet. Solving large POMDPs using real time dynamic programming. In *Proceedings of the AAAI Fall Symposium Series*, pages 61–68, 1998.

Jacques Hadamard. Sur les problemes aux derivees partielles et leur signification physique. *Princeton University Bulletin*, 13(1):49–52, 1902.

Eric A. Hansen. *Finite-Memory Control of Partially Observable Systems*. PhD thesis, University of Massachusetts Amherst, 1998.

Jesse Hoey, Axel Von Bertoldi, Pascal Poupart, and Alex Mihailidis. Assisting persons with dementia during handwashing using a partially observable Markov decision process. In *Proceedings of the 5th International Conference on Vision Systems*, Bielefeld University, Germany, 2007.

Ed Hopkins. Adaptive learning models of consumer behavior. *Journal of Economic Behavior and Organization*, 64(3–4):348–368, 2007.

Ronald. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA, 1960.

Shihao Ji, Ronald Parr, Hui Li, Xuejun Liao, and Lawrence Carin. Point-based policy iteration. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1243–1249, Vancouver, BC, USA, 2007.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

Rudolf E. Kalman. When is a linear control system optimal? *Transaction ASME, Journal Basic Engineering*, 86:51–60, 1964.

Hanna Kurniawati, David Hsu, and Wee Sun Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Proceedings of Robotics: Science and Systems*, Zurich, Switzerland, 2008.

Terran Lane and Carla E. Brodley. An empirical study of two approaches to sequence learning for anomaly detection. *Machine Learning*, 51(1):73–107, 2003.

Daeyeol Lee, Michelle L. Conroy, Benjamin P. McGreevy, and Dominic J. Barraclough. Reinforcement learning and decision making in monkeys during a competitive game. *Cognitive Brain Research*, 22(1):45–58, 2004.

George E. Monahan. A survey of partially observable Markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

P. Read Montague and Gregory S. Berns. Neural economics and the biological substrates of valuation. *Neuron*, 36(2):265–284, 2002.

Gergely Neu and Csaba Szepesvari. Apprenticeship learning using inverse reinforcement learning and gradient methods. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 295–302, Vancouver, BC, Canada, 2007.

Gergely Neu and Csaba Szepesvari. Training parsers by inverse reinforcement learning. *Machine Learning*, pages 1–35, 2009.

Allen Newell. The knowledge level. *Artificial Intelligence*, 18(1):87–127, 1982.

Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 663–670, Stanford University, Standord, CA, USA, 2000.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 278–287, Bled, Slovenia, 1999.

Yael Niv. Reinforcement learning in the brain. *Journal of Mathematical Psychology*, 53(3):139–154, 2009.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006.

Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2586–2591, Hyderabad, India, 2007.

Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 729–736, Pittburgh, Pennsylvania, USA, 2006.

Stuart Russell. Learning agents for uncertain environments (extended abstract). In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*, pages 101–103, Madison, WI, USA, 1998.

Trey Smith. *Probabilistic planning for robotic exploration*. PhD thesis, Carnegie Mellon University, The Robotics Institute, 2007.

Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 520–527, Banff, Canada, 2004.

Trey Smith and Reid Simmons. Point-based POMDP algorithms: Improved analysis and implementation. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 542–547, Edinburgh, Scotland, 2005.

Edward Jay Sondik. *The Optimal Control of Partially Observable Markov Processes*. PhD thesis, Stanford University, 1971.

Matthijs T. J. Spaan and Nikos Vlassis. A point-based pomdp algorithm for robot planning. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2399–2404, New Orleans, LA, USA, 2004.

Matthijs T. J. Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

Umar Syed and Robert E. Schapire. A game-theoretic approach to apprenticeship learning. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 1449–1456, Vancouver, British Columbia, Canada, 2008.

Umar Syed, Michael Bowling, and Robert Schapire. Apprenticeship learning using linear programming. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1032–1039, Helsinki, Finland, 2008.

Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: a large margin approach. In *Proceedings of the 22nd International Conference on Machine learning*, pages 896–903, Bonn, Germany, 2005.

Jason D. Williams and Steve Young. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422, 2007.

Qing Zhao, Lang Tong, Ananthram Swami, and Yunxia Chen. Decentralized cognitive mac for opportunistic spectrum access in ad hoc networks: A pomdp framework. *IEEE Journal on Selected Areas in Communications*, 25(3):589–600, 2007.

Brian D. Ziebart, Andrew Maas, James Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1433–1438, Chicago, IL, USA, 2008.

Journal of Machine Learning Research 12 (2011) 731-817 Submitted 8/09; Revised 11/10; Published 3/11

Information, Divergence and Risk for Binary Experiments

Mark D. Reid MARK.REID@ANU.EDU.AU
Robert C. Williamson BOB.WILLIAMSON@ANU.EDU.AU
Australian National University and NICTA
Canberra ACT 0200, Australia

Editor: Yoram Singer

Abstract

We unify f-divergences, Bregman divergences, surrogate regret bounds, proper scoring rules, cost curves, ROC-curves and statistical information. We do this by systematically studying integral and variational representations of these objects and in so doing identify their representation primitives which all are related to cost-sensitive binary classification. As well as developing relationships between generative and discriminative views of learning, the new machinery leads to tight and more general surrogate regret bounds and generalised Pinsker inequalities relating f-divergences to variational divergence. The new viewpoint also illuminates existing algorithms: it provides a new derivation of Support Vector Machines in terms of divergences and relates maximum mean discrepancy to Fisher linear discriminants.

Keywords: classification, loss functions, divergence, statistical information, regret bounds

1. Introduction

Some of the simplest machine learning problems concern binary experiments. There it is assumed that observations are drawn from a mixture of two distributions (one for each class). These distributions determine many important objects related to the learning problems they underpin such as risk, divergence and information. Our aim in this paper is to present all of these objects in a coherent framework explaining exactly how they relate to each other. Doing so brings conceptual clarity to the area as well as providing the means for a number of new technical results.

1.1 Motivation

There are many different notions that underpin the definition of machine learning problems. These include information, loss, risk, regret, ROC (Receiver Operating Characteristic) curves and the area under them, Bregman divergences and distance or divergence between probability distributions. On the surface, the problem of estimating whether two distributions are the same (as measured by, say, their Kullback-Leibler divergence) is different to the problem of minimisation of expected risk in a prediction problem. One goal of the present paper is to show how this superficial difference is indeed only superficial—deeper down they are the same problem and analytical and algorithmic insights for one can be transferred to the other.

Machine learning as an engineering discipline is still young.[1] There is no agreed language to describe machine learning problems (such is usually done with an informal mixture of English and

1. Bousquet (2006) has articulated the need for an agreed vocabulary, a clear statement of the main problems, and to "revisit what has been done or discovered so far with a fresh look".

©2011 Mark D. Reid and Robert C. Williamson.

mathematics). There is very little in the way of composability of machine learning solutions. That is, given the solution to one problem, use it to solve another. Of course one would like to not merely be able to do this, but to be certain what one might lose in doing so. In order to do that, one needs to be able to provide theoretical guarantees on how well the original problem will be solved by solving the surrogate problem. Related to these issues is the fact that there are no well understood *primitives* for machine learning. Indeed, what does that even mean? All of these issues are the underlying motivation for this paper.

Our long term goal (towards which this paper is but the first step) is to turn the field of machine learning into a more well founded engineering discipline with an agreed language and well understood composition rules. Our motivation is that until one can start building systems modularly, one is largely restricted to starting from scratch for each new problem, rather than obtaining the efficiency benefits of re-use.[2]

We are comparing *problems*, not solutions or algorithms. Whilst there have been attempts to provide a degree of unification at the level of algorithms (Altun and Smola, 2006), there are intrinsic limits to such a research program. The most fundamental is that (surprisingly) there is no satisfactory formal definition of what an algorithm really is Blass and Gurevich (2003), nor how two algorithms can be compared with a view to determining if they are the same (Blass et al., 2009).

We have started with binary experiments because they are simple and widely used. As we will show, by pursuing the high level research agenda summarised above, we have managed to unify all of the disparate concepts mentioned and furthermore have simultaneously simplified and generalised two fundamental results: Pinsker inequalities between f-divergences and surrogate regret bounds. The proofs of these new results rely essentially on the decomposition into primitive problems.

1.2 Novelty and Significance

Our initial goal was to present existing material in a unified way. We have indeed done that. In doing so we have developed new (and simpler) proofs of existing results. Additionally we have developed some novel technical results. The key ones are:

1. A link between the weighted integral representations for proper scoring rules and those for f-divergences which allows the transformation from one to the other (Theorem 10);
2. A unified derivation of the integral representations in terms of Taylor series showing their equivalence (Theorem 18);

2. Abelson et al. (1996) described the principles of constructing software with the aid of (Locke, 1690, Chapter 12, paragraph 1):

> The acts of the mind, wherein it exerts its power over simple ideas, are chiefly these three: (1) Combining several simple ideas into one compound one; and thus all *complex ideas* are made. (2) The second is bringing two ideas, whether simple or complex, together, and setting them by one another, so as to take a view of them at once, without uniting them into one; by which it gets all its *ideas of relations*. (3) The third is separating them from all other ideas that accompany them in their real existence; this is called abstraction: and thus all its *general ideas* are made

Modularity is central to computer hardware (Baldwin and Clark, 2006b,a) and other engineering disciplines (Gershenson et al., 2003) and plays a central role in some models of economic development (Varian, 2003; Weitzman, 1998; Mokyr, 1992). The reason modularity works is that components can be combined or *composed*.

3. Use of these representations to derive new bounds for divergences, Bayes risks and regrets: "surrogate regret bounds"(Theorem 25) and Pinsker inequalities (Theorem 30);
4. Showing that statistical information (and hence f-divergence) are both Bregman informations;
5. The derivation of SVMs from a variational perspective which provides a clearer explanation of the link between MMD (Maximum Mean Discrepancy) and SVMs (Support Vector Machines) §H;
6. Explicit formulae relating Bayes risk to the Neyman-Pearson function, which allows the transformation of risk curves to ROC curves and vice versa (Theorem 22).

The significance of these new connections is that they show that the choice of loss function (scoring rule), f-divergence and Bregman divergence (regret) are intimately related—choosing one implies choices for the others. Furthermore we show there are more intuitively usable parameterisations for f-divergences and scoring rules (their corresponding weight functions). The weight functions have the advantage that if two weight functions match, then the corresponding objects are identical. That is not the case for the f parameterising an f-divergence or the convex function parameterising a Bregman divergence. As well as the theoretical interest in such connections, these alternate representations suggest new algorithms for empirically estimating such quantities. We have represented all of the connections graphically in figure 1. The various symbols are defined below; the point of the picture here is to see the overall goal of the paper—the relating of a range of diverse concepts.

Given the broad scope of our work, there is of course much prior work, too much to summarise in this introduction. Appendix C summarises the main precursors and related work.

1.3 Paper Outline and Key Contributions

The following is an outline of the main structure of this paper section by section highlighting the contributions and novelty. A knowledgeable reader only interested in the core new results should be able to just read Sections 4–8 plus Appendix H with the aid of Table 1. More tedious and technical proofs and digressions are in the appendices.

§2 Many of the properties of the objects studied in this paper are directly derived from well-known properties of convex functions. In particular, a generalised form of Taylor's theorem and Jensen's inequality underpin many of the new results. Although elementary, we have started from this point because it shows how fundamental are the connections drawn later in the paper are. We rederive Savage's famous theorem (Theorem 7) from our perspective.

§3 One of the simplest type of statistical problems is that of distinguishing between two distributions. Such a problem is known as a *binary experiment*. Two classes of *measures of divergence* between the distributions are introduced: the class of Csiszár f-divergences and the class of Bregman divergences.

§4 When additional assumptions are made about a binary experiment—specifically, a prior probability for each of the two distributions—it becomes possible to talk about *risk and statistical information* of an experiment that is defined with respect to a loss function. A key result is Theorem 10 which shows that f-divergence, statistical information and Bregman divergence are all fundamentally equivalent.

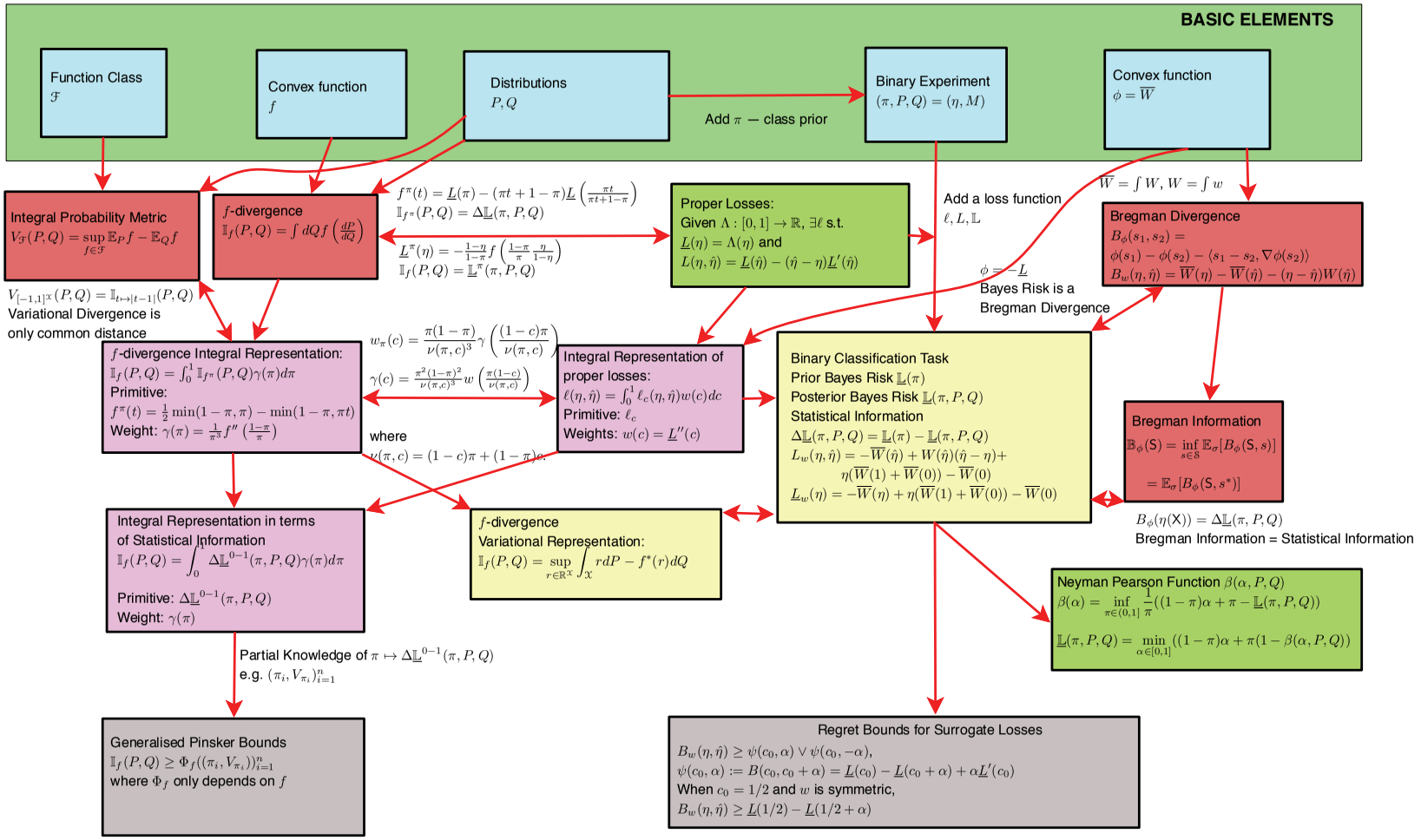


Figure 1: Diagrammatic summary of key relationships developed in the paper.

§5 A key technique we use is that of an integral representation. We show that integral representations of f-divergences and proper losses and statistical information are all essentially the same (Theorem 18). We explicitly compare the primitives for each of these representations and show their natural interpretation.

§6 The weight function view also illuminates various "graphical representations" of binary experiments, such as ROC curves. We unify several graphical representations for binary experiments and present new explicit formulae relating Bayes risk to the Neyman-Pearson function, which allows the transformation of risk curves to ROC curves and vice versa (Theorem 22).

§7 The various equivalences developed in the above sections are then used to derive new tight inequalities of interest in Machine Learning, The first is a We derive an explicit form for surrogate regret bounds for proper losses in terms of the weight function corresponding to the proper loss (Theorem 25). These are tight bounds on the conditional risk with respect to an arbitrary cost-sensitive misclassification loss when all is known is the value of the conditional risk with respect to an arbitrary proper loss. The result generalises existing results in two key ways. We also generalise the classical Pinsker inequality by deriving tight bounds on an arbitrary f-divergence when the value of several generalised variational divergences between the same distributions is known (Theorem 30). A side-effect is an explicit formula for the best possible bound on KL-divergence given knowledge of the classical variational divergence.

§8 Another representation of risks is a variational one. We systematically explore the relationship between Bayes risk and variational divergence, building upon classical results. An interesting consequence of our analysis is presented in Appendix H where we show that maximum mean discrepancy (MMD)—a kernel approach to hypothesis testing and divergence estimation—is essentially SVM learning in disguise. In doing so we present a novel, simple and interesting alternate derivation of the Support Vector Machine.

1.4 Notational Conventions

Here we record elementary notation and the conventions we adopt throughout the paper. Key notations are tabulated in table 1. We write $x \wedge y := \min(x,y)$, $x \vee y := \max(x,y)$, $(x)_+ := x \vee 0$, $(x)_- := x \wedge 0$ and the Iverson bracket $[\![p]\!] = 1$ if p is true and $[\![p]\!] = 0$ otherwise (Knuth, 1992). The generalised function $\delta(\cdot)$ is defined by $\int_a^b \delta(x) f(x) dx = f(0)$ when f is continuous at 0 and $a < 0 < b$ (Antosik et al., 1973; Friedlander, 1982). The unit step $U(x) = \int_{-\infty}^{x} \delta(t) dt$. The real numbers are denoted $\mathbb{R}$, the non-negative reals $\mathbb{R}^+$ and the extended reals $\overline{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$; the rules of arithmetic with extended real numbers and the need for them in convex analysis are explained by Rockafellar (1970). Random variables are written in sans-serif font: $\mathsf{S}, \mathsf{X}, \mathsf{Y}$. Sets are in calligraphic font: $\mathcal{X}$ (the "input" space), $\mathcal{Y}$ (the "label" space). Vectors are written in bold font: $\mathbf{a}, \boldsymbol{\alpha}, \boldsymbol{x} \in \mathbb{R}^m$. We will often have cause to take expectations ($\mathbb{E}$) of various functions over the random variable X. We write such quantities in blackboard bold: $\mathbb{I}, \mathbb{L}, \mathbb{B}, \mathbb{J}$. The elementary loss is ℓ, its conditional expectation w.r.t. Y is L and the full expectation (over the joint distribution $\mathbb{P}$ of (X, Y)) is $\mathbb{L}$. Lower bounds on quantities with an intrinsic lower bound (e.g., the Bayes optimal loss) are written with an underbar: $\underline{L}, \underline{\mathbb{L}}$. Quantities related by double integration appear in this paper and we notate the starting point in lower case, the first integral with upper case, and the second integral in upper case with an overbar: $w, W, \overline{W}$. Estimated quantities are hatted: $\hat{\eta}$. In several places we overload the notation. In all cases careful attention to the type of the arguments or subscripts reliably disambiguates.

Symbol	Meaning	Defined
I_ϕ	Perspective transform	(1)
(P,Q)	Binary experiment	§3
ℓ	Loss	§4.2
L	Conditional risk	§4.2
$\mathbb{L}$	Expected risk	§4.2
$\underline{L}$	Conditional Bayes risk	§4.2
$\underline{\mathbb{L}}$	Expected Bayes Risk	§4.2
$\mathbb{J}_\mu[\phi]$	Jensen gap	Th. 5
$\mathbb{I}_f(P,Q)$	f-divergence between P and Q	§3.2
$\phi^\diamondsuit$	Csiszár dual of ϕ	(2)
$\phi^\star$	Legendre-Fenchel dual of ϕ	(3)
B_ϕ	Bregman divergence and regret	§4.4
$\mathrm{TP}_r, \mathrm{FN}_r$	True Positive / False Negative rate for test r	(10)
$\beta(\cdot,P,Q)$	Neyman-Pearson function for (P,Q)	(11)
r,τ	Test, Test statistic	§3.1
$\mathbb{B}_\phi(P,Q)$	Generative Bregman divergence	§3.3
$\mathbb{P}$	Joint distribution on $\mathcal{X}\times\mathcal{Y}$	§4.1
M	Reference measure for (P,Q) with prior π	§4.1
π	A priori probability of positive class	§4.1
η	Probability of positive class	§4.2
$\eta(\cdot)$	Conditional probability of positive class	§4.2
$T=(\eta,M;\ell)=(\pi,P,Q;\ell)$	Task	§4.2
$\hat{\eta}(\cdot)$	Estimator of $\eta(\cdot)$	§4.2
$\mathbb{B}_\phi(\mathsf{S})$	Bregman information of S	§4.5
$w(\cdot)$	Weight function for proper loss	§5.3
$\gamma(\cdot)$	Weight function for f-divergence	§5.1
$\Delta\underline{\mathbb{L}}(\eta,M)$	Statistical information	(20)
ℓ_c, L_c	Cost-sensitive mis-classification loss	(29),(30)
$\mathrm{ROC}(\tau)$	Receiver Operating Characteristic curve	(37)
$\mathrm{AUC}(\tau)$	Area Under the ROC Curve	(38)
$V_\pi(P,Q)$	Generalised Variational divergence	(49)

Table 1: Standard notation used throughout the paper.

2. Convex Functions and Their Representations

Many of the properties of divergences and losses are best understood through properties of the convex functions that define them. One aim of this paper is to explain and relate various divergences and losses by understanding the relationships between their primitive functions. The relevant def-

initions and theory of convex functions will be introduced as required. Any terms not explicitly defined can be found in books by Hiriart-Urruty and Lemaréchal (2001) or Rockafellar (1970).

A set $\mathcal{S} \subseteq \mathbb{R}^d$ is said to be *convex* if for all $\lambda \in [0,1]$ and for all points $s_1, s_2 \in \mathcal{S}$ the point $\lambda s_1 + (1-\lambda)s_2 \in \mathcal{S}$. A function $\phi : \mathcal{S} \to \mathbb{R}$ defined on a convex set $\mathcal{S}$ is said to be a (proper) *convex function* if[3] for all $\lambda \in [0,1]$ and points $s_1, s_2 \in \mathcal{S}$ the function ϕ satisfies

$$\phi(\lambda s_1 + (1-\lambda)s_2) \leq \lambda\phi(s_1) + (1-\lambda)\phi(s_2).$$

A function is said to be *concave* if $-\phi$ is convex.

The remainder of this section presents properties, representations and transformations of convex functions that will be used throughout this paper.

2.1 The Perspective Transform and the Csiszár Dual

When $\mathcal{S} = \mathbb{R}^+$ and $\phi : \mathbb{R}^+ \to \overline{\mathbb{R}}$ is convex, the *perspective transform* of ϕ is defined for $\tau \in \mathbb{R}^+$ via

$$I_\phi(s,\tau) := \begin{cases} \tau\phi(s/\tau), & \tau > 0, s > 0 \\ 0, & \tau = 0, s = 0 \\ \tau\phi(0), & \tau > 0, s = 0 \\ s\phi'_\infty, & \tau = 0, s > 0, \end{cases} \tag{1}$$

where $\phi(0) := \lim_{s\to 0}\phi(s) \in \overline{\mathbb{R}}$ and ϕ'_∞ is the *slope at infinity* defined as

$$\phi'_\infty := \lim_{s\to+\infty} \frac{\phi(s_0+s)-\phi(s_0)}{s} = \lim_{s\to+\infty}\frac{\phi(s)}{s}$$

for every $s_0 \in \mathcal{S}$ where $\phi(s_0)$ is finite. This slope at infinity is only finite when $\phi(s) = O(s)$, that is, when ϕ grows at most linearly as s increases. When ϕ'_∞ is finite it measures the slope of the linear asymptote. The function $I_\phi : [0,\infty)^2 \to \overline{\mathbb{R}}$ is convex in both arguments (Hiriart-Urruty and Lemaréchal, 1993b) and may take on the value $+\infty$ when s or τ is zero. It is introduced here because it will form the basis of the f-divergences described in the next section.[4]

The perspective transform can be used to define the *Csiszár dual* $\phi^\diamondsuit : [0,\infty) \to \overline{\mathbb{R}}$ of a convex function $\phi : \mathbb{R}^+ \to \mathbb{R}$ by letting

$$\phi^\diamondsuit(\tau) := I_\phi(1,\tau) = \tau\phi\left(\frac{1}{\tau}\right) \tag{2}$$

for all $\tau \in (0,\infty)$ and $\phi^\diamondsuit(0) := \phi'_\infty$. The original ϕ can be recovered from I_ϕ since $\phi(s) = I_f(s,1)$.

The convexity of the perspective transform I_ϕ in both its arguments guarantees the convexity of the dual $\phi^\diamondsuit$. Some simple algebraic manipulation shows that for all $s,\tau \in \mathbb{R}^+$

$$I_\phi(s,\tau) = I_{\phi^\diamondsuit}(\tau,s).$$

This observation leads to a natural definition of symmetry for convex functions. We will call a convex function $\diamondsuit$-*symmetric* (or simply *symmetric* when the context is clear) when its perspective transform is symmetric in its arguments. That is, ϕ is $\diamondsuit$-symmetric when $I_\phi(s,\tau) = I_\phi(\tau,s)$ for all $s,\tau \in [0,\infty)$. Equivalently, ϕ is $\diamondsuit$-symmetric if and only if $\phi^\diamondsuit = \phi$.

3. The restriction of the values of ϕ to $\mathbb{R}$ will be assumed throughout unless explicitly stated otherwise. This implies the properness of ϕ since it cannot take on the values $-\infty$ or $+\infty$.
4. The perspective transform is closely related to *epi-multiplication* which is defined for all $\tau \in [0,\infty)$ and (proper) convex functions ϕ to be $\tau \otimes \phi := s \mapsto \tau\phi(s/\tau)$ for $\tau > 0$ and is 0 when $\tau = s = 0$ and $+\infty$ otherwise. Bauschke et al. (2008) summarise the properties of this operation and its relationship to other operations on convex functions.

2.2 The Legendre-Fenchel Dual Representation

A second important dual operator for convex functions is the *Legendre-Fenchel (LF) dual*. The LF dual $\phi^\star$ of a function $\phi : \mathcal{S} \to \mathbb{R}$ is a function defined by

$$\phi^\star(s^\star) := \sup_{s \in \mathcal{S}} \{\langle s, s^\star \rangle - \phi(s)\}. \tag{3}$$

The LF dual of any function is convex and, if the function ϕ is convex and closed then the *LF bidual* is a faithful representation of the original function. That is,

$$\phi^{\star\star}(s) = \sup_{s^\star \in \mathcal{S}^\star} \{\langle s^\star, s \rangle - \phi^\star(s^\star)\} = \phi(s).$$

When $\phi\colon \mathcal{S} \to \mathbb{R}$, $\mathcal{S} \subseteq \mathbb{R}$, is a function of a real argument s and the derivative $\phi'(s)$ exists, the Legendre-Fenchel conjugate $\phi^\star$ is given by the *Legendre transform* (Hiriart-Urruty and Lemaréchal, 2001; Rockafellar, 1970)

$$\phi^\star(s) = s \cdot (\phi')^{-1}(s) - \phi\left((\phi')^{-1}(s)\right).$$

2.3 Integral Representations

In this paper we are primarily concerned with convex and concave functions defined on subsets of the real line. A central tool in their analysis is the integral form of their Taylor expansion. Here, ϕ' and ϕ'' denote the first and second derivatives of ϕ respectively.

Theorem 1 (Taylor's Theorem) *Let $\mathcal{S} = [s_0, s]$ be a closed interval of $\mathbb{R}$ and let $\phi : \mathcal{S} \to \mathbb{R}$ be differentiable on $[s_0, s]$ and twice differentiable on (s_0, s). Then*

$$\phi(s) = \phi(s_0) + \phi'(s_0)(s - s_0) + \int_{s_0}^{s} (s - t)\, \phi''(t)\, dt. \tag{4}$$

The argument s appears in the limits of integral in the above theorem and consequently can be awkward to work with. Also, it will be useful to expand ϕ about some point not at the end of the interval of integration. The following corollary of Taylor's theorem removes these problems by introducing piecewise linear terms of the form $(s-t)_+ = (s-t) \vee 0$.

Corollary 2 (Integral Representation I) *Suppose $-\infty < a < b < \infty$ and let $\phi : [a, b] \to \mathbb{R}$ be a twice differentiable function. Then, for all $s, s_0 \in [a, b]$ we have*

$$\phi(s) = \phi(s_0) + \phi'(s_0)(s - s_0) + \int_{a}^{b} \phi_{s_0}(s, t)\, \phi''(t)\, dt, \tag{5}$$

where

$$\phi_{s_0}(s, t) := \begin{cases} (s - t) & s_0 < t \leq s \\ (t - s) & s < t \leq s_0 \\ 0 & \textit{otherwise} \end{cases}$$

is piecewise linear and convex in s for each $s_0, t \in [a, b]$.

This result is a consequence of the way in which ϕ_t effectively restricts the limits of integration to the interval $(s_0, s) \subseteq [a,b]$ or $(s, s_0) \subseteq [a,b]$ depending on whether $s_0 < s$ or $s_0 \geq s$ with appropriate reversal of the sign of $(s-t)$.

When $a = 0$ and $b = 1$ a second integral representation for the unit interval can be derived from (5) that removes the term involving ϕ'.

Corollary 3 (Integral Representation II) *A twice differentiable function $\phi : [0,1] \to \mathbb{R}$ can be expressed as*

$$\phi(s) = \phi(0) + (\phi(1) - \phi(0))s - \int_0^1 \psi(s,t)\,\phi''(t)\,dt, \tag{6}$$

where $\psi(s,t) = (1-t)s \wedge (1-s)t$ is piecewise linear and concave in $s \in [0,1]$ for each $t \in [0,1]$.

The result follows by integration by parts of $t\phi''(t)$. The proof can be found in Appendix A.1. It is used in Section 5 below to obtain an integral representation of losses for binary class probability estimation. This representation can be traced back to Temple (1954) who notes that the kernel $\psi(s,t)$ is the Green's function for the differential equation $\psi'' = 0$ with boundary conditions $\psi(a) = \psi(b) = 0$.

Both these integral representations state that the non-linear part of ϕ can be expressed as a weighted integral of piecewise linear terms ϕ_{s_0} or ψ. When we restrict our attention to convex ϕ we are guaranteed the "weights" $\phi''(t)$ for each of these terms are non-negative. Since the measures of risk, information and divergence we examine below do not depend on the linear part of these expansions we are able to identify convex functions with the weights $w(t) = \phi''(t)$ that define their non-linear part. The sets of piecewise linear functions $\{\phi_{s_0}(s,t)\}_{t\in[a,b]}$ and $\{\psi(s,t)\}_{t\in[0,1]}$ can be thought of as families of "primitive" convex functions from which others can be built through their weighted combination. Representations like these are often called *Choquet representations* after work by Choquet (1953) on the representation of compact convex spaces (Phelps, 2001).

2.4 Representations for Non-Differentiable Convex Functions

It is possible to weaken the conditions on the representation results so they hold for continuous but not necessarily differentiable functions. As much of this paper deals with functions that fall into this category—namely general convex functions—being able to generalise these results is essential in order to understand the weight functions corresponding to the primitive f-divergences and loss functions. We will briefly discuss these generalisations and introduce some conventions for interpreting subsequent results in an effort to avoid too many distracting technicalities.

The convention for the remainder of this paper is that the *first derivative* of a convex function ϕ over $\mathbb{R}$ is to be interpreted as a right derivative. That is, we will take $\phi'(t)$ to be $\phi'_+(t) := \lim_{\varepsilon \downarrow 0} \frac{\phi(t) - \phi(t+\varepsilon)}{\varepsilon}$. Theorem 24.1 of Rockafellar (1970) guarantees that this derivative exists and is non-decreasing and right continuous on the domain of ϕ. It is therefore possible to define a Lebesgue-Stieltjes measure $\lambda_\phi((a,b]) := \phi'(b) - \phi'(a)$ for intervals in the domain of ϕ.

Second derivatives of convex ϕ are only ever used within integrals to "weight" the contribution of the non-negative, piecewise linear functions $\phi_{s_0}(\cdot,t)$ and $\psi(\cdot,t)$ discussed above. Thus, we write $\int_a^b f(t)\,\phi''(t)\,dt$ as a short-hand for the Lebesgue-Stieltjes integral $\int_a^b f(t)\,d\lambda_\phi(t)$. For simplicity, we will often speak of weight "functions" being equal to the second derivative of general convex functions. As we only ever consider linear operators on these weight functions, it is unproblematic to treat second derivatives as Schwartz distributions or "generalised functions" (Antosik et al., 1973;

Friedlander, 1982) and add, scale, and evaluate them like normal functions. The most exotic of these we will consider explicitly are the weight functions corresponding to the primitive ϕ_{s_0} and ψ functions. They correspond to Dirac delta distributions $\delta(\cdot)$ as defined in Section 1.4.

As Liese and Vajda (2006) carefully show, it is possible to derive generalised versions of the integral representations using the interpretations above. Of course, when the functions ϕ are twice differentiable these interpretations and generalised results coincide with those for the usual first and second derivatives.

2.5 Bregman Divergence

Bregman divergences are a generalisation of the notion of distances between points. Given a differentiable[5] convex function $\phi : \mathcal{S} \to \mathbb{R}$ and two points $s_0, s \in \mathcal{S}$ the *Bregman divergence*[6] *of s from s_0* is defined to be

$$B_\phi(s, s_0) := \phi(s) - \phi(s_0) - \langle s - s_0, \nabla\phi(s_0) \rangle , \tag{7}$$

where $\nabla\phi(s_0)$ is the gradient of ϕ at s_0. A concise summary of many of the properties of Bregman divergences is given by Banerjee et al. (2005b, Appendix A); see also Censor and Zenios (1997). In particular, Bregman divergences always satisfy $B_\phi(s, s_0) \geq 0$ and $B_\phi(s_0, s_0) = 0$ for all $s, s_0 \in \mathcal{S}$, regardless of the choice of ϕ. They are not always metrics, however, as they do not always satisfy the triangle inequality and their symmetry depends on the choice of ϕ.

When $\mathcal{S} = \mathbb{R}$ and ϕ is twice differentiable, comparing the definition of a Bregman divergence in (7) to the integral representation in (4) reveals that Bregman divergences between real numbers can be defined as the non-linear part of the Taylor expansion of ϕ. Rearranging (4) shows that for all $s, s_0 \in \mathbb{R}$

$$\int_{s_0}^{s} (s - t)\, \phi''(t) dt = \phi(s) - \phi(s_0) - (s - s_0)\phi'(s_0) = B_\phi(s, s_0) \tag{8}$$

since $\nabla\phi = \phi'$ and the inner product is simply multiplication over the reals. This result also holds for more general convex sets $\mathcal{S}$. Importantly, it intuitively shows why the following holds (because the Bregman divergence depends only on the *nonlinear* part of the Taylor expansion).

Theorem 4 *Let ϕ and ψ both be real-valued, differentiable convex functions over the convex set $\mathcal{S}$ such that $\phi(s) = \psi(s) + as + b$ for some $a, b \in \mathbb{R}$. Then, for all s and s_0, $B_\phi(s, s_0) = B_\psi(s, s_0)$.*

A proof can be obtained directly by substituting and expanding ψ in the definition of a Bregman divergence.

Equation 8 also shows why $B(s, s_0)$ is decreasing as $|s - s_0|$ decreases (a fact we will exploit later): since $\phi''(t) \geq 0$ for all t, if $s_0 < s$, then the integrand in (8) is always non-negative and the result is immediate by the nature of integration. If $s_0 > s$, a similar argument holds.

2.6 Jensen's Inequality and the Jensen Gap

A central inequality in the study of convex functions is Jensen's inequality. It relates the expectation of a convex function applied to a random variable to the convex function evaluated at its mean. We will denote by $\mathbb{E}_\mu[\cdot] := \int_{\mathcal{S}} \cdot \, d\mu$ expectation over $\mathcal{S}$ with respect to a probability measure μ over $\mathcal{S}$.

5. Technically, ϕ need only be differentiable on the relative interior $\mathrm{ri}(\mathcal{S})$ of $\mathcal{S}$. We omit this requirement for simplicity and because it is not relevant to this discussion.
6. Named in reference to Bregman (1967) although he was not the first to consider such an equation, at least in the one dimensional case; confer Brunk et al. (1957, p.838).

Theorem 5 (Jensen's Inequality) *Let $\phi : \mathcal{S} \to \mathbb{R}$ be a convex function, μ be a distribution and S be an $\mathcal{S}$-valued random variable (measurable w.r.t. μ) such that $\mathbb{E}_\mu[|\mathsf{S}|] < \infty$. Then*

$$\mathbb{J}_\mu[\phi] := \mathbb{E}_\mu[\phi(\mathsf{S})] - \phi(\mathbb{E}_\mu[\mathsf{S}]) \geq 0. \tag{9}$$

The proof is straight-forward and can be found in (Dudley, 2003, §10.2). Jensen's inequality can also be used to characterise the class of convex functions. If ϕ is a function such that (9) holds for all random variables and distributions then ϕ must be convex.[7] Intuitively, this connection between expectation and convexity is natural since expectation can be seen as an operator that takes convex combinations of random variables.

We will call the difference $\mathbb{J}_\mu[\phi]$ the *Jensen gap for ϕ when $\mathsf{S} \sim \mu$*. Many measures of divergence and information studied in the subsequent sections can be expressed as the Jensen gap of some convex function. Due to the linearity of expectation, the Jensen gap is insensitive to the addition of affine terms to the convex function that defines it:

Theorem 6 *Let $\phi : \mathcal{S} \to \mathbb{R}$ be convex function and S and μ be as in Theorem 5. Then for each $a, b \in \mathbb{R}$ the convex function $\psi(s) := \phi(s) + as + b$ satisfies $\mathbb{J}_\mu[\phi(\mathsf{S})] = \mathbb{J}_\mu[\psi(\mathsf{S})]$.*

The proof is a consequence of the definition of the Jensen gap and the linearity of expectations and can be found in Appendix A.2. An implication of this theorem is that when considering sets of convex functions as parameters to the Jensen gap operator they only need be identified by their non-linear part. Thus, the Jensen gap operator can be seen to impose an equivalence relation over convex functions where two convex functions are equivalent if they have the same Jensen gap, that is, if their difference is affine.

In light of the two integral representations in Section 2.3, this means the Jensen gap only depends on the integral terms in (5) and (6) and so is completely characterised by the weights provided by ϕ''. Specifically, for suitably differentiable $\phi : [a, b] \to \mathbb{R}$ we have

$$\mathbb{J}_\mu[\phi(\mathsf{S})] = \int_a^b \mathbb{J}_\mu[\phi_{s_0}(\mathsf{S}, t)]\, \phi''(t)\, dt.$$

Since several of the measures of divergence, information and risk we analyse can be expressed as a Jensen gap, this observation implies that these quantities can be identified with the weights provided by ϕ'' as it is these that completely determine the measure's behaviour.

3. Binary Experiments and Measures of Divergence

The various properties of convex functions developed in the previous section have many implications for the study of statistical inference. We begin by considering *binary experiments* (P, Q) where P and Q are probability measures[8] over a common space $\mathcal{X}$. We will consider P the distribution over *positive* instances and Q the distribution over *negative* instances. The densities of P and Q with respect to some third reference distribution M over $\mathcal{X}$ will be defined by $dP = p\,dM$ and $dQ = q\,dM$ respectively. Unless stated otherwise we will assume that P and Q are both absolutely continuous

7. This can be seen by considering a distribution with a finite, discrete set of points as its support and applying Theorem 4.3 of Rockafellar (1970).
8. We intentionally avoid too many measure theoretic details for the sake of clarity. Appropriate σ-algebras and continuity can be assumed where necessary.

with respect to M. (One can always choose M to ensure this by setting $M := (P+Q)/2$; but see the next section.)

There are several ways in which the "separation" of P and Q in a binary experiment can be quantified. Intuitively, these all measure the difficulty of distinguishing between the two distributions using instances drawn from their mixture. The further apart the distributions are the easier discrimination becomes. This intuition is made precise through the connections with risk and MMD later in Appendix H.

A central statistic in the study of binary experiments and statistical hypothesis testing is the likelihood ratio dP/dQ. As the following section outlines, the likelihood ratio is, in the sense of preserving the distinction between P and Q, the "best" mapping from an arbitrary space $\mathcal{X}$ to the real line.

3.1 Statistical Tests and the Neyman-Pearson Lemma

In the context of a binary experiment (P,Q), a *statistical test* is any function that assigns each instance $x \in \mathcal{X}$ to either P or Q. We will use the labels 1 and 0 for P and Q respectively and so a statistical test is any function $r : \mathcal{X} \to \{0,1\}$. In machine learning, a function of this type is usually referred to as a *classifier*. The link between tests and classifiers is explored further in Section 4.

Each test r partitions the instance space $\mathcal{X}$ into positive and negative *prediction sets*:

$$\begin{aligned} \mathcal{X}_r^+ &:= \{x \in \mathcal{X} \,:\, r(x) = 1\}, \\ \mathcal{X}_r^- &:= \{x \in \mathcal{X} \,:\, r(x) = 0\}. \end{aligned}$$

There are four *classification rates* associated with these predictions sets: the true positive rate (TP), true negative rate (TN), false positive rate (FP) and the false negative rate (FN). For a given test r they are defined as follows:

$$\begin{aligned} \mathrm{TP}_r &:= P(\mathcal{X}_r^+), \quad & \mathrm{FP}_r &:= Q(\mathcal{X}_r^+), \\ \mathrm{FN}_r &:= P(\mathcal{X}_r^-), \quad & \mathrm{TN}_r &:= Q(\mathcal{X}_r^-). \end{aligned} \tag{10}$$

The subscript r will be dropped when the test is clear by the context. Since P and Q are distributions over $\mathcal{X} = \mathcal{X}_r^+ \cup \mathcal{X}_r^-$ and the positive and negative sets are disjoint we have that $\mathrm{TP} + \mathrm{FN} = 1$ and $\mathrm{FP} + \mathrm{TN} = 1$. As a consequence, the four values in (10) can be summarised by choosing one from each column.

Often, statistical tests are obtained by applying a threshold τ_0 to a real-valued *test statistic* $\tau : \mathcal{X} \to \mathbb{R}$. In this case, the statistical test is $r(x) = [\![\tau(x) \geq \tau_0]\!]$. This leads to parameterised forms of prediction sets $\mathcal{X}_\tau^y(\tau_0) := \mathcal{X}^y_{[\![\tau \geq \tau_0]\!]}$ for $y \in \{+,-\}$, and the classification rates $\mathrm{TP}_\tau(\tau_0)$, $\mathrm{FP}_\tau(\tau_0)$, $\mathrm{TN}_\tau(\tau_0)$, and $\mathrm{TP}_\tau(\tau_0)$ which are defined analogously. By varying the threshold parameter a range of classification rates can be achieved. This observation leads to a well known graphical representation of test statistics known as the ROC curve, which is discussed further in Section 6.1.

A natural question is whether there is a "best" statistical test or test statistic to use for binary experiments. This is usually formulated in terms of a test's power and size. The *power* β_r of the test r for a particular binary experiment (P,Q) is a synonym for its true positive rate (that is, $\beta_r := \mathrm{TP}_r$ and so $1 - \beta_r := \mathrm{FN}_r$[9]) and the *size* α_r of same test is just its false positive rate $\alpha_r := \mathrm{FP}_r$. Here,

9. This is opposite to the usual definition of β_r in the statistical literature. Usually, $1 - \beta_r$ is used to denote the power of a test. We have chosen to use β_r for the power (true positive rate) as this makes it easier to compare with ROC curves and it is consistent with the usage of Torgersen (1991).

"best" is considered to be the *most powerful* (MP) test of a given size (Bickel and Doksum, 2001, §4.2). That is, a test r is considered MP of size $\alpha \in [0,1]$ if, $\alpha_r = \alpha$ and for all other tests r' such that $\alpha_{r'} \leq \alpha$ we have $1-\beta_r \leq 1-\beta_{r'}$. We will denote by $\beta(\alpha) := \beta(\alpha, P, Q)$ the true positive rate of an MP test between P (the alternative hypothesis) and Q (the null hypothesis) at Q with significance α. Torgersen (1991) calls $\beta(\cdot, P, Q)$ the *Neyman-Pearson function for the dichotomy* (P, Q). Formally, for each $\alpha \in [0,1]$, the Neyman-Pearson function β measures the largest true positive rate TP_r of any measurable classifier $r\colon \mathcal{X} \to \{-1,1\}$ that has false positive rate FP_r at most α. That is,

$$\beta(\alpha) = \beta(\alpha, P, Q) := \sup_{r \in \{-1,1\}^{\mathcal{X}}} \{\mathrm{TP}_r : \ \mathrm{FP}_r \leq \alpha\}. \tag{11}$$

The Neyman-Pearson lemma (Neyman and Pearson, 1933) shows that the likelihood ratio $\tau^*(x) = dP/dQ(x)$ is the most powerful test for each choice of threshold τ_0. Since each choice of $\tau_0 \in \mathbb{R}$ results in a test $[\![dP/dQ \geq \tau_0]\!]$ of some size $\alpha \in [0,1]$ we have that[10]

$$\beta(\mathrm{FP}_{\tau^*}(\tau_0)) = \mathrm{TP}_{\tau^*}(\tau_0) \tag{12}$$

and so varying τ_0 over $\mathbb{R}$ results in a maximal ROC curve. This too is discussed further in Section 6.1.

The Neyman-Pearson lemma thus identifies the likelihood ratio dP/dQ as a particularly useful statistic. Given an experiment (P, Q) it is, in some sense, the best mapping from the space $\mathcal{X}$ to the reals. The next section shows how this statistic can be used as the basis for a variety of divergence measures between P and Q.

3.2 Csiszár f-divergences

The class of *f-divergences* (Ali and Silvey, 1966; Csiszár, 1967) provide a rich set of relations that can be used to measure the separation of the distributions in a binary experiment. An f-divergence is a function that measures the "distance" between a pair of distributions P and Q defined over a space $\mathcal{X}$ of observations. Traditionally, the f-divergence of P from Q is defined for any convex $f : (0,\infty) \to \mathbb{R}$ such that $f(1) = 0$. In this case, the f-divergence is

$$\mathbb{I}_f(P, Q) = \mathbb{E}_Q\left[f\left(\frac{dP}{dQ}\right)\right] = \int_{\mathcal{X}} f\left(\frac{dP}{dQ}\right) dQ \tag{13}$$

when P is absolutely continuous with respect to Q and equals ∞ otherwise.[11]

The above definition is not completely well-defined as the behaviour of f is not specified at the endpoints of $(0,\infty)$. This is remedied via the perspective transform of f, introduced in Section 2.1 above which defines the limiting behaviour of f. Given convex $f : (0,\infty) \to \mathbb{R}$ such that $f(1) = 0$ the *f-divergence of P from Q* is

$$\mathbb{I}_f(P, Q) := \mathbb{E}_M[I_f(p, q)] = \mathbb{E}_{\mathsf{X} \sim M}[I_f(p(\mathsf{X}), q(\mathsf{X}))], \tag{14}$$

where I_f is the perspective transform of f (see (1)).

10. Equation (43) in Section 6.3 below, shows that $\beta(\alpha)$ is the lower envelope of a family of linear functions of α and is thus concave and continuous. Hence, the equality in (12) holds.

11. Liese and Miescke (2008, pg. 34) give a definition that does not require absolute continuity.

The restriction that $f(1) = 0$ in the above definition is only present to normalise $\mathbb{I}_f$ so that $\mathbb{I}_f(Q,Q) = 0$ for all distributions Q. We can extend the definition of f-divergences to all convex f by performing the normalisation explicitly. Since $f(\mathbb{E}_Q[dP/dQ]) = f(1)$ this is done most conveniently through the definition of the Jensen gap for the function f applied to the random variable dP/dQ with distribution Q. That is, for all convex $f : (0,\infty) \to \mathbb{R}$ and for all distributions P and Q

$$\mathbb{J}_Q\left[f\left(\frac{dP}{dQ}\right)\right] = \mathbb{I}_f(P,Q) - f(1). \tag{15}$$

Due to the issues surrounding the behaviour of f at 0 and ∞ the definitions in (13), (14) and (15) are not entirely equivalent. When it is necessary to deal with the limiting behaviour, the definition in (14) will be used. However, the version in (15) will be most useful when drawing connections between f-divergences and various definitions of information in Section 4 below.

Several properties of f-divergence can be immediately obtained from the above definitions. The symmetry of the perspective I_f in (2) means that

$$\mathbb{I}_f(P,Q) = \mathbb{I}_{f^\diamondsuit}(Q,P) \tag{16}$$

for all distributions P and Q, where $f^\diamondsuit$ is the Csiszár dual of f. The non-negativity of the Jensen gap ensures that $\mathbb{I}_f(P,Q) \geq 0$ for all P and Q. Furthermore, the affine invariance of the Jensen gap (Theorem 6) implies the same affine invariance for f-divergences.

Several well-known divergences correspond to specific choices of the function f (Ali and Silvey, 1966, §5). One divergence central to this paper is the *variational divergence* $V(P,Q)$ which is obtained by setting $f(t) = |t-1|$ in Equation 14. It is the only f-divergence that is a true metric on the space of distributions over $\mathcal{X}$ (Khosravifard et al., 2007) and gets its name from its equivalent definition in the variational form

$$V(P,Q) = 2\|P-Q\|_\infty := 2\sup_{A\subseteq\mathcal{X}} |P(A) - Q(A)|.$$

(Some authors define V without the 2 above.) This form of the variational divergence is discussed further in Section 8. Furthermore, the variational divergence is one of a family of "primitive" f-divergences discussed in Section 5. These are primitive in the sense that all other f-divergences can be expressed as a weighted sum of members from this family.

Another well known f-divergence is the Kullback-Leibler (KL) divergence $\mathrm{KL}(P,Q)$, obtained by setting $f(t) = t\ln(t)$ in Equation 14. Others are given in Table 2 in Section 5.4.

3.3 Generative Bregman Divergences

Another measure of the separation of distributions can be defined as the expected Bregman divergence between the densities p and q with respect to the reference measure M. Given a convex function $\phi : \mathbb{R}^+ \to \mathbb{R}$ the *generative Bregman divergence* between the distributions P and Q is (confer (14))

$$\mathbb{B}_\phi(P,Q) := \mathbb{E}_M\left[B_\phi(p,q)\right] = \mathbb{E}_{\mathsf{X}\sim M}\left[B_\phi(p(\mathsf{X}),q(\mathsf{X}))\right].$$

We call this Bregman divergence "generative" to distinguish it from the "discriminative" Bregman divergence introduced in Section 4 below, where the adjectives "generative" and "discriminative" are explained further.

Csiszár (1995) notes that there is only one divergence common to the class of f-divergences and the generative Bregman divergences. In this sense, these two classes of divergences are "orthogonal" to each other. Their only common point is when the respective convex functions satisfy $f(t) = \phi(t) = t\ln t - at + b$ (for $a, b \in \mathbb{R}$) in which case both $\mathbb{I}_f$ and $\mathbb{B}_\phi$ are the KL divergence.

4. Risk and Statistical Information

The above discussion of f-divergences assumes an arbitrary reference measure M over the space $\mathcal{X}$ to define the densities p and q. In the previous section, the choice of reference measure was irrelevant since f-divergences are invariant to this choice.

In this section an assumption is made that adds additional structure to the relationship between P and Q. Specifically, we assume that the reference measure M is a mixture of these two distributions. That is, $M = \pi P + (1-\pi)Q$ for some $\pi \in (0,1)$. In this case, by construction, P and Q are absolutely continuous with respect to M. Intuitively, this can be seen as defining a distribution over the observation space $\mathcal{X}$ by first tossing a coin with a bias π for heads and drawing observations from P on heads or Q on tails.

This extra assumption allows us to interpret a binary experiment (P,Q) as a generalised *supervised binary task* (π, P, Q) where the positive ($y = 1$) and negative ($y = -1$) *labels* $y \in \mathcal{Y} := \{-1, 1\}$ are paired with *observations* $x \in \mathcal{X}$ through a joint distribution $\mathbb{P}$ over $\mathcal{X} \times \mathcal{Y}$. (We formally define a task later in terms of an experiment plus loss function.) Given an observation drawn from $\mathcal{X}$ according to M, it is natural to try to predict its corresponding label or estimate the probability it was drawn from P.

Below we will introduce risk, regret, and proper losses and show how these relate to discriminative Bregman divergence. We then show the connection between the generative view (f-divergence between the class conditional distributions) and Bregman divergence.

4.1 Generative and Discriminative Views

Traditionally, the joint distribution $\mathbb{P}$ of inputs $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$ is used as the starting point for analysing risk in statistical learning theory. In order to better link risks to divergences, in our analysis we will consider two related representations of $\mathbb{P}$.

The *generative* view decomposes the joint distribution $\mathbb{P}$ into two *class-conditional distributions* defined as $P(X) := \mathbb{P}(X|y=1)$, $Q(X) := \mathbb{P}(X|y=-1)$ for all $X \subseteq \mathcal{X}$ and a mixing probability or *prior* $\pi := \mathbb{P}(\mathcal{X}, y=1)$. The *discriminative* representation decomposes the joint distribution into an *observation distribution* $M(X) := \mathbb{P}(X, \mathcal{Y})$ for all $X \subseteq \mathcal{X}$ and an *observation-conditional density* or *posterior* $\eta(x) = \frac{dH}{dM}(x)$ where $H(X) := \mathbb{P}(X, y=1)$. The terms "generative" and "discriminative" are used here to suggest a distinction made by Ng and Jordan (2002): in the generative case, the aim is to model the class-conditional distributions P and Q and then use Bayes rule to compute the most likely class; in the discriminative case the focus is on estimating $\eta(x)$ directly. Although we are not directly interested in this paper in the problems of modelling or estimating we find the distinction a useful one.[12]

12. The generative-discriminative distinction usually refers to whether one is modelling the process that generates each class-conditional distribution, or instead wishes solely to perform well on a discrimination task (Drummond, 2006; Lasserre et al., 2006; Minka, 2005; Rubinstein and Hastie, 1997). There has been some recent work relating the two in the sense that if the class conditional distributions are well estimated then will one perform well in discrimination (Long and Servedio, 2006; Long et al., 2006; Goldberg, 2001; Palmer and Goldberg, 2006).

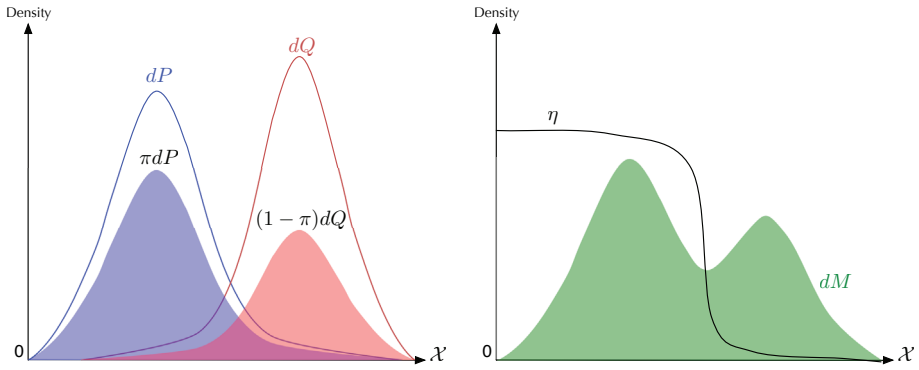


Figure 2: The generative and discriminative view of binary experiments.

Both these decompositions are exact since $\mathbb{P}$ can be reconstructed from either. Also, translating between them is straight-forward, since

$$M = \pi P + (1-\pi)Q \quad \text{and} \quad \eta = \pi \frac{dP}{dM},$$

so we will often swap between (η, M) and (π, P, Q) as arguments to functions for risk, divergence and information. A graphical representation of the generative and discriminative views of a binary experiment is shown in Figure 2.

The posterior η is closely related to the likelihood ratio dP/dQ in the supervised binary task setting. For each choice of $\pi \in (0,1)$ this relationship can be expressed by a mapping $\lambda_\pi : [0,1] \to [0,\infty]$ and its inverse λ_π^{-1} defined by

$$\begin{aligned} \lambda_\pi(c) &:= \frac{1-\pi}{\pi}\frac{c}{1-c}, \\ \lambda_\pi^{-1}(t) &= \frac{\pi t}{\pi t + 1 - \pi} \end{aligned} \tag{17}$$

for all $c \in [0,1)$ and $t \in [0,\infty)$, and $\lambda_\pi(1) := \infty$. Thus

$$\eta = \lambda_\pi^{-1}\left(\frac{dP}{dQ}\right) \quad \text{and, conversely,} \quad \frac{dP}{dQ} = \lambda_\pi(\eta).$$

These will be used later when relating f-divergences and risk.

4.2 Estimators and Risk

We will call a (M-measurable) function $\hat{\eta} : \mathcal{X} \to [0,1]$ a class probability *estimator*. Overloading the notation slightly, we will also use $\hat{\eta} = \hat{\eta}(x) \in [0,1]$ to denote an *estimate* for a specific observation $x \in \mathcal{X}$. Many of the subsequent arguments rely on this conditional perspective.

Estimate quality is assessed using a *loss function* $\ell : \mathcal{Y} \times [0,1] \to \mathbb{R}$ and the loss of the estimate $\hat{\eta}$ with respect to the label $y \in \mathcal{Y}$ is denoted $\ell(y, \hat{\eta})$. If $\eta \in [0,1]$ is the probability of observing the label $y = 1$ then the *point-wise risk* of the estimate $\hat{\eta} \in [0,1]$ is defined to be the η-average of the point-wise loss for $\hat{\eta}$:

$$L(\eta, \hat{\eta}) := \mathbb{E}_{Y \sim \eta}[\ell(Y, \hat{\eta})] = \ell(0, \hat{\eta})(1-\eta) + \ell(1, \hat{\eta})\eta. \tag{18}$$

(This is what Steinwart 2006 calls the *inner risk*.) When $\eta : \mathcal{X} \to [0,1]$ is an observation-conditional density, taking the M-average of the point-wise risk gives the *(full) risk* of the estimator $\hat{\eta}$:

$$\begin{aligned}\mathbb{L}(\eta,\hat{\eta},M) &:= \mathbb{E}_M[L(\eta,\hat{\eta})] = \mathbb{E}_{\mathsf{X}\sim M}[L(\eta(\mathsf{X}),\hat{\eta}(\mathsf{X}))] \\ &= \int_{\mathcal{X}} L(\eta(x),\hat{\eta}(x))\,dM(x) =: \mathbb{L}(\pi,\hat{\eta},P,Q).\end{aligned}$$

The convention of using ℓ, L and $\mathbb{L}$ for the loss, point-wise and full risk is used throughout this paper. Any names or parameters associated to ℓ will be propagated to L and $\mathbb{L}$.

We call the combination of a loss ℓ and the distribution $\mathbb{P}$ a *task* and denote it discriminatively as $T = (\eta, M; \ell)$ or generatively as $T = (\pi, P, Q; \ell)$. A natural measure of the difficulty of a task is its minimal achievable risk, or *Bayes risk*:

$$\underline{\mathbb{L}}(\eta,M) = \underline{\mathbb{L}}(\pi,P,Q) := \inf_{\hat{\eta}\in[0,1]^{\mathcal{X}}} \mathbb{L}(\eta,\hat{\eta},M) = \mathbb{E}_{\mathsf{X}\sim M}\left[\underline{L}(\eta(\mathsf{X}))\right],$$

where

$$[0,1] \ni \eta \mapsto \underline{L}(\eta) := \inf_{\hat{\eta}\in[0,1]} L(\eta,\hat{\eta})$$

is the *point-wise Bayes risk*. Note the use of the underline on $\underline{\mathbb{L}}$ and $\underline{L}$ to indicate that the corresponding functions $\mathbb{L}$ and L are minimised.

4.3 Proper Losses

If $\hat{\eta}$ is to be interpreted as an estimate of the true positive class probability η then it is desirable to require that $L(\eta,\hat{\eta})$ be minimised when $\hat{\eta} = \eta$ for all $\eta \in [0,1]$. Losses that satisfy this constraint are said to be *Fisher consistent* and are known as *proper scoring rules* (Buja et al., 2005; Gneiting and Raftery, 2007). To use common machine learning terminology we will refer to Fisher consistent losses as *proper losses*. This implies that a proper loss ℓ satisfies $\underline{L}(\eta) = L(\eta,\eta)$ for all $\eta \in [0,1]$.

There are a few properties of losses that we will require to establish certain key theorems below. The first of these is that we will say a loss is *fair* whenever $\eta \mapsto \ell(0,\eta)$ and $\eta \mapsto \ell(1,\eta)$ are, respectively, right continuous at 0 and left continuous at 1, and

$$\ell(0,0) = \ell(1,1) = 0.$$

That is, no loss incurred for perfect prediction and there are no sudden "jumps" in penalty for near-perfect prediction. The main place fairness is relied upon is in the integral representation of Theorem 16 where it is used to get rid of some constants of integration. In order to explicitly construct a proper loss from its associated "weight function" as shown in Theorem 17 we will require that the loss be *definite*, that is, its point-wise Bayes risk at 0 and 1 must be bounded from below:

$$\underline{L}(0) > -\infty\ ,\ \underline{L}(1) > -\infty.$$

Since properness of a loss ensures $\underline{L}(\eta) = L(\eta,\eta)$ we see that a fair proper loss is necessarily definite since $L(0,0) = \ell(0,0) = 0 > -\infty$, and similarly for $L(1,1)$. Conversely, if a proper loss is definite then the finite values $\ell(0,0)$ and $\ell(1,1)$ can be subtracted from $\ell(0,\cdot)$ and $\ell(1,\cdot)$ to make it fair.

Finally, for Theorem 7 below to hold at the endpoints of the unit interval we require a loss to be *regular*, that is,

$$\lim_{\eta\searrow 0} \eta\ell(1,\eta) = \lim_{\eta\nearrow 1} (1-\eta)\ell(0,\eta) = 0. \tag{19}$$

Intuitively, this condition ensures that making mistakes on events that never happen should not incur a penalty. It is not difficult to show that any fair, definite loss is also regular (thus, a proper and fair loss is also regular) but the converse does not hold. Since properness and fairness imply definiteness and regularity, most of the situations we consider in the remainder of this paper will involve losses which are both proper and fair.

Proper losses for probability estimation and surrogate margin losses (confer Bartlett et al. 2006) for classification are closely related. (Surrogate margin losses are considered in more detail in Appendix D.) Buja et al. (2005) note that "the surrogate criteria of classification are exactly the primary criteria of class probability estimation" and that most commonly used surrogate margin losses are just proper losses mapped from $[0,1]$ to $\mathbb{R}$ via a link function. The main exceptions are hinge losses;[13] Buja et al. (2005, pg. 4) state that SVMs are "the only case that truly bypasses estimation of class probabilities and directly aims at classification." However, commonly used margin losses of the form $\phi(yF(x))$ are a more restrictive class than proper losses since, as Buja et al. (2005, §23) note, "[t]his dependence on the margin limits all theory and practice to a symmetric treatment of class 0 and class 1". The relation between link functions, proper losses and margin losses is considered in more detail by Reid and Williamson (2010).

The following important property of proper losses seems to be originally due to Savage (1971). It shows that a proper loss is completely characterised by a concave function defining its point-wise Bayes risk along with a simple structural relationship between its point-wise risk and Bayes risk.

Theorem 7 *A loss function ℓ is proper if and only if its point-wise Bayes risk $\underline{L}(\eta)$ is concave and for each $\eta, \hat{\eta} \in (0,1)$*

$$L(\eta, \hat{\eta}) = \underline{L}(\hat{\eta}) + (\eta - \hat{\eta})\underline{L}'(\hat{\eta}).$$

Furthermore if ℓ is regular this characterisation also holds at the endpoints $\eta, \hat{\eta} \in \{0,1\}$.

For general concave functions $\underline{L}$ which may not be differentiable, $(-\underline{L})'$ is to be taken to be a right derivative as discussed in Section 2.4. The following proof uses an argument in Buja et al. (2005, §17) for the forward direction and the generalised Taylor's theorem due to Liese and Vajda (2006) for the converse.

Proof By definition, the point-wise Bayes risk $\underline{L}(\eta) = \inf_{\hat{\eta}} L(\eta, \hat{\eta})$ which, for each $\eta \in [0,1]$ is just the lower envelope of the lines $L(\eta, \hat{\eta}) = (1-\eta)\ell(0, \hat{\eta}) + \eta\ell(1, \hat{\eta})$ and thus $\underline{L}$ is concave.[14] The properness of ℓ means $\underline{L}(\eta) = L(\eta, \eta)$ and the $\hat{\eta}$-derivative of L is 0 when $\hat{\eta} = \eta$. Hence

$$\frac{\partial}{\partial \hat{\eta}} L(\eta, \hat{\eta}) \bigg|_{\hat{\eta}=\eta} = (1-\eta)\ell'(0, \eta) + \eta\ell'(1, \eta) = 0$$

for all $\eta \in [0,1]$. Using this and expanding $\underline{L}'(\eta)$ via the product rule, a little algebra shows $\underline{L}'(\eta) = \ell(1,\eta) - \ell(0,\eta)$. Thus

$$\begin{aligned} \underline{L}(\hat{\eta}) + (\eta - \hat{\eta})\underline{L}'(\hat{\eta}) &= (1-\hat{\eta})\ell(0,\hat{\eta}) + \hat{\eta}\ell(1,\hat{\eta}) + (\eta - \hat{\eta})[\ell(1,\hat{\eta}) - \ell(0,\hat{\eta})] \\ &= (1-\eta)\ell(0,\hat{\eta}) + \eta\ell(1,\hat{\eta}), \end{aligned}$$

which is the definition of $L(\eta, \hat{\eta})$. The result holds at the endpoints if the loss is regular by applying the assumptions in (19).

13. And powers of absolute divergence $|y - r|^\alpha$ for $\alpha \neq 2$.

14. Since this argument made no use of the properness of ℓ we see the concavity of the Bayes risk holds for any loss.

Conversely, now suppose Λ is a concave function and let $\ell(y,\hat{\eta}) = \Lambda(\hat{\eta}) + (y-\hat{\eta})\Lambda'(\hat{\eta})$. The Taylor expansion of Λ is

$$\Lambda(\eta) \quad = \quad \Lambda(\hat{\eta}) + (\eta-\hat{\eta})\Lambda'(\hat{\eta}) + \int_{\hat{\eta}}^{\eta} (\eta - c)\,\Lambda''(c)\,dc$$

and so

$$L(\eta,\hat{\eta}) = \Lambda(\hat{\eta}) - \int_{\hat{\eta}}^{\eta} (\eta - c)\,\Lambda''(c)\,dc \geq \Lambda(\eta) = \underline{L}(\eta)$$

because the concavity of Λ means $\Lambda'' \leq 0$ and so the integral term is positive and is minimised to 0 when $\hat{\eta} = \eta$. This shows ℓ is proper, completing the proof. ■

This characterisation of the concavity of $\underline{L}$ means proper losses have a natural connection to Bregman divergences.

4.4 Discriminative Bregman Divergence

Recall from Section 2.5 that if $\mathcal{S} \subseteq \mathbb{R}^d$ is a convex set, then a convex function $\phi : \mathcal{S} \to \mathbb{R}$ defines a *Bregman divergence*

$$B_\phi(s, s_0) := \phi(s) - \phi(s_0) - \langle s - s_0, \nabla\phi(s_0)\rangle\,.$$

When $\mathcal{S} = [0,1]$, the concavity of $\underline{L}$ means $\phi(s) = -\underline{L}(s)$ is convex and so induces the Bregman divergence[15]

$$B_\phi(s, s_0) = -\underline{L}(s) + \underline{L}(s_0) - (s_0 - s)\underline{L}'(s_0) = L(s, s_0) - \underline{L}(s)$$

by Theorem 7. The converse also holds. Given a Bregman divergence B_ϕ over $\mathcal{S} = [0,1]$ the convexity of ϕ guarantees that $\underline{L} = -\phi$ is concave. Thus, we know that there is a proper loss ℓ with Bayes risk equal to $-\phi$. As noted by Buja et al. (2005, §19), the difference

$$B_\phi(\eta, \hat{\eta}) = L(\eta, \hat{\eta}) - \underline{L}(\eta)$$

is also known as the *point-wise regret* of the estimate $\hat{\eta}$ w.r.t. η. The corresponding *(full) regret* is the M-average point-wise regret

$$\mathbb{E}_{\mathsf{X}\sim M}[B_\phi(\eta(\mathsf{X}), \hat{\eta}(\mathsf{X}))] = \mathbb{L}(\eta, \hat{\eta}, M) - \underline{\mathbb{L}}(\eta, M).$$

4.5 Bregman Information

Banerjee et al. (2005a) recently introduced the notion of the *Bregman information* $\mathbb{B}_\phi(\mathsf{S})$ of a random variable S drawn according to some distribution σ over $\mathcal{S}$. It is the minimal σ-average Bregman divergence that can be achieved by an element $s^* \in \mathcal{S}$ (the *Bregman representative*). In symbols,

$$\mathbb{B}_\phi(\mathsf{S}) := \inf_{s\in\mathcal{S}} \mathbb{E}_{\mathsf{S}\sim\sigma}\left[B_\phi(\mathsf{S}, s)\right] = \mathbb{E}_{\mathsf{S}\sim\sigma}\left[B_\phi(\mathsf{S}, s^*)\right].$$

The authors show that the mean $s := \mathbb{E}_{\mathsf{S}\sim\sigma}[\mathsf{S}]$, is the unique Bregman representative. That is, $\mathbb{B}_\phi(\mathsf{S}) = \mathbb{E}_\sigma[B_\phi(\mathsf{S}, s)]$. Surprisingly, this minimiser *only* depends on $\mathcal{S}$ and σ, not the choice of ϕ

15. Technically, $\mathcal{S}$ is the 2-simplex $\{(s_1, s_2) \in [0,1]^2 : s_1 + s_2 = 1\}$ but we identify $s \in [0,1]$ with $(s, 1-s)$. Also, we once again interpret $(-\underline{L})'$ as a right derivative for general concave $\underline{L}$ as discussed in Section 2.4.

defining the divergence and is a consequence of Jensen's inequality and the form of the Bregman divergence.

Since regret is a Bregman divergence, it is natural to ask what is the corresponding Bregman information. In this case, $\phi = -\underline{L}$ and the random variable $\mathsf{S} = \eta(\mathsf{X}) \in [0,1]$, where $\mathsf{X} \in \mathcal{X}$ is distributed according to the observation distribution M. Noting that $\mathbb{E}_{\mathsf{X}\sim M}[\eta(\mathsf{X})] = \pi$, the proof of the following theorem stems from the definition of Bregman information and some simple algebra showing that $\inf_\eta \mathbb{L}(\eta,\pi,M) = \underline{\mathbb{L}}(\pi,M)$, since by assumption ℓ is a proper loss.

Theorem 8 *Suppose ℓ is a proper loss. Given a discriminative task (η, M) and letting $\phi = -\underline{L}$, the corresponding Bregman information of $\eta(\mathsf{X})$ satisfies*

$$\mathbb{B}_\phi(\eta(\mathsf{X})) = \mathbb{B}_\phi(\eta, M) := \underline{\mathbb{L}}(\pi, M) - \underline{\mathbb{L}}(\eta, M).$$

4.6 Statistical Information

The reduction in risk (from prior $\pi \in [0,1]$ to posterior $\eta \in [0,1]^{\mathcal{X}}$)

$$\Delta\underline{\mathbb{L}}(\eta, M) = \Delta\underline{\mathbb{L}}(\pi, P, Q) := \underline{\mathbb{L}}(\pi, M) - \underline{\mathbb{L}}(\eta, M) \tag{20}$$

is known as *statistical information* and was introduced by DeGroot (1962) motivated by Lindley (1956). This reduction can be interpreted as how much risk is removed by knowing observation-specific class probabilities η rather than just the prior π.

DeGroot originally introduced statistical information in terms of what he called an *uncertainty function* which, in the case of binary experiments, is any function $U : [0,1] \to [0,\infty)$. The statistical information is then the average reduction in uncertainty which can be expressed as a concave Jensen gap

$$-\mathbb{J}_M[U(\eta)] = \mathbb{J}_M[-U(\eta)] = U(\mathbb{E}_{\mathsf{X}\sim M}[\eta(\mathsf{X})]) - \mathbb{E}_{\mathsf{X}\sim M}[U(\eta(\mathsf{X}))].$$

DeGroot noted that Jensen's inequality implies that for this quantity to be non-negative the uncertainty function must be concave, that is, $-U$ must be convex.

Theorem 8 shows that statistical information is a Bregman information and corresponds to the Bregman divergence obtained by setting $\phi = -\underline{L}$. This connection readily shows that $\Delta\underline{\mathbb{L}}(\eta, M) \geq 0$ (DeGroot, 1962, Thm 2.1) since the minimiser of the Bregman information is $\pi = \mathbb{E}_{\mathsf{X}\sim M}[\eta(\mathsf{X})]$ regardless of loss and $B_\phi(\eta,\pi) \geq 0$ since it is a regret.

4.7 Unifying Information and Divergence

From a generative perspective, f-divergences can be used to assess the difficulty of a learning task by measuring the divergence between the class-conditional distributions P and Q. The more divergent the distributions for the two classes, the easier the classification task. Österreicher and Vajda (1993, Thm 2) made this relationship precise by showing that f-divergence and statistical information have a one-to-one correspondence:

Theorem 9 *If $(\pi, P, Q; \ell)$ is an arbitrary task and $\underline{L}$ is the associated conditional Bayes risk then defining*

$$f^\pi(t) := \underline{L}(\pi) - (\pi t + 1 - \pi)\underline{L}\left(\frac{\pi t}{\pi t + 1 - \pi}\right) \tag{21}$$

for $\pi \in [0,1]$ *implies* f^π *is convex,* $f^\pi(1) = 0$ *and*

$$I_{f^\pi}(P,Q) = \Delta\underline{\mathbb{L}}(\pi,P,Q)$$

for all distributions P and Q. Conversely, if f *is convex and* $f(1) = 0$ *then defining*

$$\underline{L}^\pi(\eta) := -\frac{1-\eta}{1-\pi} f\left(\frac{1-\pi}{\pi}\frac{\eta}{1-\eta}\right), \quad \pi \in [0,1]$$

implies

$$I_f(P,Q) = \Delta\underline{\mathbb{L}}^\pi(\pi,P,Q)$$

for all distributions P and Q, where $\Delta\underline{\mathbb{L}}^\pi$ *is the statistical information associated with* $\underline{L}^\pi$.

The proof, given in Appendix A.3, is a straight-forward calculation that exploits the relationships between the generative and discriminative views presented earlier. Combined with the link between Bregman and statistical information, this result means that they and f-divergences are *interchangeable* as measures of task difficulty. The theorem leads to some correspondences between well known losses and divergence: log-loss with $\mathrm{KL}(P,Q)$; square loss with triangular discrimination; and 0-1 loss with $V(P,Q)$. (See Section 5.5 for an explicitly worked out example.)

This connection generalises the link between f-divergences and F-errors (expectations of concave functions of η) in Devroye et al. (1996) and can be compared to the more recent work of Nguyen et al. (2005) who show that each f-divergence corresponds to the negative Bayes risk for a *family* of surrogate margin losses. The one-to-many nature of their result may seem at odds with the one-to-one relationship here. However, the family of margin losses given in their work can be recovered by combining the proper losses with link functions. Working with proper losses also addresses a limitation pointed out by Nguyen et al. (2005, pg. 14), namely that "asymmetric f-divergences cannot be generated by *any* (margin-based) surrogate loss function" and extends their analysis "to show that asymmetric f-divergences can be realized by general (asymmetric) loss functions".

4.8 Summary

The main results of this section can be summarised as follows.

Theorem 10 *Let* $f : \mathbb{R}^+ \to \mathbb{R}$ *be a convex function and for each* $\pi \in [0,1]$ *define for* $c \in [0,1)$*:*

$$\begin{aligned} \phi(c) &:= \frac{1-c}{1-\pi} f\left(\lambda_\pi(c)\right), \\ \underline{L}(c) &:= -\phi(c), \end{aligned}$$

where λ_π *is defined by (17). Then for every binary experiment* (P,Q) *we have*

$$\mathbb{I}_f(P,Q) = \Delta\underline{\mathbb{L}}(\eta,M) = \mathbb{B}_\phi(\eta,M),$$

where $M := \pi P + (1-\pi)Q$, $\eta := \pi dP/dM$ *and* $\underline{\mathbb{L}}$ *is the expectation (in* X*) of the conditional Bayes risk* $\underline{L}$*. Equivalently,*

$$\mathbb{J}_Q[f(dP/dQ)] = \mathbb{J}_M[-\underline{L}(\eta)] = \mathbb{J}_M[\phi(\eta)].$$

What this says is that for each choice of π the classes of f-divergences $\mathbb{I}_f$, statistical informations $\Delta\underline{\mathbb{L}}$ and (discriminative) Bregman informations $\mathbb{B}_\phi$ can all be defined in terms of the Jensen gap of some convex function. Additionally, there is a bijection between each of these classes due to the mapping λ_π that identifies likelihood ratios with posterior probabilities.

The class of f-divergences is "more primitive" than the other measures since its definition does not require the extra structure that is obtained by assuming that the reference measure M can be written as the convex combination of the distributions P and Q. Indeed, each $\mathbb{I}_f$ is invariant to the choice of reference measure and so is invariant to the choice of π. The results in the next section provide another way of looking at this invariance of $\mathbb{I}_f$. In particular, we see that every f-divergence is a weighted "average" of statistical informations or, equivalently, $\mathbb{I}_{f^\pi}$ divergences.

5. Primitives and Weighted Integral Representations

When given a class of functions like f-divergences, risks or measures of information it is natural to ask what the "simplest" elements of these classes are. We would like to know which functions are "primitive" in the sense that they can be used to express other measures but themselves cannot be so expressed.

The connections between risk, f-divergence, and statistical information discussed in Section 4 are all in terms of the convex functions that define each type of measurement. As discussed in Section 2.3, integral representations allow these convex functions to be expressed as weighted combinations of simple, convex, piecewise linear functions. By thinking of the set of these simple functions as a "basis" for convex functions, we are able to identify any convex function with its "coordinates"—that is, its weight function—relative to this basis.

The main result of this section essentially "lifts" this weight function representation of convex functions through the definitions of proper risks and f-divergence (and therefore also statistical and Bregman information) so they can be expressed as weighted integrals of primitive elements corresponding to the simple convex functions acting as the "basis". In the case of f-divergences and information the weight function in these integrals completely determines their behaviour. This means the weight functions can be used as a proxy for the analysis of these measures, or as a knob the user can adjust in choosing what to measure.

We also show that the close relationships between information and f-divergence in terms of their convex generators can be directly translated into a relationship between the respective weight functions associated with these measures. That is, given the weight function that determines an f-divergence there is, for each choice of the prior π, a simple transformation that yields the weight function for the corresponding statistical information, and *vice versa*.

This shift from "function as graph of evaluations" to "function as weighted combination of primitive functions" permeates the remainder of the paper and is (loosely!) analogous to the way the Fourier transform represents functions as sums of simple, periodic signals. In Section 6, risk curves are used to graphically summarise the values of all the primitive risks for a given binary experiment. In Section 7, surrogate regret bounds for proper losses and a tight generalisation of Pinsker's inequality are derived by considering the relationship between general regrets or divergences and the primitive ones comprising them. In both cases, the bounds are established by using weight functions to understand the relative contribution of each primitive to the weighted sum. In particular, the Pinkser-like inequalities in Appendix B for specific f-divergences are obtained via direct manipulation of their weight functions.

5.1 Integral Representations of f-divergences

The following result shows that the class of f-divergences (and, by the result of the previous section, statistical and Bregman information) is closed under conic combination.

Theorem 11 *For all convex functions $f_1, f_2 \colon (0,\infty) \to \mathbb{R}$ and all $\alpha_1, \alpha_2 \in [0,\infty)$, the function*

$$(0,\infty) \ni t \mapsto g(t) := \alpha_1 f_1(t) + \alpha_2 f_2(t) \tag{22}$$

is convex. Furthermore, for all distributions P and Q, we have

$$\mathbb{I}_g(P,Q) = \alpha_1 \mathbb{I}_{f_1}(P,Q) + \alpha_2 \mathbb{I}_{f_2}(P,Q). \tag{23}$$

Conversely, given f_1, f_2, α_1 and α_2, if (23) holds for all P and Q then g must be, up to affine additions, of the form (22).

The proof is a straight-forward application of the definition of convexity and of f-divergences.

One immediate consequence of this result is that the set of f-divergences is closed under conic combinations $\sum_i \alpha_i \mathbb{I}_{f_i}$. Furthermore, the arguments in Section 2.4 can be used to extend this observation beyond finite linear combination to generalised weight functions α. By Corollary 2, if f is a convex function then expanding it about 1 in (5) and setting $\alpha(s) = f''(s)$ means that

$$\mathbb{I}_f(P,Q) = \int_0^\infty \mathbb{I}_{F_s}(P,Q)\,\alpha(s)\,ds \tag{24}$$

where $F_s(t) = [\![s \le 1]\!](s-t)_+ + [\![s > 1]\!](t-s)_+$.[16] The functions F_s, $s \in \mathbb{R}^+$ can therefore be seen as the generators of the class of primitive f-divergences. As a function of t, each F_s is piecewise linear, with a single "hinge" at s. Of course, any affine translation of any F_s is also a primitive. In fact, each F_s may undergo a different affine translation without changing the f-divergence $\mathbb{I}_f$. The weight function α is what completely characterises the behaviour of $\mathbb{I}_f$.

The integral in (24) need not always exist since the integrand may not be integrable. When the Cauchy Principal Value diverges we say the integral takes on the value ∞. We note that many (not all) f-divergences can sometimes take on infinite values.

The integral form in (24) can be readily transformed into an integral representation that does not involve an infinite integrand. This is achieved by mapping the interval $[0,\infty)$ onto $[0,1)$ via the change of variables $\pi = \frac{1}{1+s} \in [0,1]$. In this case, $s = \frac{1-\pi}{\pi}$ and so $ds = -\frac{d\pi}{\pi^2}$ and the integral of (24) becomes

$$\begin{aligned} \mathbb{I}_f(P,Q) &= -\int_1^0 \mathbb{I}_{F_{\frac{1-\pi}{\pi}}}(P,Q)\,\alpha(\tfrac{1-\pi}{\pi})\,\pi^{-2}\,d\pi \\ &= \int_0^1 \mathbb{I}_{\tilde{f}_\pi}(P,Q)\,\gamma(\pi)\,d\pi \end{aligned} \tag{25}$$

where

$$\tilde{f}_\pi(t) := \pi F_{\frac{1-\pi}{\pi}}(t) = \begin{cases} (1-\pi(1+t))_+\,, & \pi \ge \frac{1}{2} \\ (\pi(1+t)-1)_+\,, & \pi < \frac{1}{2} \end{cases} \tag{26}$$

16. Technically, one must assume that f is twice differentiable for this result to hold. However, the convexity of f implies it has well-defined one-sided derivatives f'_+ and $\alpha(s)$ can be expressed as the measure corresponding to $df'_+/d\lambda$ for the Lebesgue measure λ. Details can be found in Liese and Vajda (2006). The representation of a general f-divergence in terms of elementary ones is not new; see for example Österreicher and Feldman (1981) and Feldman and Österreicher (1989).

and

$$\gamma(\pi) := \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right).$$

This observation forms the basis of the following restatement of a theorem by Liese and Vajda (2006). We include it here with a short proof to discuss the connection between f-divergences and statistical information.[17]

Theorem 12 *Let f be convex such that $f(1) = 0$. Then there exists a (generalised) function $\gamma : (0,1) \to \mathbb{R}$ such that, for all P and Q:*

$$\mathbb{I}_f(P,Q) = \int_0^1 \mathbb{I}_{f_\pi}(P,Q)\,\gamma(\pi)\,d\pi, \text{ where } f_\pi(t) = (1-\pi)\wedge\pi - (1-\pi)\wedge(\pi t).$$

Proof The earlier discussion giving the derivation of Equation (25) implies the result. The only discrepancy is over the form of f_π. We determine the precise form by noting that the family of $\tilde{f}_\pi$ given in (26) can be transformed by affine addition without affecting the representation of $\mathbb{I}_f$. Specifically,

$$\begin{aligned} f_\pi(t) &:= (1-\pi)\wedge\pi - (1-\pi)\wedge(\pi t) \\ &= \begin{cases} (1-\pi(1+t))_+\,, & \pi \geq \frac{1}{2} \\ (\pi(1+t)-1)_+ + \pi(1-t)\,, & \pi < \frac{1}{2} \end{cases} \\ &= \tilde{f}_\pi(t) + [\![\pi < \tfrac{1}{2}]\!]\pi(1-t), \end{aligned}$$

and so $\tilde{f}_\pi$ and f_π are in the same affine equivalence class for each $\pi \in [0,1]$. Thus, by Theorem 6 we have $\mathbb{I}_{f_\pi} = \mathbb{I}_{\tilde{f}_\pi}$ for each $\pi \in [0,1]$, proving the result. ■

The specific choice of f_π in the above theorem from all of the affine equivalents was made to make simpler the connection between integral representations for losses and f-divergences, discussed in Section 5.4.

One can easily verify that f_π are convex hinge functions of t with a hinge at $\frac{1-\pi}{\pi}$ and $f_\pi(1) = 0$. Thus $\{\mathbb{I}_{f_\pi}\}_{\pi\in(0,1)}$ is a family of primitive f-divergences; confer Österreicher and Feldman (1981) and Feldman and Österreicher (1989). This theorem implies an existing representation of f-divergences due to Österreicher and Vajda (1993, Theorem 1) and Gutenbrunner (1990). They show that an f-divergence can be represented as a weighted integral of statistical informations for 0-1 loss: for all P, Q

$$\mathbb{I}_f(P,Q) = \int_0^1 \Delta\underline{\mathbb{L}}^{0-1}(\pi,P,Q)\gamma(\pi)d\pi, \tag{27}$$

$$\gamma(\pi) = \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right). \tag{28}$$

An f divergence is *symmetric* if $\mathbb{I}_f(P,Q) = \mathbb{I}_f(Q,P)$ for all P, Q. The representation of $\mathbb{I}_f$ in terms of γ and Theorem 15 provides an easy test for symmetry:

17. The $1/\pi^3$ term in the definition of γ seems a little unusual at first glance. However, it is easily understood as the product of two terms: $1/\pi^2$ from the second derivative of $(1-\pi)/\pi$, and $1/\pi$ from a transformation of variables within the integral to map the limits of integration from $(0,\infty)$ to $(0,1)$ via λ_π.

Corollary 13 *Suppose $\mathbb{I}_f$ is an f-divergence with corresponding weight function γ given by (28). Then $\mathbb{I}_f$ is symmetric iff $\gamma(\pi) = \gamma(1-\pi)$ for all $\pi \in [0,1]$.*

The proof is in Appendix A.4.

Corollary 13 provides a way of generating all convex f such that $\mathbb{I}_f$ is symmetric that is simpler than that proposed by Hiriart-Urruty and Martínez-Legaz (2007): let $\gamma(\pi) = \beta(\pi \wedge (1-\pi))$ where $\beta \in (\mathbb{R}^+)^{[0,\frac{1}{2}]}$ (i.e., all symmetric weight functions) and generate f from γ by inverting (28); explicitly,

$$f(s) = \int_0^s \left(\int_0^t \frac{1}{(\tau+1)^3} \gamma\left(\frac{1}{\tau+1}\right) d\tau \right) dt, \quad s \in \mathbb{R}^+.$$

5.2 Proper Losses and Cost-Weighted Risk

We now consider a representation of proper losses in terms of primitive losses that originates with Shuford et al. (1966). Our discussion follows that of Buja et al. (2005) and then examines its implications in light of the connections between information and divergence just presented.

The *cost-weighted losses* are a family of losses parameterised by a false positive cost $c \in [0,1]$ that defines a loss for $y \in \{\pm 1\}$ and $\hat{\eta} \in [0,1]$ by

$$\ell_c(y, \hat{\eta}) = c[\![y = -1]\!][\![\hat{\eta} \geq c]\!] + (1-c)[\![y = 1]\!][\![\hat{\eta} < c]\!]. \tag{29}$$

Intuitively, a cost-weighted loss thresholds $\hat{\eta}$ at c and assigns a cost if the resulting classification disagrees with y. These correspond to the "signatures" for eliciting the probability η as described by Lambert et al. (2008). Substituting $c = \frac{1}{2}$ will verify that $2\ell_{\frac{1}{2}}$ is equivalent to 0-1 misclassification loss ℓ^{0-1}. Taking expectations with respect to Y we have

$$L_c(\eta, \hat{\eta}) = (1-\eta)c[\![\hat{\eta} \geq c]\!] + \eta(1-c)[\![\hat{\eta} < c]\!]. \tag{30}$$

We will use L_c, $\mathbb{L}_c$ and $\Delta\underline{\mathbb{L}}_c$ to denote the cost-weighted point-wise risk, full risk and statistical information associated with each cost-weighted loss. The following theorems collect some useful observations about these primitive quantities. The first shows that the point-wise Bayes risk is a simple, concave "tent" function. The second shows that cost-weighted statistical information is invariant under the switching of the classes provided the costs are also switched and that π and $1-c$ are interchangeable.

Theorem 14 *For all $\eta, c \in [0,1]$ the point-wise Bayes risk $\underline{L}_c(\eta) = (1-\eta)c \wedge (1-c)\eta$ and is therefore concave in both c and η.*

Proof From the definition of ℓ_c in Equation 29 and the definition of point-wise Bayes risk, we have, for $\eta \in [0,1]$,

$$\begin{aligned}
\underline{L}_c(\eta) &= \inf_{\hat{\eta} \in [0,1]} L_c(\eta, \hat{\eta}) \\
&= \inf_{\hat{\eta} \in [0,1]} \{(1-\eta)c[\![\hat{\eta} \geq c]\!] + \eta(1-c)[\![\hat{\eta} < c]\!]\} \\
&= \inf_{\hat{\eta} \in [0,1]} \{\eta(1-c) + (c-\eta)[\![\hat{\eta} \geq c]\!]\},
\end{aligned}$$

where the last step makes use of the identity $[\![\hat{\eta} < c]\!] = 1 - [\![\hat{\eta} \geq c]\!]$. Since $(c-\eta)$ is negative if and only if $\eta > c$, the infimum is obtained by having $[\![\hat{\eta} \geq c]\!] = 1$ if and only if $\eta \geq c$, that is, by letting

$\hat{\eta} = \eta$. In this case, when $\hat{\eta} \geq c$ we have $\underline{L}_c(\eta) = c(1-\eta)$ and when $\hat{\eta} < c$ we have $\underline{L}_c(\eta) = (1-c)\eta$. The concavity of $\underline{L}_c$ is evident as this function is the minimum of two linear functions of c and η. ■

Theorem 15 *For all $c \in [0,1]$ and tasks $(\eta, M; \ell_c) = (\pi, P, Q; \ell_c)$ the statistical information satisfies 1)*

$$\Delta\underline{\mathbb{L}}_c(1-\eta, M) = \Delta\underline{\mathbb{L}}_{1-c}(\eta, M),$$

or equivalently,

$$\Delta\underline{\mathbb{L}}_c(1-\pi, Q, P) = \Delta\underline{\mathbb{L}}_{1-c}(\pi, P, Q);$$

and 2)

$$\Delta\underline{\mathbb{L}}_\pi(1-c, P, Q) = \Delta\underline{\mathbb{L}}_c(1-\pi, P, Q).$$

Proof By Theorem 14 we know $\underline{L}_c(\eta) = \min\{(1-\eta)c, (1-c)\eta\}$ and so $\underline{L}_c(1-\eta) = \underline{L}_{1-c}(\eta)$ for all $\eta, c \in [0,1]$. Therefore, $\underline{\mathbb{L}}_c(1-\eta, M) = \underline{\mathbb{L}}_{1-c}(\eta, M)$ for any $\eta : \mathcal{X} \to [0,1]$ including the constant function $\mathbb{E}_M[\eta]$. By definition, $\Delta\underline{\mathbb{L}}_c(\eta, M) = \underline{\mathbb{L}}(\mathbb{E}_M[\eta], M) - \underline{\mathbb{L}}(\eta, M)$ and so $\Delta\underline{\mathbb{L}}_{1-c}(\eta, M) = \Delta\underline{\mathbb{L}}_c(1-\eta, M)$ proving part 1.

Part 2 also follows from Theorem 14 by noting that $\underline{L}_c(1-\pi) = \underline{L}_\pi(1-c)$ and $\mathbb{E}_M[\underline{L}_c(\eta)] = \int_{\mathcal{X}} \min\{(1-c)\pi\, dP, (1-\pi)c\, dQ\}$. ■

5.3 Integral Representations of Proper Losses

The cost-weighted losses are primitive in the sense that they form the basis for a Choquet integral representation of proper losses. This representation is essentially a consequence of Taylor's theorem and was originally studied by Shuford et al. (1966) and later generalised by Schervish (1989). The recent presentation of this result by Lambert et al. (2008) gives yet a more general formulation in terms of the elicitability of properties of distributions, along with a geometric derivation. An historical summary of decompositions of scoring rules is given by Winkler et al. (1990, Section 4).

Theorem 16 *Let $\ell : \mathcal{Y} \times [0,1] \to \mathbb{R}$ be a fair, proper loss. Then for each $\hat{\eta} \in (0,1)$ and $y \in \mathcal{Y}$*

$$\ell(y, \hat{\eta}) = \int_0^1 \ell_c(y, \hat{\eta})\, w(c)\, dc \tag{31}$$

where the weight function[18] *$w : (0,1) \to \mathbb{R}^+$ satisfies*

$$w(c) = -\underline{L}''(c) \geq 0 \tag{32}$$

for all $c \in (0,1)$. Conversely, if ℓ is defined by (31) for some weight function $w : (0,1) \to \mathbb{R}^+$ then it is proper.

The proof is almost a direct consequence of Taylor's theorem.

18. The weight function and second derivative of $-\underline{L}$ are to be interpreted distributionally as discussed in Section 2.4.

Proof We first assume ℓ is a proper loss so that $L(\eta,\hat{\eta}) = \mathbb{E}_{Y\sim\eta}[\ell(Y,\hat{\eta})]$ and $\underline{L}(\eta) = L(\eta,\eta)$. Expanding $\underline{L}(\eta)$ about $\hat{\eta} \in (0,1)$ using Corollary 2 yields

$$\begin{aligned}\underline{L}(\eta) &= \underline{L}(\hat{\eta}) + (\eta-\hat{\eta})\underline{L}'(\hat{\eta}) + \int_0^1 \phi_c(\eta,\hat{\eta})\,\underline{L}''(c)\,dc \\ &= L(\eta,\hat{\eta}) + \int_0^1 \phi_c(\eta,\hat{\eta})\,\underline{L}''(c)\,dc \end{aligned} \tag{33}$$

by Theorem 7. The generalised function $w(c) = -\underline{L}''(c) \geq 0$ by the concavity of $\underline{L}$. Rearranging (33) gives

$$L(\eta,\hat{\eta}) = \underline{L}(\eta) + \int_0^1 \phi_c(\eta,\hat{\eta})\,w(c)\,dc.$$

The definition of L in (18) implies $L(y,\hat{\eta}) = \ell(y,\hat{\eta})$ for $y \in \{0,1\}$ and so

$$\ell(y,\hat{\eta}) = \underline{L}(y) + \int_0^1 \phi_c(y,\hat{\eta})\,w(c)\,dc, \tag{34}$$

where

$$\phi_c(y,\hat{\eta}) = [\![\hat{\eta} \leq c < y]\!](y-c) + [\![y \leq c < \hat{\eta}]\!](c-y),$$

which is equal to the definition of ℓ_c in (29) since the left (resp. right) term is only non-zero when $y = 1$ (resp. $y = 0$). Observe that $\underline{L}(0) = \underline{L}(1) = 0$ since $\underline{L}(0) = L(0,0) = \ell(0,0) = 0$ by the assumption that the loss is fair, and similarly for $\underline{L}(1)$.

This shows that (34) is equivalent to (31), completing the forward direction of the theorem.

If we now assume the function $w \geq 0$ is given and ℓ defined as in (31) then it suffices to show $\underline{L}(\eta) = L(\eta,\eta)$. First note that

$$\begin{aligned} L(\eta,\hat{\eta}) &= \mathbb{E}_{Y\sim\eta}\left[\int_0^1 \ell_c(Y,\hat{\eta})\,w(c)\,dc\right] \\ &= \int_0^1 L_c(\eta,\hat{\eta})\,w(c)\,dc. \end{aligned}$$

Each of the L_c are proper and so are minimised when $\hat{\eta} = \eta$. Since $w(c) \geq 0$ this must also be sufficient to minimise L. ■

We will write ℓ_w, L_w and $\mathbb{L}_w$ to explicitly indicate the parameterisation of the loss, conditional loss and expected loss by the weight function w. A proper loss ℓ_w corresponding to a given weight function can be explicitly derived using the following theorem.

Theorem 17 *Given a weight function $w : [0,1] \to \mathbb{R}^+$, let $W(t) = \int^t w(c)\,dc$ and $\overline{W}(t) = \int^t W(c)\,dc$. Then the loss ℓ_w defined by*

$$\ell_w(y,\hat{\eta}) = -\overline{W}(\hat{\eta}) - (y-\hat{\eta})W(\hat{\eta})$$

is a proper loss. Additionally, if $\overline{W}(0)$ and $\overline{W}(1)$ are both finite then

$$(y,\hat{\eta}) \mapsto \ell_w(y,\hat{\eta}) + (\overline{W}(1) - \overline{W}(0))y + \overline{W}(0) \tag{35}$$

is a fair, proper loss.

Proof First we define the loss

$$\ell(y,\hat{\eta}) := \int_0^1 \ell_c(y,\hat{\eta})\, w(c)\, dc$$

and proceed to show it is equal to the definition of ℓ_w. Theorem 16 guarantees that ℓ is proper and that $w = -\underline{L}''$. By definition of the improper integrals $\overline{W}$ and W and the fundamental theorem of calculus we know that $W' = w = -\underline{L}''$ and so $\overline{W}'(t) = W(t) = -\underline{L}'(t) + a$ and

$$\overline{W}(t) = -\underline{L}(t) + at + b, \tag{36}$$

where $a, b \in \mathbb{R}$ are constants of integration. Substituting these into the Savage representation of Theorem 7 for proper losses we see that

$$\begin{aligned} L(\eta,\hat{\eta}) &= \underline{L}(\hat{\eta}) + (\eta - \hat{\eta})\underline{L}'(\hat{\eta}) \\ &= -\overline{W}(\hat{\eta}) + a\hat{\eta} + b + (\eta - \hat{\eta})[-W(\hat{\eta}) + a] \\ &= -\overline{W}(\hat{\eta}) - (\eta - \hat{\eta})W(\hat{\eta}) + a\eta + b. \end{aligned}$$

Since $L(y,\hat{\eta}) = \ell(y,\hat{\eta})$ for $y \in \{0,1\}$ we have $\ell(0,\hat{\eta}) = \ell_w(0,\hat{\eta}) + b$ and $\ell(1,\hat{\eta}) = \ell_w(1,\hat{\eta}) + a + b$ for all $a, b \in \mathbb{R}$. Choosing $a = b = 0$ achieves the result.

If $\overline{W}(0)$ and $\overline{W}(1)$ are both finite then letting $a = \overline{W}(1) - \overline{W}(0)$ and $b = \overline{W}(0)$ means (36) implies $\overline{W}(0) = -\underline{L}(0) + \overline{W}(0)$ and so $\underline{L}(0) = 0$. Similarly, $\underline{L}(1) = 0$ showing that (35) is fair. ■

As an example of how this theorem lets us explicitly construct proper losses from weight functions, consider the weight function $w(c) = 1$. In this case, $W(t) = t$ and $\overline{W}(t) = \frac{t^2}{2}$. Thus, noting that $y^2 = y$ for $y \in \{0,1\}$ we have

$$\ell_w(y,\hat{\eta}) = -\tfrac{1}{2}\hat{\eta}^2 - (y - \hat{\eta})\hat{\eta} + \tfrac{1}{2}y = \tfrac{1}{2}(\hat{\eta} - y)^2$$

which is the square loss.

As a second example, consider $w(c) = \frac{1}{(1-c)c}$. In this case, $W(t) = \ln\left(\frac{t}{1-t}\right)$ and $\overline{W}(t) = (1-t)\ln(1-t) + t\ln(t)$. Since $\lim_{\varepsilon \to 0} \varepsilon \ln(\varepsilon) = 0$ we define $0\ln(0) := 0$ so that $b = \overline{W}(0) = 0$ and $a = \overline{W}(1) - \overline{W}(0) = 0$. This implies

$$\begin{aligned} \ell_w(y,\hat{\eta}) &= -(1-\hat{\eta})\ln(1-\hat{\eta}) - \hat{\eta}\ln(\hat{\eta}) - (y - \hat{\eta})\ln\left(\frac{\hat{\eta}}{1-\hat{\eta}}\right) \\ &= [-(1-\hat{\eta}) + (y-\hat{\eta})]\ln(1-\hat{\eta}) + [-\hat{\eta} - (y-\hat{\eta})]\ln(\hat{\eta}) \\ &= -(1-y)\ln(1-\hat{\eta}) - y\ln(\hat{\eta}) \end{aligned}$$

which is log loss.

5.4 Relating Integral Representations for $\mathbb{L}$ and $\mathbb{I}_f$

There is also the following direct relationship between the weight functions γ for an f-divergence and w for the corresponding statistical information. Since the weight functions are an attractive parameterization, it is convenient to be able to directly translate between the two respective weight functions. The proof is in Appendix A.5.

Theorem 18 *Let $f\colon \mathbb{R}^+ \to \mathbb{R}$ be convex (with $f(1)=0$) define $\mathbb{I}_f$ with corresponding weight function γ. Then for each $\pi \in (0,1)$ the weight function w^π in Theorem 16 for the loss ℓ^π given by Theorem 9 satisfies*

$$w^\pi(c) = \frac{\pi(1-\pi)}{\nu(\pi,c)^3}\gamma\left(\frac{(1-c)\pi}{\nu(\pi,c)}\right)$$

or, inversely,

$$\gamma(c) = \frac{\pi^2(1-\pi)^2}{\nu(\pi,c)^3} w\left(\frac{\pi(1-c)}{\nu(\pi,c)}\right),$$

where $\nu(\pi,c) = (1-c)\pi + (1-\pi)c$.

The representation (27,28) allows the determination of weights for standard f-divergences. Kullback-Liebler divergence $\mathrm{KL}(P,Q)$ corresponds to $\gamma(\pi) = \frac{1}{\pi^2(1-\pi)}$. Thus $J(P,Q) = \mathrm{KL}(P,Q) + \mathrm{KL}(Q,P)$ corresponds to $\gamma(\pi) = \frac{1}{\pi^2(1-\pi)^2}$. Several f-divergences are presented with their corresponding weight function in Table 2. The weight for $\mathrm{KL}(P,Q)$ has a double pole at $\pi = 0$ which is why KL-divergence is hard to estimate—it puts a lot of weight on $\Delta\underline{\mathbb{L}}^{0-1}(\pi, PQ)$ for $\pi \approx 0$ which by Theorem 15 means a lot of weight on $\Delta\underline{\mathbb{L}}_c(\frac{1}{2})$ for $c \approx 1$ which requires a good estimate of $\mathbb{L}_c(\eta, M)$ which is difficult with modest data sample sizes.[19]

A loss function corresponding to each f-divergence in Table 2 is also shown. The weight function $w(c)$ for the loss is for the case when $\pi = \frac{1}{2}$, that is, it is a loss for a binary classification problem with equal proportions of positive and negative examples. In this case, the relationship between w and γ simplifies to $w^{\frac{1}{2}}(c) = 2\gamma(1-c)$ since $\nu(\frac{1}{2}, c) = \frac{1}{2}c + \frac{1}{2}(1-c) = \frac{1}{2}$.

The entries in Table 2 without a name for the loss correspond to losses that are not definite. It turns out that weight functions whose tail behaviour is not $o(c^{-2})$ or $o((1-c)^{-2})$ as c goes to 0 or 1, respectively (confer Buja et al., 2005, §6) imply non-definiteness of a proper loss.

5.5 Example—Squared Loss

We illustrate some of the above concepts with a simple example. Consider squared loss. We have

$$L(\eta, \hat\eta) = \hat\eta^2(1-\eta) + (\hat\eta - 1)^2\eta$$

and thus $\underline{L}(\eta) = L(\eta,\eta) = \eta(1-\eta)$ and $\underline{L}''(\eta) = -2$ and thus by (32) $w(\eta) = 2$. From (21) we thus have

$$f^\pi(t) = \frac{\pi(1-\pi)(\pi t + 1 - \pi) - (1-\pi)\pi t}{\pi t + 1 - \pi}.$$

Choosing $\pi = \frac{1}{2}$ this becomes $f^{\frac{1}{2}}(t) = \frac{1-t}{4t+4}$. One can check that $8 \cdot f^{\frac{1}{2}}(t) + t - 1 = \frac{(t-1)^2}{t+1}$ which agrees with the f corresponding to Triangular Discrimination in Table 2. Scaling is just a question of normalisation and we have already seen that $\mathbb{I}_f$ is insensitive to affine offsets in f. This illustrates the awkwardness of parameterising $\mathbb{I}_f$ in terms of f: at first sight $\frac{1-t}{4t+4}$ and $\frac{(t-1)^2}{t+1}$ seem quite

19. Considering KL-divergence from the weight function perspective suggests a scheme to estimate it: avoid attempting to estimate the regions near zero and one where the weight function diverges. A particular example of this is the divergence $\mathrm{KL}_\varepsilon(P,Q)$ which has weight function $\gamma(\pi) = \frac{1}{\pi^2(1-\pi)}[\![\pi \in [\varepsilon, 1-\varepsilon]]\!]$. The corresponding f can be worked out but has the rather less intuitively clear form $f(t) = [\![t < \frac{\varepsilon}{1-\varepsilon}]\!](t(\ln(\frac{\varepsilon}{1-\varepsilon}) + 1) - \frac{\varepsilon}{1-\varepsilon}) + [\![\frac{\varepsilon}{1-\varepsilon} \le t \le \frac{1-\varepsilon}{\varepsilon}]\!] t\ln t + [\![\frac{1-\varepsilon}{\varepsilon} < t]\!](t(\ln(\frac{1-\varepsilon}{\varepsilon}) + 1) - \frac{1-\varepsilon}{\varepsilon})$, $\varepsilon \in [0,1)$. This approach to regularizing the estimation of the KL-divergence was suggested by Gutenbrunner (1990, page 454).

Symbol	Divergence	$f(t)$	$\gamma(\pi)$	$w^{\frac{1}{2}}(c)$	$\ell(0,\hat{\eta})$	$\ell(1,\hat{\eta})$	Loss
$V(P,Q)$	Variational	$\lvert t-1\rvert$	$16\delta\left(\pi-\frac{1}{2}\right)$	$32\delta\left(\frac{1}{2}-c\right)$	$16[\![\hat{\eta}>\frac{1}{2}]\!]$	$16[\![\hat{\eta}\leq\frac{1}{2}]\!]$	0-1
$\mathrm{KL}(P,Q)$	Kullback-Leibler	$t\ln t$	$\frac{1}{\pi^2(1-\pi)}$	$\frac{2}{(1-c)^2c}$	$2\left[2\ln(1-\hat{\eta})+\frac{\hat{\eta}}{1-\hat{\eta}}\right]$	$2\left[\ln\frac{1-\hat{\eta}}{\hat{\eta}}-1\right]$	—
$\Delta(P,Q)$	Triangular Discrimination	$(t-1)^2/(t+1)$	8	16	$8\hat{\eta}^2$	$8(1-\hat{\eta})^2$	Square
$\mathrm{I}(P,Q)$	Jensen-Shannon	$\frac{t}{2}\ln\left(\frac{t}{t+1}\right)-\frac{1}{2}\ln\left(\frac{t+1}{4}\right)$	$\frac{1}{2\pi(1-\pi)}$	$\frac{1}{(1-c)c}$	$-\ln(1-\hat{\eta})$	$-\ln(\hat{\eta})$	Log
$\mathrm{T}(P,Q)$	Arith.-Geo. Mean	$\left(\frac{t+1}{2}\right)\ln\left(\frac{t+1}{2\sqrt{t}}\right)$	$\frac{2\pi^2-2\pi+1}{4\pi^2(1-\pi)^2}$	$\frac{2c^2-2c+1}{2c^2(1-c)^2}$	$\frac{1}{2}\left[\ln((1-\hat{\eta})\hat{\eta})-\frac{1-2\hat{\eta}}{1-\hat{\eta}}\right]$	$\frac{1}{2}\left[\ln((1-\hat{\eta})\hat{\eta})+\frac{1-2\hat{\eta}}{\hat{\eta}}\right]$	—
$\mathrm{J}(P,Q)$	Jeffreys	$(t-1)\ln(t)$	$\frac{1}{\pi^2(1-\pi)^2}$	$\frac{2}{(1-c)^2c^2}$	$2\left[\ln\left(\frac{\hat{\eta}}{1-\hat{\eta}}\right)+\frac{1}{1-\hat{\eta}}\right]$	$2\left[\ln\left(\frac{1-\hat{\eta}}{\hat{\eta}}\right)+\frac{1}{\hat{\eta}}\right]$	—
$h^2(P,Q)$	Hellinger	$(\sqrt{t}-1)^2$	$\frac{1}{2[\pi(1-\pi)]^{3/2}}$	$\frac{1}{[(1-c)c]^{3/2}}$	$2\sqrt{\frac{\hat{\eta}}{1-\hat{\eta}}}$	$2\sqrt{\frac{1-\hat{\eta}}{\hat{\eta}}}$	Boosting
$\chi^2(P,Q)$	Pearson χ^2	$(t-1)^2$	$\frac{2}{\pi^3}$	$\frac{4}{(1-c)^3}$	$2\frac{2\hat{\eta}-1}{(1-\hat{\eta})^2}$	$\frac{4}{1-\hat{\eta}}$	—
$\Psi(P,Q)$	Symmetric χ^2	$\frac{(t-1)^2(t+1)}{t}$	$\frac{2}{\pi^3}+\frac{2}{(1-\pi)^3}$	$\frac{4}{(1-c)^3}+\frac{4}{c^3}$	$2\frac{3\hat{\eta}-2}{(1-\hat{\eta})^2\hat{\eta}}$	$2\frac{1-3\hat{\eta}}{(1-\hat{\eta})\hat{\eta}^2}$	—

Table 2: Divergences and their corresponding functions f and weights γ along with the weights w and partial losses—see Section 5.4 ; confer Taneja (2005a); Liese and Vajda (2006). Topsøe (2000) calls $C(P,Q)=2\mathrm{I}(P,Q)$ and $\tilde{C}(P,Q)=2\mathrm{T}(P,Q)$ the Capacitory and Dual Capacitory discrimination respectively. Several of the above divergences are "symmetrised" versions of others. For example, $T(P,Q)=\frac{1}{2}[\mathrm{KL}(\frac{P+Q}{2},P)+\mathrm{KL}(\frac{P+Q}{2},Q)]$, $I(P,Q)=\frac{1}{2}[\mathrm{KL}(P,\frac{P+Q}{2})+\mathrm{KL}(Q,\frac{P+Q}{2})]$, $\mathrm{J}(P,Q)=\mathrm{KL}(P,Q)+\mathrm{KL}(Q,P)$, and $\Psi(P,Q)=\chi^2(P,Q)+\chi^2(Q,P)$. The Boosting loss is also know as the "exponential" loss (Buja et al., 2005). Losses without a name are all indefinite losses and the forms given for $\ell(y,\cdot)$ in these cases are not normalised.

.

different. Using weight functions automatically filters out the effect of any affine offsets—if the weight functions corresponding to f_1 and f_2 match, then $\mathbb{I}_{f_1} = \mathbb{I}_{f_2}$. Finally observe that substituting $\gamma(\pi) = 8$ from the table into Theorem 18 we obtain $w^{\frac{1}{2}}(c) = \frac{1/4}{\nu(\pi,c)^3} \cdot 8 = 2$, consistent with the weight obtained above.

6. Graphical Representations

The last section described representations of risks and f-divergences in terms of weighted integrals of primitive functions. The values of the primitive functions lend themselves to a graphical interpretation that is explored in this section. In particular, a diagram called a *risk curve* is introduced. Risk curves are a useful aid to intuition when reasoning about risks, divergences and information and they are used in Section 7 to derive bounds between various divergences and risks.

Risk curves are closely related to the *cost curves* of Drummond and Holte (2006) as well as idealised *receiver operating characteristic, or ROC curves* (Fawcett, 2004). Proposition 20 makes this latter relationship explicit via a point-line duality between risk and ROC curves. Additionally, results about the Neyman-Pearson function by Torgersen (1981) allow us to establish a transformation between suitably smooth maximal ROC and minimal risk curves in Theorem 22. Despite the close ties between f-divergences and risks, and between risk curves and ROC curves, we show in Proposition 19 that the *area* under an ROC curve cannot be interpreted as an f-divergence.

6.1 ROC Curves

Plotting a *receiver operating characteristic curve* or *ROC curve* is a way of graphically summarising the performance of a test statistic. Recall from Section 3.1 that in the context of a binary experiment (P,Q) on a space $\mathcal{X}$, a test statistic τ is any function that maps points in $\mathcal{X}$ to the real line. Each choice of threshold $\tau_0 \in \mathbb{R}$ results in a classifier $r(x) = [\![\tau(x) \geq \tau_0]\!]$ and its corresponding classification rates. An ROC curve for the test statistic τ is simply a plot of the true positive rate of these classifiers as a function of their false positive rate as the threshold τ_0 varies over $\mathbb{R}$. Formally,

$$\mathrm{ROC}(\tau) := \{(\mathrm{FP}_\tau(\tau_0), \mathrm{TP}_\tau(\tau_0)) : \tau_0 \in \mathbb{R}\} \subset [0,1]^2. \tag{37}$$

A graphical example of an ROC curve is shown as the solid black line in Figure 3.

For a fixed experiment (P,Q), the Neyman-Pearson lemma provides an upper envelope for ROC curves. It guarantees that the ROC curve for the likelihood ratio $\tau^* = dP/dQ$ will lie above, or *dominate*, that of any other test statistic τ as shown in Figure 3. This is an immediate consequence of the likelihood ratio being the most powerful test since for each false positive rate (or size) α it will have the largest true positive rate (or power) β of all tests (Eguchi and Copas, 2001). Thus $\mathrm{ROC}(dP/dQ)$ is the *maximal* ROC curve.

The performance of a test statistic τ shown in an ROC curve is commonly summarised by the *Area Under the ROC Curve*, $\mathrm{AUC}(\tau)$, and is closely related to the Mann-Whitney-Wilcoxon statistic. Formally, if (P,Q) is a binary experiment and τ a test statistic the AUC is

$$\begin{aligned} \mathrm{AUC}(\tau) &:= \int_0^1 \beta_\tau(\alpha)\, d\alpha & (38)\\ &= \int_{-\infty}^{\infty} \mathrm{TP}_\tau(\tau_0)\, \mathrm{FP}'_\tau(\tau_0)\, d\tau_0, & (39) \end{aligned}$$

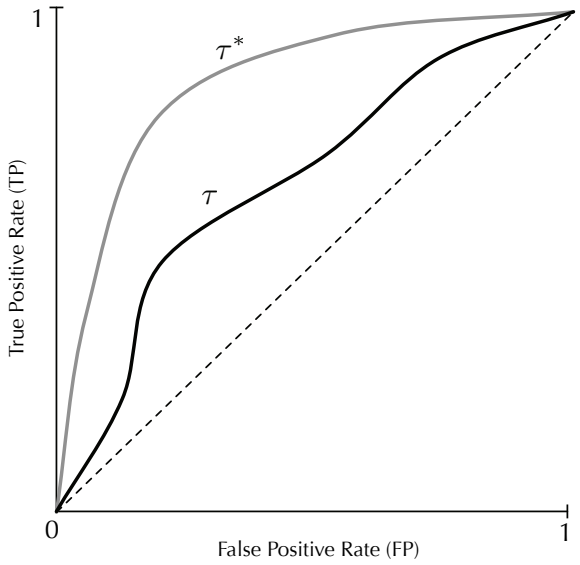


Figure 3: Example of an ROC diagram showing an ROC curve for an arbitrary statistical test τ (middle, bold curve) as well as an optimal statistical test τ^* (top, grey curve). The dashed line represents the ROC curve for a random, or uninformative statistical test.

where $\beta_\tau(\alpha) = \mathrm{TP}_\tau(\tau_0)$ for a $\tau_0 \in \mathbb{R}$ such that $\mathrm{FP}_\tau(\tau_0) = \alpha$.

In Section 3.1 the Neyman-Pearson lemma was used to argue that the curve $\beta(\alpha)$ for the likelihood ratio dominates all other curves. Since the likelihood ratio is used to define f-divergences, it is natural to ask whether the area under the maximal ROC curve is an f-divergence. Interestingly, the answer is "no".

Proposition 19 *There is no convex f such that $\mathbb{I}_f(P,Q) = \mathrm{AUC}(dP/dQ)$ for all distributions P and Q.*

Proof Note that an f-divergence's integral can be decomposed as follows

$$\mathbb{I}_f(P,Q) = \int_0^\infty f(t) \int_{\mathcal{X}_t} dQ\, dt, \tag{40}$$

where $\mathcal{X}_t := \{x \in \mathcal{X} : dP/dP(x) = t\} = (dP/dQ)^{-1}(t)$. Compare this to the definition of $\mathrm{AUC}(\tau)$ given in (39) when $\tau = dP/dQ$

$$\begin{aligned} \mathrm{AUC}(dP/dQ) &= \int_{-\infty}^\infty \mathrm{TP}_\tau(t)\, \mathrm{FP}'_\tau(t)\, dt \\ &= -\int_0^\infty (P \circ \tau^{-1})([t,\infty)) \int_{\mathcal{X}_t} dQ\, dt \end{aligned} \tag{41}$$

since $\mathrm{FP}'_\tau(t) = d/dt \int_t^\infty \int_{\mathcal{X}_t} dQ(x)\, dt = -\int_{\mathcal{X}_t} dQ$ and $dP/dQ \geq 0$. If we assume there exists an f such that for all binary experiments (P,Q), $\mathbb{I}_f(P,Q) = \mathrm{AUC}(dP/dQ)$ we would require the integrals in (40) and (41) to be equal for all (P,Q). This would require $f(t) = -(P \circ (dP/dQ)^{-1})([t,\infty))$ for

all $t \in [0,\infty)$ which is not possible for all binary experiments (P,Q) simultaneously. ■

Although the maximal AUC for (P,Q) cannot be expressed as an f-divergence, Torgersen (1991) shows how it can be expressed as the variational divergence between the *product measures* $P \times Q$ and $Q \times P$. That is, $\mathrm{AUC}(dP/dQ) = V(P \times Q, Q \times P)$. Following up this connection and considering other f-divergences of product measures is left as future work.

It is important to realise that AUC is not a particularly intrinsic measure—just a common one. As the earlier discussion of integral representations has shown, there is value in considering weighted versions of integrals such as (38). As Hand (2008) notes in his commentary on a recent paper (outlining another type of performance curve): "To use all the values of the diagnostic instrument, when integrating to yield the overall AUC measure, it is necessary to decide what weight to give to each value in the integration. The AUC implicitly does this using a weighting derived empirically from the data." Along these lines, Xie and Priebe (2002) and Eguchi and Copas (2001) have suggested generalisations of the AUC that incorporates weights and show that certain choice of weight functions yield well-known losses.

A closer investigation of these generalisations of AUC and their connection to measures of divergence is also left as future work.

6.2 Risk Curves

Risk curves are a graphical representation closely related to ROC curves that take into account a prior π in addition to the binary experiment (P,Q). They provide a concise summary of the risk of an estimator $\hat{\eta}$ for the full range of costs $c \in [0,1]$ for a fixed prior $\pi \in [0,1]$, or, alternatively, for the full range of priors π given a fixed cost c.

A *risk curve for costs* for the estimator $\hat{\eta}$ is the set $\{(c, \mathbb{L}_c(\hat{\eta},\pi,P,Q)) : c \in [0,1]\}$ of points parameterised by cost.[20] A *risk curve for priors* for the estimator $\hat{\eta}$ is the set $\{(\pi, \mathbb{L}^{0\text{-}1}(\hat{\eta},\pi,P,Q)) : \pi \in [0,1]\}$.

Figure 4 shows an example of a *risk curve diagram*. On it is plotted the cost curves for an estimate $\hat{\eta}$ of a true posterior η on the same graph. The "tent" function also shown is the risk curve for the majority class predictor $\min\{(1-\pi)c, (1-c)\pi\}$. Here $\pi = \frac{1}{2}$. Other choices of $\pi \in (0,1)$ skew the tent and the curves under it towards 0 or 1.

In light of the weighted integral representations described in Theorem 16, several of the quantities can be associated with properties of a cost curve diagram. The weight function $w(c)$ associated with a loss ℓ can be interpreted as a weighting on the horizontal axis of a risk curve diagram. When the area under a risk curve is computed with respect to this weighting the result is the full risk $\mathbb{L}$ since $\mathbb{L}(\eta,\hat{\eta}) = \int_0^1 \mathbb{L}_c(\eta,\hat{\eta})\, w(c)\, dc$.

Furthermore, the weighted area between the risk curves for an estimate $\hat{\eta}$ and the true posterior η is the regret $\mathbb{L}(\eta,\hat{\eta}) - \underline{\mathbb{L}}(\eta)$ and the statistical information $\Delta\underline{\mathbb{L}}(\eta,M) = \underline{\mathbb{L}}(\pi,M) - \underline{\mathbb{L}}(\eta,M)$ is the weighted area between the "tent" risk curve for π and the risk curve for η.

The correspondence between ROC and risks curves is due to the relationship between the true class probability η and the likelihood ratio dP/dQ for a fixed π. As shown in Section 4.1, this

20. Unlike the cost curves originally described by Drummond and Holte (2006), the version presented here does not normalise the risk, and plots the cost on the horizontal axis rather than the product of the prior probability and cost.

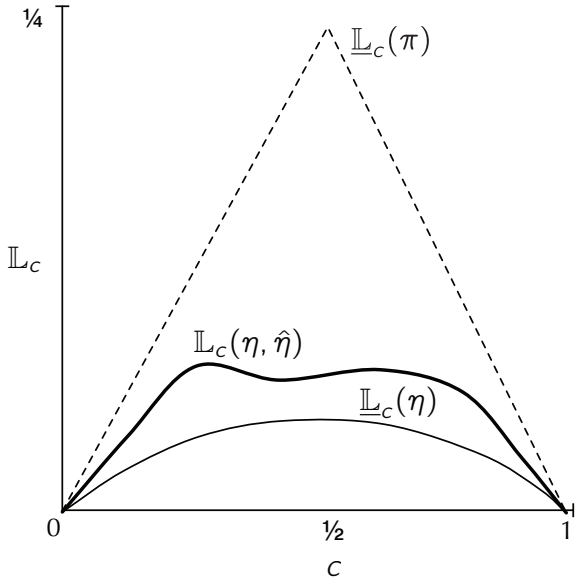


Figure 4: Example of a risk curve for costs diagram showing risk curves for costs for the true posterior probability η (bottom, solid curve), an estimate $\hat{\eta}$ (middle, bold curve) and the majority class or prior estimate (top, dashed curve).

relationship is

$$\frac{dP}{dQ} = \lambda_\pi(\eta) = \frac{1-\pi}{\pi}\frac{\eta}{1-\eta}.$$

Each cost $c \in [0,1]$ can be mapped to a corresponding test statistic threshold $\tau_0 = \lambda_\pi(c)$ and *vice versa*.

Drummond and Holte (2006) show that their cost curves have a point-line dual relationship with ROC curves. As can be established with some straight-forward algebra, the same result holds for our risk diagrams.

Proposition 20 *For a given point* $(\mathrm{FP}, \mathrm{TP})$ *on an ROC diagram the corresponding line in a risk diagram is*

$$\mathbb{L}_c = (1-\pi)\,c\,\mathrm{FP} + \pi(1-c)\,(1-\mathrm{TP}), \quad c \in [0,1]$$

Conversely, the line in ROC space corresponding to a point $(c, \mathbb{L}_c)$ *in risk space is*

$$\mathrm{TP} = \frac{(1-\pi)c}{\pi(1-c)}\mathrm{FP} + \frac{(1-\pi)c - \mathbb{L}_c}{\pi(1-c)}, \quad \mathrm{FP} \in [0,1].$$

An example of this relationship is shown graphically[21] in Figure 5 between the point A and the line A*.

21. An applet that demonstrates the relationship can be found at `http://mark.reid.name/iem/visualising-roc-and-cost-curve-duality.html`.

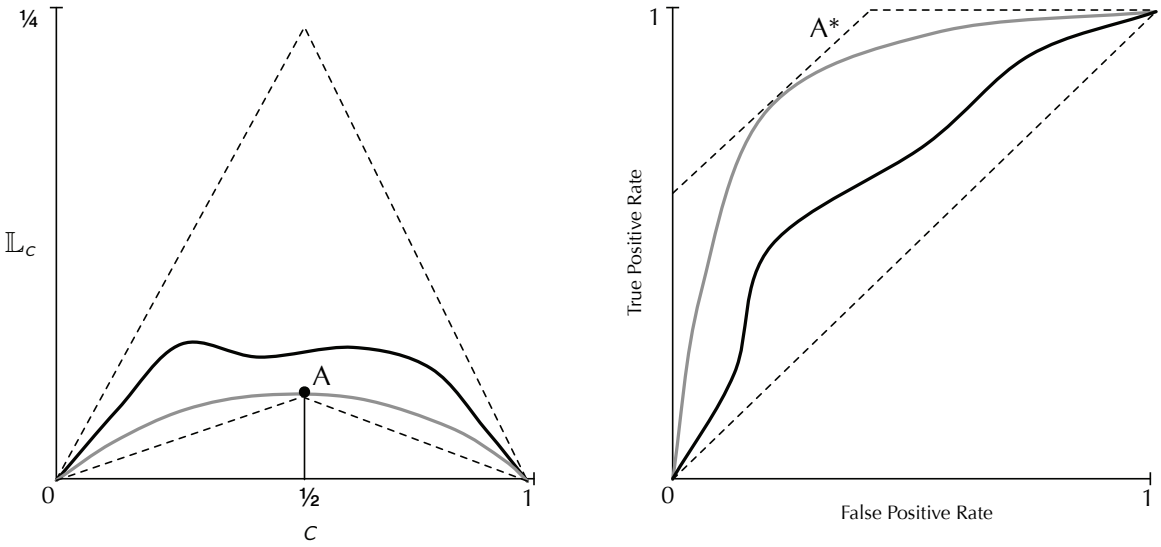


Figure 5: Cost curve diagram (left) and corresponding ROC diagram (right). The black curves on the left and right represent risk and classification rates of an example predictor. The grey Bayes risk curve on the left corresponds to the dominating grey ROC curve on the right for the likelihood statistic. Similarly, the dashed tent on the left corresponds to the dashed diagonal ROC line on the right. The point labelled A in the risk diagram corresponds to the line labelled A* in the ROC diagram.

6.3 Transforming from ROC to Risk Curves and Back

As mentioned earlier, the Neyman-Pearson lemma guarantees the ROC curve for η is maximal. This corresponds to the cost curve being minimal. In fact, these relationships are dual in the sense that there exists a transformation from one to the other as we shall now show. We make use of a connection between the Neyman-Pearson function in (11) and the maximal ROC curve due to Torgersen (1981). For completeness, a proof using our nomenclature can be found in Appendix A.7.

Theorem 21 *Let $\beta(\alpha, P, Q)$ be the Neyman-Pearson function for the binary experiment (P, Q) and let $\underline{\mathbb{L}}(\pi, P, Q)$ be the 0-1 Bayes risk on the same experiment for the prior π. Then, for any choice of $\pi \in [0,1]$ we have*

$$\underline{\mathbb{L}}(\pi, P, Q) = \underline{\mathbb{L}} = \min_{\alpha \in [0,1]} ((1-\pi)\alpha + \pi(1-\beta(\alpha, P, Q)) \tag{42}$$

and conversely for any $\alpha \in [0,1]$,

$$\beta(\alpha, P, Q) = \inf_{\pi \in (0,1]} \frac{1}{\pi}((1-\pi)\alpha + \pi - \underline{\mathbb{L}}(\pi, P, Q)). \tag{43}$$

$\pi \mapsto \underline{L}(\pi, P, Q)$ is the lower envelope of a parameterized (by π) family of affine functions (in α) and is thus concave. When $\beta(\cdot)$ and $\underline{\mathbb{L}}(\cdot)$ are smooth, explicit closed form formulas can be found:

Theorem 22 *Suppose* β *and* $\underline{\mathbb{L}}$ *are differentiable on* $(0,1]$ *and* $[0,1]$ *respectively. Then*

$$\underline{\mathbb{L}}(\pi) = (1-\pi)\check{\beta}(\pi) + \pi(1-\beta(\check{\beta}(\pi))), \quad \pi \in [0,1], \tag{44}$$

where

$$\check{\beta}(\pi) := \beta'^{-1}\left(\frac{1-\pi}{\pi}\right)$$

and

$$\beta(\alpha) = \frac{1}{\check{\underline{\mathbb{L}}}(\alpha)}\left[(1-\check{\underline{\mathbb{L}}}(\alpha))\alpha + \check{\underline{\mathbb{L}}}(\alpha) - \underline{\mathbb{L}}(\check{\underline{\mathbb{L}}}(\alpha))\right], \quad \alpha \in (0,1], \tag{45}$$

where

$$\begin{aligned} \check{\underline{\mathbb{L}}}(\alpha) &:= \tilde{\underline{\mathbb{L}}}^{-1}(\alpha) \wedge 1, \\ \tilde{\underline{\mathbb{L}}}(\pi) &:= \underline{\mathbb{L}}(\pi) - \pi\underline{\mathbb{L}}'(\pi). \end{aligned}$$

The proof can be found in Appendix A.6.

Using (45) we present an example. Consider $\underline{\mathbb{L}}(\pi) = \gamma\pi(1-\pi)$ for $\gamma \in [0,1]$ One can readily check that $\tilde{\underline{\mathbb{L}}}_{(\gamma)}(\pi) = \gamma\pi^2$. Hence $\tilde{\underline{\mathbb{L}}}^{-1}_{(\gamma)}(\alpha) = \sqrt{\frac{\alpha}{\gamma}} \in \left[0, \frac{1}{\gamma}\right]$. Thus $\check{\underline{\mathbb{L}}}_{(\gamma)}(\alpha) = 0 \vee \tilde{\underline{\mathbb{L}}}^{-1}_{(\gamma)}(\alpha) \wedge 1 = \sqrt{\alpha/\gamma} \wedge 1$. Substituting and rearranging we find that the corresponding β is given by

$$\beta_\gamma(\alpha) = \frac{\alpha + \gamma + (\sqrt{\alpha/\gamma} \wedge 1)(1-\alpha-\gamma)}{\sqrt{\alpha/\gamma} \wedge 1}.$$

A graph of this $\beta(\cdot)$ is given in figure 6.

By construction $\beta(1) = 1$ and β is concave and continuous on $(0,1]$. The following lemma is due to Torgersen (1991). Given mild conditions on the space of instances, this gives a corollary which guarantees that all concave curves on a risk diagram can be realised by some pair of distributions. Their proofs can be found in Appendix A.8 and Appendix A.9, respectively.

Lemma 23 *Suppose* $\mathcal{X}$ *contains a connected component* $\mathcal{C}$. *Let* $\phi\colon [0,1] \to [0,1]$ *be an arbitrary function that is concave and continuous on* $(0,1]$ *such that* $\phi(1) = 1$. *Then there exists distributions* P *and* Q *on* $\mathcal{X}$ *such that* $\beta(\alpha, P, Q) = \phi(\alpha)$ *for all* $\alpha \in [0,1]$.

Corollary 24 *Suppose* $\mathcal{X}$ *contains a connected component. Let* $\psi\colon [0,1] \to [0,1]$ *be an arbitrary concave function such that for all* $\pi \in [0,1]$, $0 \le \psi(\pi) \le \pi \wedge (1-\pi)$. *Then there exists distributions* P *and* Q *on* $\mathcal{X}$ *such that* $\underline{\mathbb{L}}(\pi, P, Q) = \psi(\pi)$ *for all* $\pi \in [0,1]$.

The corollary shows that reasoning about cost-weighted risks for all possible binary experiments (P,Q) can be done purely geometrically. Each experiment can be associated with a concave curve and *vice versa* so that the existence of an experiment becomes equivalent to the existence of a concave curve with certain properties. This relationship is exploited in the next section to establish bounds for f-divergences in Theorem 30.

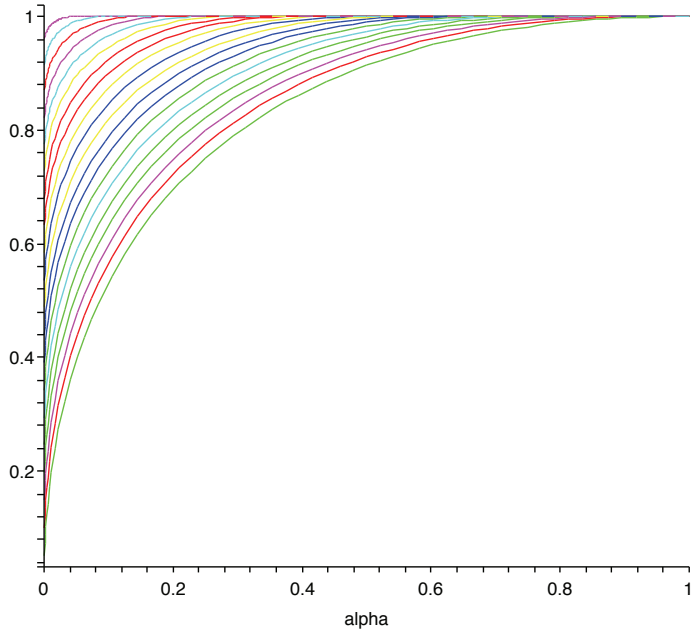


Figure 6: Graph of the parameterised Neyman-Pearson function $\alpha \mapsto \beta_\gamma(\alpha, P, Q)$ for $\gamma = i/20$, $i = 1, \ldots, 20$. (See text.)

7. Bounding General Objects in Terms of Primitives

All of the above results are exact—they are exact representations of particular primitives or general objects in terms of other primitives. Another type of relationship is an inequality. In this section we consider how we can (tightly) bound the value of a general object ($\mathbb{I}_f$ or B_w) in terms of primitive objects (V_π—the generalised variational divergence defined below—or B_c, the regret with respect to the cost weight loss (29)). Bounding $\mathbb{I}_f(P,Q)$ in terms of $V_\pi(P,Q)$ is a generalisation of the classical Pinsker inequality (Pinsker, 1964). Bounding $B_w(\eta,\hat{\eta})$ in terms of $B_c(\eta,\hat{\eta})$ is a generalisation of the so-called "surrogate regret bounds" (Zhang, 2004b; Bartlett et al., 2006).

As explained previously, we work with the *conditional* Bregman divergence $B_w(\eta,\hat{\eta})$. Results in terms of $B_w(\eta,\hat{\eta})$, $\eta,\hat{\eta} \in [0,1]$ immediately imply results for $\mathbb{B}_w(\eta,\hat{\eta})$, where $\eta,\hat{\eta} \in [0,1]^{\mathcal{X}}$ by taking expectations with respect to X.

7.1 Surrogate Regret Bounds

Suppose for some fixed $c_0 \in (0,1)$ that $B_{c_0}(\eta,\hat{\eta}) = \alpha$. What can be said concerning the value of $B_w(\eta,\hat{\eta})$ for an arbitrary weight function w? Surrogate regret bounds answer this question by showing how the value of B_{c_0} is controlled by a function of B_w. That is, $B_{c_0} \leq F(B_w)$ for some non-decreasing F. The main result of this subsection, Theorem 25, presents a general surrogate

bound for proper losses implicitly as $B_w \geq F^{-1}(B_{c_0})$. However, as Corollary 28 shows, this implicit bound can always be inverted.

Previous work on this problem is summarised in Appendix D. Apart from their theoretical interest, these bounds have direct practical implications: it can often be much simpler to minimise $B_w(\eta,\hat{\eta})$ over $\hat{\eta}$ than to minimise $B_c(\eta,\hat{\eta})$. The bounds below will tell the user of such a scheme the maximum price they will have to pay, in terms of statistical performance, for using a particular surrogate.

Theorem 25 *Let $c_0 \in (0,1)$ and let $B_{c_0}(\eta,\hat{\eta})$ denote the point-wise regret for the cost-weighted loss ℓ_{c_0}. Suppose it is known that $B_{c_0}(\eta,\hat{\eta}) = \alpha$. Then the point-wise regret $B(\eta,\hat{\eta})$ for any proper surrogate loss ℓ with point-wise risk L and Bayes risk $\underline{L}$ satisfies*

$$B(\eta,\hat{\eta}) \geq \psi(c_0,\alpha) \vee \psi(c_0,-\alpha), \tag{46}$$

where

$$\psi(c_0,\alpha) := B(c_0,c_0+\alpha) = \underline{L}(c_0) - \underline{L}(c_0+\alpha) + \alpha\underline{L}'(c_0).$$

Furthermore (46) is tight.

The proof of this bound is almost a direct consequence of the fact that regrets for proper losses are Bregman divergences (see Section 4.4). This is a simplified version of an earlier proof by Reid and Williamson (2009). We will make use of the following expression for B_c derived by Buja et al. (2005). Its proof can be found in Appendix A.10.

Lemma 26 *Suppose L_c is the conditional risk for cost-sensitive misclassification loss (see 5.2). For any loss $c \in [0,1]$ the cost-weighted regret $B_c(\eta,\hat{\eta}) := L_c(\eta,\hat{\eta}) - \underline{L}_c(\eta)$ satisfies*

$$B_c(\eta,\hat{\eta}) = |\eta - c| [\![\eta \wedge \hat{\eta} < c \leq \eta \vee \hat{\eta}]\!].$$

Proof (Theorem 25) Let B be the conditional regret associated with some arbitrary proper loss ℓ and suppose that we know the cost-weighted regret $B_{c_0}(\eta,\hat{\eta}) = \alpha$. By Lemma 26, this implies that $\alpha = \eta - c_0$ when $\hat{\eta} \leq c_0 < \eta$ and $\alpha = c_0 + \eta$ when $\eta \leq c_0 < \hat{\eta}$. Since $B(\eta,\hat{\eta})$ is a Bregman divergence its value decreases as $|\eta - \hat{\eta}|$ decreases (see Section 2.5). Thus, in the first case we have $\hat{\eta} \leq c_0 < c_0 + \alpha = \eta$ and so $B(\eta,\hat{\eta}) = B(c_0+\alpha,\hat{\eta}) \geq B(c_0+\alpha,c_0)$ and is minimised when $\hat{\eta} = c_0$.

The proof of the second case, when $\eta = c_0 - \alpha \leq c_0 < \hat{\eta}$ proceeds identically. Thus, $B(\eta,\hat{\eta})$ is no smaller than each of $B(c_0+\alpha,c_0)$ and $B(c_0-\alpha,c_0)$, giving the required result. ■

By restricting attention to the case when $c_0 = \frac{1}{2}$ and symmetric losses we obtain, as a corollary, a result similar to that presented by Bartlett et al. (2006) for surrogate margin losses since $B_{\frac{1}{2}}$ is easily shown to be half the 0-1 regret. It is obtained by substituting $\alpha = \frac{1}{2}$ and noting the symmetry of L implies $\underline{L}'(\frac{1}{2}) = 0$; Appendix D contains some examples illustrating this special case.

Corollary 27 *If $\underline{L}$ is symmetric—that is, $\underline{L}(\frac{1}{2}-c) = \underline{L}(\frac{1}{2}+c)$ for $c \in [0,\frac{1}{2}]$—and $B_{\frac{1}{2}}(\eta,\hat{\eta}) = \alpha$, then*

$$B(\eta,\hat{\eta}) \geq \underline{L}(\tfrac{1}{2}) - \underline{L}(\tfrac{1}{2}+\alpha).$$

The bounds in Theorem 25 can be inverted to allow the approximate minimisation of a cost-weighted loss via the minimisation of a surrogate loss.

Corollary 28 *Minimising $B(\eta,\hat{\eta})$ w.r.t. $\hat{\eta}$ minimises the bound on $B_c(\eta,\hat{\eta})$ for each $c \in (0,1)$.*

Proof To see this, let $\psi'(c_0,\alpha) := \frac{\partial}{\partial\alpha}\psi(c_0,\alpha) = -\underline{L}'(c_0+\alpha)+\underline{L}'(c_0)$. Since $\underline{L}$ is concave, $\underline{L}'$ is non-increasing and hence $\underline{L}'(c_0+\alpha) \le \underline{L}'(c_0)$ and so $\psi'(c_0,\alpha) \ge 0$ and therefore $\alpha \mapsto \psi(c_0,\alpha)$ is non-decreasing and thus invertible (although there may be non-uniqueness at points where $\psi(c_0,\alpha)$ is constant in α). This invertibility means minimising $B(\eta,\hat{\eta})$ w.r.t. $\hat{\eta}$, minimises the bound on $B_c(\eta,\hat{\eta})$. ■

Finally, Theorem 25 can be used to immediately establish a loose, second-order bound in α for symmetric losses in terms of their weight function, similar to a result due to Buja et al. (2005).

Corollary 29 *Suppose B_w is the regret for a symmetric proper loss ℓ with associated weight function w. Then*

$$B_w(\eta,\hat{\eta}) \ge \frac{w(\frac{1}{2})}{2}\left[B_{\frac{1}{2}}(\eta,\hat{\eta})\right]^2.$$

Proof A Taylor series expansion of the second term in the bound of Corollary 27 about $\alpha = \frac{1}{2}$ gives

$$B_w(\eta,\hat{\eta}) \ge \frac{w(\frac{1}{2})}{2}\alpha^2 + \frac{w''(\frac{1}{2})}{24}\alpha^4 + \cdots$$

since the linear term cancels and there is no third order term since w is symmetric and thus $w'(\frac{1}{2}) = 0$. Setting $\alpha = B_{\frac{1}{2}}(\eta,\hat{\eta})$ gives the result. ■

Some extensions to the above result have been recently presented by Scott (2010).

7.2 General Pinsker Inequalities for Divergences

The many different f divergences are single number summaries of the relationship between two distributions P and Q. Each f-divergence emphasises different aspects. Merely considering the functions f by which f-divergences are traditionally defined makes it hard to understand these different aspects, and harder still to understand how knowledge of $\mathbb{I}_{f_1}$ constrains the possible values of $\mathbb{I}_{f_2}$. When $\mathbb{I}_{f_1} = V$ (a special primitive for $\mathbb{I}_f$) and $\mathbb{I}_{f_2} = \mathrm{KL}$, this is a classical problem that has been studied for decades; Appendix E summarises the history.

Vajda (1970) posed the question of a *tight lower bound* on KL-divergence in terms of variational divergence. This "best possible Pinsker inequality" takes the form

$$L(V) := \inf_{V(P,Q)=V} \mathrm{KL}(P,Q), \quad V \in [0,2), \tag{47}$$

where the infimum is over all P and Q such that $V(P,Q) = V$. Recently Fedotov et al. (2003) presented an *implicit* (parametric) version of the form

$$\begin{aligned} &(V(t),L(t))_{t\in\mathbb{R}^+}, \\ &V(t) = t\left(1-\left(\coth(t)-\frac{1}{t}\right)^2\right), \quad L(t) = \ln\left(\frac{t}{\sinh(t)}\right) + t\coth(t) - \frac{t^2}{\sinh^2(t)}. \end{aligned} \tag{48}$$

We will now show how viewing f-divergences in terms of their weighted integral representation simplifies the problem of understanding the relationship between different divergences and leads, amongst other things, to an explicit formula for (47).

We make use of a generalised notion of variational divergence:

$$V_\pi(P,Q) := 2 \sup_{r \in [-1,1]^{\mathcal{X}}} |\pi \mathbb{E}_P r - (1-\pi)\mathbb{E}_Q r|, \tag{49}$$

where $\pi \in (0,1)$ and the supremum is over all measurable functions from $\mathcal{X}$ to $[-1,1]$.

Fix a positive integer n. Consider a sequence $0 < \pi_1 < \pi_2 < \cdots < \pi_n < 1$. Suppose we "sampled" the value of $V_\pi(P,Q)$ at these discrete values of π. Since $\pi \mapsto V_\pi(P,Q)$ is concave, the piecewise linear concave function passing through points

$$\{(\pi_i, V_{\pi_i}(P,Q))\}_{i=1}^n$$

is guaranteed to be an upper bound on the variational curve $(\pi, V_\pi(P,Q))_{\pi \in (0,1)}$. This therefore gives a lower bound on the f-divergence given by a weight function γ. This observation forms the basis of the theorem stated below.

Theorem 30 *For a positive integer n consider a sequence $0 < \pi_1 < \pi_2 < \cdots < \pi_n < 1$. Let $\pi_0 := 0$ and $\pi_{n+1} := 1$ and for $i = 0, \ldots, n+1$ let*

$$\psi_i := (1-\pi_i) \wedge \pi_i - V_{\pi_i}(P,Q)$$

(observe that consequently $\psi_0 = \psi_{n+1} = 0$). Let

$$\begin{aligned} A_n \quad := \quad & \Bigg\{ \mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{R}^n : \\ & \frac{\psi_{i+1} - \psi_i}{\pi_{i+1} - \pi_i} \leq a_i \leq \frac{\psi_i - \psi_{i-1}}{\pi_i - \pi_{i-1}},\ i = 1, \ldots, n \Bigg\}. \end{aligned} \tag{50}$$

The set A_n defines the allowable slopes of a piecewise linear function majorizing $\pi \mapsto V_\pi(P,Q)$ and matching it at each of $\pi_1, \ldots, \pi_n$. For $\mathbf{a} = (a_1, \ldots, a_n) \in A_n$, let

$$\tilde{\pi}_i := \frac{\psi_i - \psi_{i+1} + a_{i+1}\pi_{i+1} - a_i \pi_i}{a_{i+1} - a_i},\ i = 0, \ldots, n, \tag{51}$$

$$j := \{k \in \{1, \ldots, n\} : \tilde{\pi}_k < \tfrac{1}{2} \leq \tilde{\pi}_{k+1}\}, \tag{52}$$

$$\pi_i := [\![i < j]\!] \tilde{\pi}_i + [\![i = j]\!] \tfrac{1}{2} + [\![j < i]\!] \tilde{\pi}_{i-1}, \tag{53}$$

$$\alpha_{\mathbf{a},i} := [\![i \leq j]\!](1 - a_i) + [\![i > j]\!](-1 - a_{i-1}), \tag{54}$$

$$\beta_{\mathbf{a},i} := [\![i \leq j]\!](\psi_i - a_i \pi_i) + [\![i > j]\!](\psi_{i-1} - a_{i-1}\pi_{i-1}) \tag{55}$$

for $i = 0, \ldots, n+1$ and let γ_f be the weight corresponding to f given by (28).

For arbitrary $\mathbb{I}_f$ and for all distributions P and Q on $\mathcal{X}$ the following bound holds. If in addition $\mathcal{X}$ contains a connected component, it is tight.

$$\mathbb{I}_f(P,Q) \geq \min_{\mathbf{a} \in A_n} \sum_{i=0}^n \int_{\pi_i}^{\pi_{i+1}} (\alpha_{\mathbf{a},i}\pi + \beta_{\mathbf{a},i}) \gamma_f(\pi) d\pi \tag{56}$$

$$\begin{aligned} = \min_{\mathbf{a} \in A_n} \sum_{i=0}^n \big[& (\alpha_{\mathbf{a},i}\pi_{i+1} + \beta_{\mathbf{a},i}) \Gamma_f(\pi_{i+1}) - \alpha_{\mathbf{a},i} \Gamma_f(\pi_{i+1}) \\ & - (\alpha_{\mathbf{a},i}\pi_i + \beta_{\mathbf{a},i}) \Gamma_f(\pi_i) + \alpha_{\mathbf{a},i} \Gamma_f(\pi_i) \big], \end{aligned} \tag{57}$$

where $\Gamma_f(\pi) := \int^{\pi} \gamma_f(t)dt$ *and* $\Gamma_f(\pi) := \int^{\pi} \Gamma_f(t)dt$.

Equation 57 follows from (56) by integration by parts. The remainder of the proof is in Section A.12. Although (57) looks daunting, we observe: (1) the constraints on $\mathbf{a}$ are convex (in fact they are a box constraint); and (2) the objective is a relatively benign function of $\mathbf{a}$.

When $n = 1$ the result simplifies considerably. If in addition $\pi_1 = \frac{1}{2}$ then $V_{\frac{1}{2}}(P,Q) = \frac{1}{4}V(P,Q)$. It is then a straightforward exercise to explicitly evaluate (56), especially when γ_f is symmetric. The following theorem expresses the result in terms of $V(P,Q)$ for comparability with previous results. The result for $\mathrm{KL}(P,Q)$ is a (best-possible) improvement on the classical Pinsker inequality.

Theorem 31 *For any distributions P,Q on $\mathcal{X}$, let $V := V(P,Q)$. Then the following bounds hold and, if in addition $\mathcal{X}$ has a connected component, are tight.*

When γ is symmetric about $\frac{1}{2}$ and convex,

$$\mathbb{I}_f(P,Q) \geq 2\left[\Gamma_f\left(\tfrac{1}{2} - \tfrac{V}{4}\right) + \tfrac{V}{4}\Gamma_f\left(\tfrac{1}{2}\right) - \Gamma_f\left(\tfrac{1}{2}\right)\right]$$

and Γ_f and Γ_f are as in Theorem 30.

This theorem gives the first explicit representation of the optimal Pinsker bound.[22]

Corollary 32 *The following special cases hold (γ symmetric about $1/2$).*

$$\begin{aligned}
h^2(P,Q) &\geq 2 - \sqrt{4 - V^2}, \\
\mathrm{J}(P,Q) &\geq 2V \ln\left(\tfrac{2+V}{2-V}\right), \\
\Psi(P,Q) &\geq \frac{8V^2}{4 - V^2}, \\
\mathrm{I}(P,Q) &\geq \left(\tfrac{1}{2} - \tfrac{V}{4}\right)\ln(2-V) + \left(\tfrac{1}{2} + \tfrac{V}{4}\right)\ln(2+V) - \ln(2), \\
\mathrm{T}(P,Q) &\geq \ln\left(\tfrac{4}{\sqrt{4-V^2}}\right) - \ln(2).
\end{aligned}$$

The following special cases hold (γ is not symmetric)

$$\chi^2(P,Q) \geq [\![V < 1]\!]V^2 + [\![V \geq 1]\!]\tfrac{V}{(2-V)}, \tag{58}$$

$$\mathrm{KL}(P,Q) \geq \min_{\beta \in [V-2, 2-V]} \left(\tfrac{V+2-\beta}{4}\right)\ln\left(\tfrac{\beta-2-V}{\beta-2+V}\right) + \left(\tfrac{\beta+2-V}{4}\right)\ln\left(\tfrac{\beta+2-V}{\beta+2+V}\right). \tag{59}$$

By plotting both (48) and (59) one can confirm that the two bounds (implicit and explicit) coincide; see Figure 7.

The above theorem suggests a means by which one can *estimate* an f-divergence by estimating a sequence $(\mathbb{L}_{c_i}(\pi, P, Q))_{i=1}^n$. A simpler version of such an idea (more directly using the representation (27)) has been studied by Song et al. (2008).

22. A summary of existing results and their relationship to those presented here is given in Appendix E.

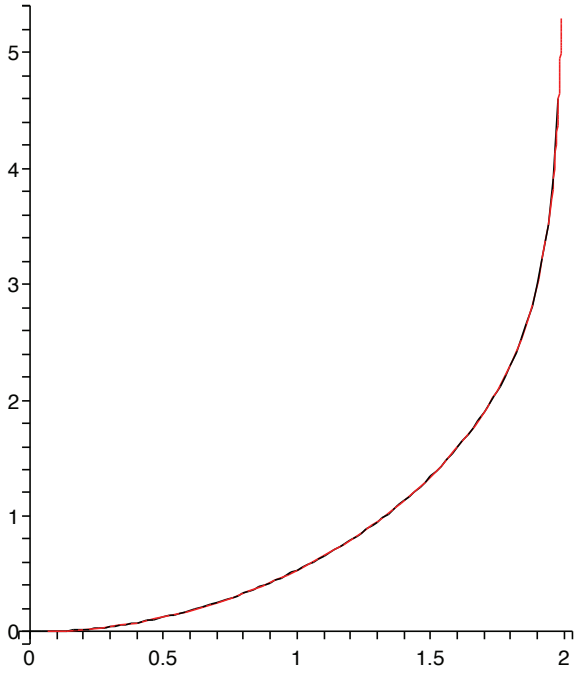


Figure 7: Lower bound on $\mathrm{KL}(P,Q)$ as a function of the variational divergence $V(P,Q)$. Both the explicit bound (59) and Fedotorev et al.'s implicit bound (48) are plotted.

8. Variational Representations

We have already seen a number of connections between the Bayes risk

$$\underline{\mathbb{L}}(\pi,P,Q) = \inf_{\hat{\eta}\in[0,1]^{\mathcal{X}}} \mathbb{E}_{\mathsf{X}\sim M}\left[\ell(\eta(\mathsf{X}),\hat{\eta}(\mathsf{X}))\right]$$

and the f-divergence

$$\mathbb{I}_f(P,Q) = \mathbb{E}_Q\left[f\left(\frac{dP}{dQ}\right)\right]. \tag{60}$$

Comparing these definitions leads to an obvious and intriguing point: the definition of $\underline{\mathbb{L}}$ involves an optimisation, whereas that for $\mathbb{I}_f$ does not. Observe that the normal usage of these quantities is that one wishes to know not just the real number $\underline{\mathbb{L}}(\pi,P,Q)$, but also the estimate $\hat{\eta}\colon \mathcal{X}\to[0,1]$ that attains the minimal risk. In this section we will explore two views of $\mathbb{I}_f$—relating the standard definition to a *variational* one that explains where the optimisation is hidden in (60). We then explore some simpler relationships when using the linear "loss". In Appendix F we consider the variational representation of $\mathbb{I}_f$ obtained by representing f in terms of the LF dual $f^\star$. We also explore some generalisations that naturally arise from this representation and relate them to each other and to the standard f-divergence.

The easiest place to start, unsurprisingly, is with the variational divergence. Below we derive a straight-forward extension of the classical result relating $\mathbb{L}^{0-1}(\frac{1}{2}, P, Q)$ to $V(P,Q)$. We then explore variational representations for general f-divergences.

8.1 Generalised Variational Divergence

Let $\mathcal{C} \subseteq \{-1,1\}^{\mathcal{X}}$ denote a collection of measurable binary *classifiers* on $\mathcal{X}$. Consider the (constrained[23]) Bayes risk for 0-1 loss minimised over this set:

$$\mathbb{L}^{0-1}_{\mathcal{C}}(\pi, P, Q) = \inf_{r \in \mathcal{C}} \mathbb{E}_{(\mathsf{X},\mathsf{Y})\sim\mathbb{P}}[\ell^{0-1}(r(\mathsf{X}), \mathsf{Y})]. \tag{61}$$

The variational divergence is so called because it can be written

$$V(P,Q) = 2 \sup_{A \subseteq \mathcal{X}} |P(A) - Q(A)|, \tag{62}$$

where the supremum is over all measurable subsets of $\mathcal{X}$. Since

$$V(P,Q) = \sup_{r \in [-1,1]^{\mathcal{X}}} |\mathbb{E}_P r - \mathbb{E}_Q r|,$$

consider the following generalisation of V:

$$V_{\mathcal{R},\pi}(P,Q) := 2 \sup_{r \in \mathcal{R} \subseteq [-1,1]^{\mathcal{X}}} |\pi \mathbb{E}_P r - (1-\pi)\mathbb{E}_Q r|, \tag{63}$$

where $\pi \in (0,1)$ and the supremum is over all measurable functions from $\mathcal{X}$ to $[-1,1]$. (If $\mathcal{R} = [-1,1]^{\mathcal{X}}$ we just write $V_\pi(P,Q)$.) When $\pi = \frac{1}{2}$ this is a scaled version of what Müller (1997a,b) calls an *integral probability metric*.[24]

If $\mathcal{R}$ is *symmetric about zero* ($r \in \mathcal{R} \Rightarrow -r \in \mathcal{R}$), then the absolute value signs in (63) can be removed. To see this, suppose the supremum was attained at $\bar{r}$ and that $\alpha := \pi\mathbb{E}_P\bar{r} - (1-\pi)\mathbb{E}_Q\bar{r} < 0$. Choose $\bar{r}' := -\bar{r}$ and observe that $\pi\mathbb{E}_P\bar{r}' - (1-\pi)\mathbb{E}_Q\bar{r}' = -\alpha > 0$. Thus $V_{\mathcal{R},\pi}(P,Q) = 2\sup_{r\in\mathcal{R}\subseteq[-1,1]^{\mathcal{X}}}(\pi\mathbb{E}_P r - (1-\pi)\mathbb{E}_Q r)$.

Let $\operatorname{sgn}\mathcal{R} := \{\operatorname{sgn} r\colon r \in \mathcal{R}\}$ and for $a, b \in \mathbb{R}$, let $a\mathcal{R} + b := \{ar + b\colon r \in \mathcal{R}\}$.

Theorem 33 *Suppose $\mathcal{R} \subseteq [-1,1]^{\mathcal{X}}$ is symmetric about zero and $\operatorname{sgn}\mathcal{R} \subseteq \mathcal{R}$. For all $\pi \in (0,1)$ and all P and Q*

$$\mathbb{L}^{0-1}_{(\operatorname{sgn}\mathcal{R}+1)/2}(\pi, P, Q) = \tfrac{1}{2} - \tfrac{1}{4} V_{\mathcal{R},\pi}(P,Q) \tag{64}$$

and the infimum in (61) corresponds to the supremum in (63).

The proof is in Appendix A.11.

23. Tong and Koller (2000) call this the *restricted* Bayes risk.

24. Zolotarev (1984) calls this a *probability metric with ζ-structure*. There are probability metrics that are neither f-divergences nor integral probability metrics. A large collection is due to Rachev (1991). A recent survey on relationships (inequalities and some representations) has been given by Gibbs and Su (2002). The idea of generalising variational divergence by restricting the set the supremum is taken over is also used by Ben-David et al. (2010).

8.2 The Linear "Loss" and the Generalised Variational Divergence

Theorem 33 shows that computing $V_{\mathcal{R},\pi}$ involves an optimisation problem equivalent to that arising in the determination of $\underline{\mathbb{L}}$. The arg min in the definition of $\underline{\mathbb{L}}$ is usually called the *hypothesis* (or *Bayes optimal hypothesis*). Following Borgwardt et al. (2006) we will call the arg max in (63) the *witness*.

When $\mathcal{R} = [-1,1]^{\mathcal{X}}$ and $\pi = \frac{1}{2}$, $\operatorname{sgn}\mathcal{R} \subseteq \mathcal{R}$ and furthermore $\mathcal{C} = (\operatorname{sgn}\mathcal{R}+1)/2 = \{0,1\}^{\mathcal{X}}$ and so Theorem 33 reduces to the classical result that $\underline{\mathbb{L}}^{0-1}(\frac{1}{2},P,Q) = \frac{1}{2} - \frac{1}{4}V(P,Q)$ (Devroye et al., 1996).

The requirement that $\operatorname{sgn}\mathcal{R} \subseteq \mathcal{R}$ is unattractive. It is necessitated by the use of 0-1 loss. It can be removed by instead considering the *linear loss*

$$\ell^{\text{lin}}(r(x),y) := 1 - yr(x), \quad y \in \{-1,1\}.$$

If r is unrestricted, then there is no guarantee that $\ell^{\text{lin}} > -\infty$ and is thus a legitimate loss function. Below we will always consider $r \in \mathcal{R}$ such that the linear loss is bounded from below. Observe that the common hinge loss (Steinwart and Christmann, 2008) is simply $\ell^{\text{hinge}}(f(x),y) = 0 \vee \ell^{\text{lin}}(f(x),y)$.

Theorem 34 *Assume that $\mathcal{R} \subseteq [-a,a]^{\mathcal{X}}$ for some $a > 0$ and is symmetric about zero. Then for all $\pi \in (0,1)$ and all distributions P and Q on $\mathcal{X}$*

$$\underline{\mathbb{L}}^{\text{lin}}_{\mathcal{R}}(\pi,P,Q) = 1 - \frac{1}{2}V_{\mathcal{R},\pi}(P,Q)$$

and the r that attains $\underline{\mathbb{L}}^{\text{lin}}_{\mathcal{R}}(\pi,P,Q)$ corresponds to the r that obtains the supremum in the definition of $V_{\mathcal{R},\pi}(P,Q)$.

Proof

$$\begin{aligned}
\underline{\mathbb{L}}^{\text{lin}}_{\mathcal{R}}(\pi,P,Q) &= \inf_{r\in\mathcal{R}} \left(\pi\mathbb{E}_{\mathsf{X}\sim P}\ell^{\text{lin}}(r(\mathsf{X}),-1) + (1-\pi)\mathbb{E}_{\mathsf{X}\sim Q}\ell^{\text{lin}}(r(\mathsf{X}),+1)\right)\\
&= \inf_{r\in\mathcal{R}} \left(\pi\mathbb{E}_{\mathsf{X}\sim P}(1+r(\mathsf{X})) + (1-\pi)\mathbb{E}_{\mathsf{X}\sim Q}(1-r(\mathsf{X}))\right)\\
&= \inf_{r\in\mathcal{R}} \left(\pi + \pi\mathbb{E}_P r + (1-\pi) - (1-\pi)\mathbb{E}_Q r\right)\\
&= 1 + \inf_{r\in\mathcal{R}} \left(\pi\mathbb{E}_P r - (1-\pi)\mathbb{E}_Q r\right)\\
&= 1 - \sup_{r\in\mathcal{R}} \left(\pi\mathbb{E}_P(-r) - (1-\pi)\mathbb{E}_Q(-r)\right)\\
&= 1 - \sup_{r\in\mathcal{R}} \left(\pi\mathbb{E}_P r - (1-\pi)\mathbb{E}_Q r\right)\\
&= 1 - \frac{1}{2}V_{\mathcal{R},\pi}(P,Q),
\end{aligned}$$

where the penultimate step exploits the symmetry of $\mathcal{R}$. ■

Now suppose that $\mathcal{R} = B_{\mathcal{H}} := \{r\colon \|r\|_{\mathcal{H}} \le 1\}$, the unit ball in $\mathcal{H}$, a Reproducing Kernel Hilbert Space (RKHS) (Schölkopf and Smola, 2002). Thus for all $r \in \mathcal{R}$ there exists a *feature map* $\phi\colon \mathcal{X} \to \mathcal{H}$ such that $r(x) = \langle r, \phi(x)\rangle_{\mathcal{H}}$ and $\langle \phi(x), \phi(y)\rangle_{\mathcal{H}} = k(x,y)$, where k is a positive definite *kernel* function. Borgwardt et al. (2006) show that

$$V^2_{B_{\mathcal{H}},\frac{1}{2}}(P,Q) = \frac{1}{4}\|\mathbb{E}_P\phi - \mathbb{E}_Q\phi\|^2_{\mathcal{H}}. \tag{65}$$

Given	Assumed	Derived
(P,Q)	$f \leftrightarrow \gamma$	$\mathbb{I}_f(P,Q)$
(π,P,Q)	$U \leftrightarrow w,W,\overline{W}$	$\mathbb{J}(U(\eta)) = \Delta\underline{\mathbb{L}}(\pi,P,Q)$ $\underline{\mathbb{L}}(\eta)$
	$\hat{\eta}$	$L_w(\eta,\hat{\eta}), B_w(\eta,\hat{\eta})$

Table 3: Summary relationships between key objects arising in Binary Experiments. "Given" indicates the object is given or provided by the world; "Assumed" is something the user of assumes or imposes in order to create a well defined problem; "Derived" indicates quantities that are derived from the primitives.

Thus

$$\underline{\mathbb{L}}^{\text{lin}}_{\mathcal{R}}(\pi,P,Q) = 1 - \frac{1}{4}\|\mathbb{E}_P\phi - \mathbb{E}_Q\phi\|_{\mathcal{H}}. \tag{66}$$

Empirical estimators derived from the correspondence between (65) and (66) lead to the ν-Support Vector Machine and Maximum Mean Discrepancy; see Appendix H. Further generalizations of variational representations of $\mathbb{I}_f$ are explored in Appendix F.

9. Conclusions

There are several existing concepts that can be used to quantify the amount of information in a task and its difficulty: Uncertainty, Bregman information, statistical information, Bayes risk and regret, and f-divergences. Information is a difference in uncertainty; regret is a difference in risk. In the case of supervised binary class probability estimation, we have connected and extended several existing results in the literature to show how to translate between these perspectives. The representations allow a precise answer to the question of what are the primitives for binary experiments.

We have derived the integral representations in a simple and unified manner, and illustrated the value of the representations. Along the way we have drawn connections to a diverse set of concepts related to binary experiments: risk curves, cost curves, ROC curves and the area under them; variational representations of f-divergences, risks and regrets.

Two key consequences are surrogate regret bounds that are at once more general and simpler than those in the literature, and a generalisation of the classical Pinkser inequality providing, *inter alia*, an explicit form for the best possible Pinsker inequality relating Kullback-Leibler divergence and Variational divergence. We have also presented a new derivation of support vector machines and their relationship to Maximum Mean Discrepancy (integral probability metrics).

The key relationships between the basic objects of study are summarised in Table 3 and Figure 1 in §1.2.

All of the results we have presented demonstrate the fundamental and elementary nature of the cost-weighted misclassification loss, which is becoming increasingly appreciated in the Machine Learning literature (Bach et al., 2006; Beygelzimer et al., 2008). The viewpoint developed in this paper has also recently been used to better understand the structure of composite binary losses (losses involving a link function)—see Reid and Williamson (2010).

More generally, the present work is small part of a larger structural research agenda to understand the whole field of machine learning in terms of *relations* between problems. We envisage these relations being richer and more powerful than the already valuable *reductions* between learning problems. Much of the present literature on machine learning is highly solution focussed. Of course one does indeed like to *solve* problems, and we do not suggest otherwise. But it is hard to see structure in the panoply of solutions which continue to grow each year. The present paper is a first step to a pluralistic unification of a diverse set of machine learning problems. The goal we have in mind can be explained by analogy. There are several such analogies:

Computational Complexity Within the field of NP-completeness (Garey and Johnson, 1979; Johnson, 1982–1992; 2005–2007) lead to a detailed and structured understanding of the *relationships* between many fundamental problems and consequently guides the search for solutions for new problems.

Functional Analysis Compare Machine Learning problems with mathematical *functions*. In the 19th century, each function was considered separately. Functional Analysis (Lindström, 2008) *catalogued* them by considering *sets* of functions and *relations* (mappings) between them and subsequently developed many new and powerful tools. The increasing abstraction and focus on relations has remained a powerful force in mathematics (Wikipedia, 2007).

Biology A systematic *cataloging* (taxonomy) resonates with Biology's Linnean past—and taxonomies can indeed lead to standardisation and efficiency (Bowker and Star, 1999). But taxonomies alone are inadequate—it seems necessary to understand the relationships in a manner analogous to *Systems Biology* which "is about putting together rather than taking apart, integration rather than reduction.... Successful integration at the systems level must be built on successful reduction, but reduction alone is far from sufficient" (Noble, 2006).

Geology Finally, Lyell's *Principles of Geology* (Lyell, 1830) was a watershed in Geology's history (Bowker, 2005); prior work is *pre*-historical. Lyell's key insight was to explain the huge diversity of geological formations in terms of a relatively simple set of transformations applied repeatedly.

These analogies encourage our aspiration that by more systematically understanding the *relationships* between machine learning problems and how they can be *transformed* into each other, we will develop a better organised and more powerful toolkit for solving existing and future problems, and will make progress along the lines suggested by Hand (1994).

Acknowledgments

This work was motivated in part by discussions with Alekh Agarwal, Arthur Gretton, Ulrike von Luxburg and Bernhard Schölkopf in 2006. The authors were also aided by helpful conversations with Suvrit Sra and Bharath Sriperumbudur in Tübingen in 2008. Thanks also to Marcus Hutter for reading an early version of this paper and suggesting some useful changes and to Tim van Erven for finding several errors. Thanks also to the editor and reviewers for going along with our plan of a single (large!) integrated paper and their detailed and insightful reviews.

Parts of this work appeared earlier in the International Conference on Machine Learning 2009, and the Conference on Learning Theory, 2009.

This work was supported by the Australian Research Council and NICTA. NICTA is supported by the Australian Government through Backing Australia's Ability.

Appendix A. Proofs

This appendix presents the proofs that were omitted in the main body of the paper.

A.1 Proof of Corollary 3

Integration by parts of $t\phi''(t)$ gives $\int_0^1 t\,\phi''(t)\,dt = \phi'(1) - (\phi(1) - \phi(0))$ which can be rearranged to give

$$\phi'(1) = \int_0^1 t\,\phi''(t)\,dt + (\phi(1) - \phi(0)).$$

Substituting this into the Taylor expansion of $\phi(s)$ about 1 yields

$$\begin{aligned}
\phi(s) &= \phi(1) + \phi'(1)(s-1) + \int_s^1 (t-s)\,\phi''(t)\,dt \\
&= \phi(1) + \left[\int_0^1 t\,\phi''(t)\,dt + (\phi(1) - \phi(0))\right](s-1) + \int_0^1 (t-s)_+\,\phi''(t)\,dt \\
&= \phi(1) + (\phi(1) - \phi(0))(s-1) + \int_0^1 t(s-1)\,\phi''(t)\,dt + \int_0^1 (t-s)_+\,\phi''(t)\,dt \\
&= \phi(0) + (\phi(1) - \phi(0))s - \int_0^1 \psi(s,t)\,\phi''(t)\,dt,
\end{aligned}$$

where $\psi(s,t) := \min\{(1-t)s, (1-s)t\}$. This form of ψ is valid since

$$\begin{aligned}
-(t(s-1) + (t-s)_+) &= \begin{cases} -ts + t - t + s, & t \geq s \\ -ts + t, & t < s \end{cases} \\
&= \begin{cases} s - ts, & t \geq s \\ t - ts, & t < s \end{cases} \\
&= \min\{(1-t)s, (1-s)t\}
\end{aligned}$$

as required.

A.2 Proof of Theorem 6

Expanding the definition of the Jensen gap using the definition of ψ gives

$$\begin{aligned}
\mathbb{J}_\mu[\psi(\mathsf{S})] &= \mathbb{E}_\mu[\psi(\mathsf{S})] - \psi(\mathbb{E}_\mu[\mathsf{S}]) \\
&= \mathbb{E}_\mu[\phi(\mathsf{S}) + b\mathsf{S} + a] - (\phi(\mathbb{E}_\mu[\mathsf{S}]) + b\mathbb{E}_\mu[\mathsf{S}] + a) \\
&= \mathbb{E}_\mu[\phi(\mathsf{S})] + b\mathbb{E}_\mu[\mathsf{S}] + a - \phi(\mathbb{E}_\mu[\mathsf{S}]) - b\mathbb{E}_\mu[\mathsf{S}] - a \\
&= \mathbb{J}_\mu[\phi(\mathsf{S})]
\end{aligned}$$

as required.

A.3 Proof of Theorem 9

Proof Given a task $(\pi, P, Q; \ell)$ we need to first check that

$$f^{\pi}(t) := \underline{L}(\pi) - (\pi t + 1 - \pi)\underline{L}\left(\frac{\pi t}{\pi t + 1 - \pi}\right) \tag{67}$$

is convex and that $f^{\pi}(1) = 0$. This latter fact is obtained immediately by substituting $t = 1$ into $f^{\pi}(t)$ yielding $\underline{L}(\pi) - \underline{L}(\pi) = 0$. The convexity of f^{π} is guaranteed by Theorem 7, which shows that $\underline{L}$ is concave and the fact that the perspective transform of a convex function is always convex (see Section 2.1). Thus the function

$$t \mapsto I_{-\underline{L}}(\pi t, \pi t + 1 - \pi) = -(\pi t + 1 - \pi)\underline{L}\left(\frac{\pi t}{\pi t + 1 - \pi}\right)$$

is the composition of a convex function and an affine one and therefore convex.

Substituting (67) into the definition of f-divergence in (13) yields

$$\begin{aligned}
\mathbb{E}_Q\left[f^{\pi}(dP/dQ)\right] &= \mathbb{E}_Q\left[\underline{L}(\pi) - \left(\pi\frac{dP}{dQ} + 1 - \pi\right)\underline{L}\left(\frac{\pi dP}{\pi dP + (1-\pi)dQ}\right)\right] \\
&= \underline{L}(\pi) - \int_{\mathcal{X}} \underline{L}\left(\pi\frac{dP}{dM}\right) dM
\end{aligned}$$

since $dM = \pi dP + (1-\pi)dQ$. Recall that $\eta = \pi dP/dM$. Since $\underline{L}(\pi)$ is constant we note that $\underline{L}(\pi) = \mathbb{E}_M\left[\underline{L}(\pi)\right] = \underline{\mathbb{L}}(\pi, M)$ and so

$$\begin{aligned}
\mathbb{E}_Q\left[f^{\pi}(dP/dQ)\right] &= \underline{L}(\pi) - \mathbb{E}_M\left[\underline{L}(\eta)\right] \\
&= \underline{\mathbb{L}}(\pi, M) - \underline{\mathbb{L}}(\eta, M) \\
&= \Delta\underline{\mathbb{L}}(\eta, M)
\end{aligned}$$

as required for the forward direction.

Starting with

$$\underline{L}^{\pi}(\eta) := -\frac{1-\eta}{1-\pi} f\left(\frac{1-\pi}{\pi}\frac{\eta}{1-\eta}\right)$$

and substituting into the definition of statistical information in (20) gives us

$$\begin{aligned}
\Delta\underline{\mathbb{L}}^{\pi}(\eta, M) &= \mathbb{E}_M\left[\underline{L}^{\pi}(\pi)\right] - \mathbb{E}_M\left[\underline{L}^{\pi}(\eta)\right] \\
&= \int_{\mathcal{X}} -\frac{1-\pi}{1-\pi} f(1)\, dM - \int_{\mathcal{X}} -\frac{1-\eta}{1-\pi} f\left(\frac{1-\pi}{\pi}\frac{\eta}{1-\eta}\right) dM \\
&= 0 + \int_{\mathcal{X}} f\left(\frac{dP}{dQ}\right) dQ
\end{aligned}$$

since $f(1) = 0, dQ = (1-\eta)/(1-\pi)dM$ and

$$dP/dQ = \frac{1-\pi}{\pi}\frac{\eta}{1-\eta}$$

by the discussion in Section 4.1. This proves the converse statement of the theorem. ■

A.4 Proof of Corollary 13

Proof Let $f^\diamondsuit(t) := tf(1/t)$ denote the Csiszár-dual of f as described in Section 2.1 above. It is known (see (16) and, for example, Liese and Vajda, 2006) that

$$\mathbb{I}_f(P,Q) = \mathbb{I}_{f^\diamondsuit}(Q,P) \quad \text{if and only if} \quad f(t) = f^\diamondsuit(t) + c_1 t + c_2$$

for some $c_1, c_2 \in \mathbb{R}$. Since f and γ are related by $f''\left(\frac{1-\pi}{\pi}\right) = \pi^3\gamma(\pi)$ we can argue as follows. Observe that $f^{\diamondsuit\prime}(t) = f(1/t) - f'(1/t)/t$ and $f^{\diamondsuit\prime\prime}(t) = f''(1/t)/t^3$. Hence $f^{\diamondsuit\prime\prime}\left(\frac{1-\pi}{\pi}\right) = f''\left(\frac{\pi}{1-\pi}\right)\left(\frac{\pi}{1-\pi}\right)^3$. Let $\pi' = 1-\pi$. Thus $\frac{1-\pi}{\pi} = \frac{\pi'}{1-\pi'}$. Hence

$$\begin{aligned} f^{\diamondsuit\prime\prime}\left(\frac{1-\pi}{\pi}\right) &= f''\left(\frac{1-\pi'}{\pi'}\right)\left(\frac{\pi}{1-\pi}\right)^3 \\ &= \pi'^3\gamma(\pi')\left(\frac{\pi}{1-\pi}\right)^3 \\ &= \pi^3\gamma(1-\pi). \end{aligned}$$

Thus if $\gamma(1-\pi) = \gamma(\pi)$, we have shown $\pi \mapsto \gamma(1-\pi)$ is the weight corresponding to $f^\diamondsuit$. Observing that $\frac{\partial^2}{\partial t^2}(f^\diamondsuit(t) + c_1 t + c_2) = f^{\diamondsuit\prime\prime}$ concludes the proof. ■

A.5 Proof of Theorem 18

Proof Theorem 9 shows that

$$\underline{L}^\pi(\eta) = -\frac{1-\eta}{1-\pi} f\left(\frac{1-\pi}{\pi}\frac{\eta}{1-\eta}\right). \tag{68}$$

and we have seen from (32) that $w^\pi(c) = -(\underline{L}^\pi)''(c)$. The remainder of this proof involves taking the second derivative of $\underline{L}$, doing some messy algebra and matching the result to the relationship between γ and f'' in (Equation 28).

Letting $r_\pi = r_\pi(\eta) = \frac{1-\pi}{\pi}\frac{\eta}{1-\eta}$ and taking derivatives of (68) yields

$$\begin{aligned} -(\underline{L}^\pi)'(\eta) &= (1-\pi)^{-1}[-f(r_\pi) + (1-\eta)f'(r_\pi)r'_\pi] \\ -(\underline{L}^\pi)''(\eta) &= (1-\pi)^{-1}[-f'(r_\pi)r'_\pi + (1-\eta)(f'(r_\pi)r''_\pi + f''(r_\pi)(r'_\pi)^2) - f'(r_\pi)r'_\pi] \\ &= (1-\pi)^{-1}[(-2r'_\pi + (1-\eta)r''_\pi)f'(r_\pi) + (1-\eta)(r''_\pi)^2 f''(r_\pi)]. \end{aligned}$$

However, the form of r_π means $r'_\pi = \frac{1-\pi}{\pi}\frac{1}{(1-\eta)^2}$ and so $r''_\pi = \frac{1-\pi}{\pi}\frac{2}{(1-\eta)^3}$. This means the coefficient of $f'(r_\pi)$ in the above expression vanishes

$$(-2r'_\pi + (1-\eta)r''_\pi) = \frac{1-\pi}{\pi}\left[\frac{-2}{(1-\eta)^2} + (1-\eta)\frac{2}{(1-\eta)^3}\right] = 0.$$

Substituting this back into $-(\underline{L})''$ gives us

$$\begin{aligned} -(\underline{L}^\pi)''(\eta) &= \frac{1-\eta}{1-\pi} f''(r_\pi)(r'_\pi)^2 \\ &= \frac{1-\eta}{1-\pi} f''\left(\frac{1-\pi}{\pi}\frac{\eta}{1-\eta}\right)\frac{(1-\pi)^2}{\pi^2}\frac{1}{(1-\eta)^4} \\ w(\eta) &= \frac{1-\pi}{\pi^2(1-\eta)^3} f''\left(\frac{1-\pi}{\pi}\frac{\eta}{1-\eta}\right). \end{aligned}$$

By Equation 28 we have

$$\gamma(t) = \frac{1}{t^3} f''\left(\frac{1-t}{t}\right).$$

Letting $t = \frac{(1-c)\pi}{(1-c)\pi+(1-\pi)c}$ in that expression gives

$$\gamma\left(\frac{(1-c)\pi}{\nu(\pi,c)}\right) = \frac{\nu(\pi,c)^3}{(1-c)^3\pi^3} f''\left(\frac{1-\pi}{\pi}\frac{c}{1-c}\right).$$

Thus

$$\frac{\pi(1-\pi)}{\nu(\pi,c)^3}\gamma\left(\frac{(1-c)\pi}{\nu(\pi,c)}\right) = \frac{1-\pi}{\pi^2(1-c)^3} f''\left(\frac{1-\pi}{\pi}\frac{c}{1-c}\right) = w(c)$$

as required. The argument to show the inverse relationship is essentially the same. ■

A.6 Proof of Theorem 22

Proof Consider the right side of (42) and differentiate with respect to α:

$$\frac{\partial}{\partial\alpha}(1-\pi)\alpha + \pi(1-\beta(\alpha)) = (1-\pi) - \pi\beta'(\alpha).$$

Setting this to zero we have $(1-\pi) = \pi\beta'(\alpha)$ and thus $\beta'(\alpha) = \frac{1-\pi}{\pi}$. Since β is monotonically increasing and concave, β' is monotonically decreasing and non-negative. Thus we can set

$$\alpha = \beta'^{-1}\left(\frac{1-\pi}{\pi}\right) \in [0,1].$$

Substituting back into $(1-\pi)\alpha + \pi(1-\beta(\alpha))$ we obtain (44).

Now consider the right side of (43):

$$\frac{1}{\pi}((1-\pi)\alpha + \pi - \underline{\mathbb{L}}(\pi)). \tag{69}$$

Differentiating with respect to π we have $\frac{-\alpha}{\pi} - \frac{\underline{\mathbb{L}}'(\pi)}{\pi} + \frac{\underline{\mathbb{L}}(\pi)}{\pi^2}$. Setting this equal to zero we obtain

$$\begin{aligned} & \frac{-\alpha}{\pi} - \frac{\underline{\mathbb{L}}'(\pi)}{\pi} + \frac{\underline{\mathbb{L}}(\pi)}{\pi^2} = 0, \ \ \pi \in (0,1] \\ \Rightarrow \quad & \alpha + \pi\underline{\mathbb{L}}'(\pi) - \underline{\mathbb{L}}(\pi) = 0. \end{aligned}$$

Observing the definition of $\tilde{\underline{\mathbb{L}}}$ we thus have that $\tilde{\underline{\mathbb{L}}}(\pi) = \alpha$. Now

$$\begin{aligned} \tilde{\underline{\mathbb{L}}}'(\pi) &= \frac{\partial}{\partial\pi}(-\pi\underline{\mathbb{L}}'(\pi) + \underline{\mathbb{L}}(\pi)) \\ &= -\pi\underline{\mathbb{L}}''(\pi) - \underline{\mathbb{L}}'(\pi) + \underline{\mathbb{L}}'(\pi) \\ &= -\pi\underline{\mathbb{L}}''(\pi) \\ &\geq 0 \end{aligned}$$

since $\underline{\mathbb{L}}$ is concave. Thus $\tilde{\underline{\mathbb{L}}}(\cdot)$ is monotonically non-decreasing and we can write $\pi = \tilde{\underline{\mathbb{L}}}^{-1}(\alpha)$. In order to ensure $\pi \in [0,1]$ we substitute $\pi = \check{\underline{\mathbb{L}}}(\alpha)$ into (69) to obtain (45). ■

A.7 Proof of Theorem 21

Proof Since the true positive rate for $r \in \{-1,1\}^{\mathcal{X}}$ is $\mathrm{TP}_r = P(r^{-1}(1))$ and the false positive rate for r is $\mathrm{FP}_r = Q(r^{-1}(1))$ we have

$$\beta(\alpha, P, Q) = \sup_{r \in \{-1,1\}^{\mathcal{X}}} \{P(\mathcal{X}_r^+) : \ Q(\mathcal{X}_r^+) \leq \alpha\},$$

where $\mathcal{X}_r^+ := r^{-1}(1)$.

Noting that the 0-1 loss of r is simply its probability of error—that is, the average of the false positive and false negative rates—we have for each $\pi \in [0,1]$ that the Bayes optimal 0-1 loss is

$$\underline{\mathbb{L}}(\pi, P, Q) = \inf_{r \in \{-1,1\}^{\mathcal{X}}} \{(1-\pi)Q(\mathcal{X}_r^+) + \pi(1 - P(\mathcal{X}_r^+))\},$$

since the false negative rate $\mathrm{FN}_r = P(\mathcal{X} \setminus \mathcal{X}_r^+) = 1 - P(\mathcal{X}_r^+)$. Thus for all $\pi, \alpha \in [0,1]$, and all measurable functions $r \colon \mathcal{X} \to \{-1,1\}$,

$$\begin{aligned} \underline{\mathbb{L}}(\pi, P, Q) &\leq (1-\pi)Q(\mathcal{X}_r^+) + \pi(1 - P(\mathcal{X}_r^+)) \\ &\leq (1-\pi)\alpha + \pi(1 - P(\mathcal{X}_r^+)) \\ &\leq (1-\pi)\alpha + \pi(1 - \beta(\alpha, P, Q)). \end{aligned}$$

Thus, we see that $\underline{\mathbb{L}}(\pi, P, Q)$ is the largest number $\underline{\mathbb{L}}$ such that $(1-\pi)\alpha + \pi(1-\beta(\alpha)) \geq \underline{\mathbb{L}}$ for all $\alpha \in [0,1]$ and hence one can set

$$\underline{\mathbb{L}}(\pi, P, Q) = \underline{\mathbb{L}} = \min_{\alpha \in [0,1]} ((1-\pi)\alpha + \pi(1-\beta(\alpha))$$

for each $\pi \in [0,1]$.

Conversely, we can express the Neyman-Pearson function β in terms of the Bayes risk. That is, for any $\alpha \in [0,1]$, $\beta(\alpha, P, Q)$ is the largest number β such that

$$\begin{aligned} & \forall \pi \in [0,1] \quad (1-\pi)\alpha + \pi(1-\beta) \geq \underline{\mathbb{L}}(\pi) \\ \Leftrightarrow \ & \forall \pi \in [0,1] \quad (1-\pi)\alpha - \underline{\mathbb{L}}(\pi) \geq \pi(\beta - 1) \\ \Rightarrow \ & \forall \pi \in (0,1] \quad \frac{1}{\pi}((1-\pi)\alpha - \underline{\mathbb{L}}(\pi)) \geq \beta - 1 \\ \Leftrightarrow \ & \forall \pi \in (0,1] \quad \beta \leq \frac{1}{\pi}((1-\pi)\alpha + \pi - \underline{\mathbb{L}}(\pi)). \end{aligned}$$

Thus we can set

$$\beta(\alpha) = \inf_{\pi \in (0,1]} \frac{1}{\pi}((1-\pi)\alpha + \pi - \underline{\mathbb{L}}(\pi)), \ \ \alpha \in [0,1].$$

■

A.8 Proof of Lemma 23

Proof Let $\mathcal{X}' = [0,1]$ and P be the uniform distribution on $\mathcal{X}'$. Overload P and Q to also denote the respective cumulative distribution functions (i.e., $P(x) = P([0,x])$). Thus $P(\pi) = \pi$. Set $Q(\pi) = \phi(\pi)$. Since $\phi(\cdot)$ is increasing it suffices to consider $r(\cdot)$ of the form $r_\pi(x) = [\![x < \pi]\!]$. Hence

$$\beta(\alpha) = \max\{\phi(\pi) \colon 0 \leq \pi \leq 1,\ \pi \leq \alpha\},\ \alpha \in [0,1].$$

The maximum will always be obtained for $\pi = \alpha$ and thus $\beta(\alpha) = \phi(\alpha)$ for $\alpha \in [0,1]$. Finally, a pair of distributions on $\mathcal{X}$ can be constructed by embedding the connected component $\mathcal{C} \subset \mathcal{X}$ into $\mathcal{X}'$. Choose $g \colon \mathcal{C} \to \mathcal{X}'$ such that g is invertible. Such a g always exists since $\mathcal{C}$ is connected. Then g^{-1} induces distributions P' and Q' on $\mathcal{C}$ and thus on $\mathcal{X}$ by subsethood. ■

A.9 Proof of Corollary 24

Proof Choose a ψ satisfying the conditions and substitute into (43). This gives a corresponding $\phi(\cdot)$. We know from the preceding lemma that there exist P and Q such that $\beta(\cdot, P, Q) = \phi(\cdot)$ which corresponds to $\underline{\mathbb{L}}(\cdot, P, Q)$. Thus it remains to show that the function ϕ defined by

$$\phi(\alpha) = \inf_{\pi \in (0,1]} \frac{1}{\pi}((1-\pi)\alpha + \pi - \psi(\pi))$$

is concave and satisfies $\phi(1) = 1$. Observe that $\beta(1) = \inf_{\pi \in (0,1]} \frac{1-\psi(\pi)}{\pi}$. Now by the upper bound on ψ, we have $\frac{1-\psi(\pi)}{\pi} \geq \frac{1-1+\pi}{\pi} = \frac{1}{\pi} \geq 1$. But $\lim_{\pi \to 1} \frac{1-\psi(\pi)}{\pi} = 1$ and thus $\beta(1) = 1$. Finally note that

$$\beta(\alpha) = \inf_{\pi \in (0,1]} \left(\frac{1-\pi}{\pi} \right) \alpha + (1 - \psi(\pi)).$$

This is the lower envelope of a parameterized (by π) family of affine functions (in α) and is thus concave. ■

A.10 Proof of Lemma 26

Proof From Theorem 14 we know that $\underline{L}_c(\eta) = \min\{(1-\eta)c, (1-c)\eta\}$ and note that $(1-\eta)c \leq (1-c)\eta \iff c \leq \eta$. Then, by the definition of L_c and the identity $1 - [\![p]\!] = [\![\neg p]\!]$ we have

$$\begin{aligned} B_c(\eta, \hat{\eta}) &= (1-\eta)c[\![\hat{\eta} \geq c]\!] + (1-c)\eta[\![\hat{\eta} < c]\!] - \min\{(1-\eta)c, (1-c)\eta\} \\ &= (1-\eta)c[\![\hat{\eta} \geq c]\!] + (1-c)\eta[\![\hat{\eta} < c]\!] - (1-\eta)c[\![\eta \geq c]\!] - (1-c)\eta[\![\eta < c]\!] \\ &= (1-\eta)c([\![\hat{\eta} \geq c]\!] - [\![\eta \geq c]\!]) + (1-c)\eta([\![\hat{\eta} < c]\!] - [\![\eta < c]\!]). \end{aligned}$$

Note that $[\![\hat{\eta} \geq c]\!] - [\![\eta \geq c]\!]$ is either 1 or -1 depending on whether $\hat{\eta} \geq c > \eta$ or $\hat{\eta} < c \leq \eta$ and is zero otherwise. Similarly, $[\![\hat{\eta} < c]\!] - [\![\eta < c]\!]$ is 1 when $\hat{\eta} < c \leq \eta$, is -1 when $\hat{\eta} \geq c > \eta$ and is zero

otherwise. This means

$$\begin{aligned} B_c(\eta,\hat{\eta}) &= \begin{cases} (1-\eta)c-(1-c)\eta, & \hat{\eta}\geq c>\eta \\ -(1-\eta)c+(1-c)\eta, & \eta\geq c>\hat{\eta} \end{cases} \\ &= \begin{cases} c-\eta, & \hat{\eta}\geq c>\eta \\ \eta-c, & \eta\geq c>\hat{\eta} \end{cases} \\ &= |\eta-c|[\![\min\{\eta,\hat{\eta}\}\leq c<\max\{\eta,\hat{\eta}\}]\!] \end{aligned}$$

as required. ■

A.11 Proof of Theorem 33

Proof Let $\mathcal{C} := (\operatorname{sgn}\mathcal{R}+1)/2 \subseteq \{0,1\}^{\mathcal{X}}$ and so $\operatorname{sgn}\mathcal{R} = 2\mathcal{C}-1$. Then

$$\begin{aligned} \mathbb{L}_{\mathcal{C}}^{0-1}(\pi,P,Q) &= \inf_{r\in\mathcal{C}} \mathbb{E}_{(\mathsf{X},\mathsf{Y})\sim\mathbb{P}}\ell^{0-1}(r(\mathsf{X}),\mathsf{Y}) \\ &= \inf_{r\in\mathcal{C}} \left(\pi\mathbb{E}_{\mathsf{X}\sim P}\ell^{0-1}(r(\mathsf{X}),0)+(1-\pi)\mathbb{E}_{\mathsf{X}\sim Q}\ell^{0-1}(r(\mathsf{X}),1)\right) \\ &= \inf_{r\in\mathcal{C}} (\pi\mathbb{E}_{\mathsf{X}\sim P}[\![r(\mathsf{X})=1]\!]+(1-\pi)\mathbb{E}_{\mathsf{X}\sim Q}[\![r(\mathsf{X})=0]\!]) \\ &= \inf_{r\in\mathcal{C}} (\pi\mathbb{E}_P r+(1-\pi)\mathbb{E}_Q(1-r)) \end{aligned}$$

since $\operatorname{Ran} r=\{0,1\} \Rightarrow \mathbb{E}_{\mathsf{X}\sim P}[\![r(\mathsf{X})=1]\!] = \mathbb{E}_{\mathsf{X}\sim P} r(\mathsf{X})$ and $\mathbb{E}_{\mathsf{X}\sim Q}[\![r(\mathsf{X})=0]\!] = \mathbb{E}_{\mathsf{X}\sim Q}(1-r(\mathsf{X}))$. Let $\rho = 2r-1 \in 2\mathcal{C}-1$. Thus $r=\frac{\rho+1}{2}$. Hence

$$\begin{aligned} \mathbb{L}_{\mathcal{C}}^{0-1}(\pi,P,Q) &= \inf_{\rho\in 2\mathcal{C}-1} \left(\pi\mathbb{E}_P\left(\frac{\rho+1}{2}\right)-(1-\pi)\mathbb{E}_Q\left(1-\frac{\rho+1}{2}\right)\right) \\ &= \frac{1}{2}\inf_{\rho\in 2\mathcal{C}-1}(\pi\mathbb{E}_P(\rho+1)+(1-\pi)\mathbb{E}_Q(1-\rho)) \\ &= \frac{1}{2}\inf_{\rho\in 2\mathcal{C}-1}(\pi\mathbb{E}_P\rho+(1-\pi)\mathbb{E}_Q(-\rho)+\pi+(1-\pi)) \\ &= \frac{1}{2}+\frac{1}{2}\inf_{\rho\in 2\mathcal{C}-1}(\pi\mathbb{E}_P\rho-(1-\pi)\mathbb{E}_Q\rho) \\ &= \frac{1}{2}-\frac{1}{2}\sup_{\rho\in 2\mathcal{C}-1}(\pi\mathbb{E}_P(-\rho)-(1-\pi)\mathbb{E}_Q(-\rho)). \end{aligned}$$

Since $\mathcal{R}$ is symmetric about zero, $\operatorname{sgn}(\mathcal{R}) = 2\mathcal{C}-1$, $\mathcal{C}\subseteq\{0,1\}^{\mathcal{X}}$ is symmetric about $\frac{1}{2}$; that is, $\rho\in\mathcal{C}\Rightarrow(1-\rho)\in\mathcal{C}$. Thus

$$\begin{aligned} \mathbb{L}_{\mathcal{C}}^{0-1}(\pi,P,Q) &= \frac{1}{2}-\frac{1}{2}\sup_{\rho\in 2\mathcal{C}-1}(\pi\mathbb{E}_P\rho-(1-\pi)\mathbb{E}_Q\rho) \\ &= \frac{1}{2}-\frac{1}{4}V_{2\mathcal{C}-1,\pi}(P,Q) \\ &= \frac{1}{2}-\frac{1}{4}V_{\operatorname{sgn}\mathcal{R},\pi}(P,Q). \end{aligned} \tag{70}$$

Since by assumption $\operatorname{sgn}\mathcal{R} \subseteq \mathcal{R}$, the supremum in (63) will be ± 1-valued everywhere. Thus $V_{\operatorname{sgn}\mathcal{R},\pi}(P,Q) = V_{\mathcal{R},\pi}(P,Q)$. Combining this fact with (70) leads to (64).

Finally observe that by replacing inf and sup by arg min and arg max the final part of the theorem is apparent. ■

A.12 Pinsker Theorems

Proof (Theorem 30) Given a binary experiment (P,Q) denote the corresponding statistical information as

$$\phi(\pi) = \phi_{(P,Q)}(\pi) := \Delta\underline{\mathbb{L}}^{0-1}(\pi,P,Q) = \pi \wedge (1-\pi) - \psi_{(P,Q)}(\pi),$$

where $\psi_{(P,Q)}(\pi) = \psi(\pi) = \underline{\mathbb{L}}^{0-1}(\pi,P,Q)$. We know that ψ is non-negative and concave and satisfies $\psi(\pi) \leq \pi \wedge (1-\pi)$ and thus $\psi(0) = \psi(1) = 0$.

Since

$$\mathbb{I}_f(P,Q) = \int_0^1 \phi(\pi)\gamma_f(\pi)d\pi, \tag{71}$$

$\mathbb{I}_f(P,Q)$ is minimized by minimizing $\phi_{(P,Q)}$ over all (P,Q) such that

$$\phi(\pi_i) = \phi_i = \pi_i \wedge (1-\pi_i) - \psi_{(P,Q)}(\pi_i).$$

Let $\psi_i := \psi(\pi_i) = \frac{1}{2} - \frac{1}{4}V_{\pi_i}(P,Q)$. The problem becomes:

$$\begin{aligned}
\text{Given } (\pi_i,\psi_i)_{i=1}^n \qquad & \text{find the maximal } \psi\colon [0,1] \to [0,\tfrac{1}{2}] \text{ such that} && (72)\\
& \psi(\pi_i) = \psi_i, \;\; i = 0,\ldots,n+1, && (73)\\
& \psi(\pi) \leq \pi \wedge (1-\pi), \;\; \pi \in [0,1], && (74)\\
& \psi \text{ is concave.} && (75)
\end{aligned}$$

This will tell us the optimal ϕ to use since optimising over ψ is equivalent to optimizing over $\underline{\mathbb{L}}(\cdot,P,Q)$. Under the additional assumption on $\mathcal{X}$, Corollary 24 implies that for any ψ satisfying (73), (74) and (75) there exists P,Q such that $\underline{\mathbb{L}}(\cdot,P,Q) = \psi(\cdot)$.

Let Ψ be the set of piecewise linear concave functions on $[0,1]$ having $n+1$ segments such that $\psi \in \Psi \Rightarrow \psi$ satisfies (73) and (74). We now show that in order to solve (72) it suffices to consider $\psi \in \Psi$.

If g is a concave function on $\mathbb{R}$, then

$$\eth g(x) := \{s \in \mathbb{R}\colon\, g(y) \leq g(x) + \langle s, y-x\rangle,\; y \in \mathbb{R}\}$$

denote the *sup-differential* of g at x. (This is the obvious analogue of the *sub*-differential for convex functions Rockafellar, 1970.) Suppose $\tilde{\psi}$ is a general concave function satisfying (73) and (74). For $i = 1,\ldots,n$, let

$$G_i^{\tilde{\psi}} := \left\{ [0,1] \ni g_i^{\tilde{\psi}} : \pi_i \mapsto \psi_i \in \mathbb{R} \;\text{ is linear and }\; \tfrac{\partial}{\partial\pi} g_i^{\tilde{\psi}}(\pi)\Big|_{\pi=\pi_i} \in \eth\tilde{\psi}(\pi_i) \right\}.$$

Observe that by concavity, for all concave $\tilde{\psi}$ satisfying (73) and (74), for all $g \in \bigcup_{i=1}^n G_i^{\tilde{\psi}}$, $g(\pi) \geq \psi(\pi), \pi \in [0,1]$.

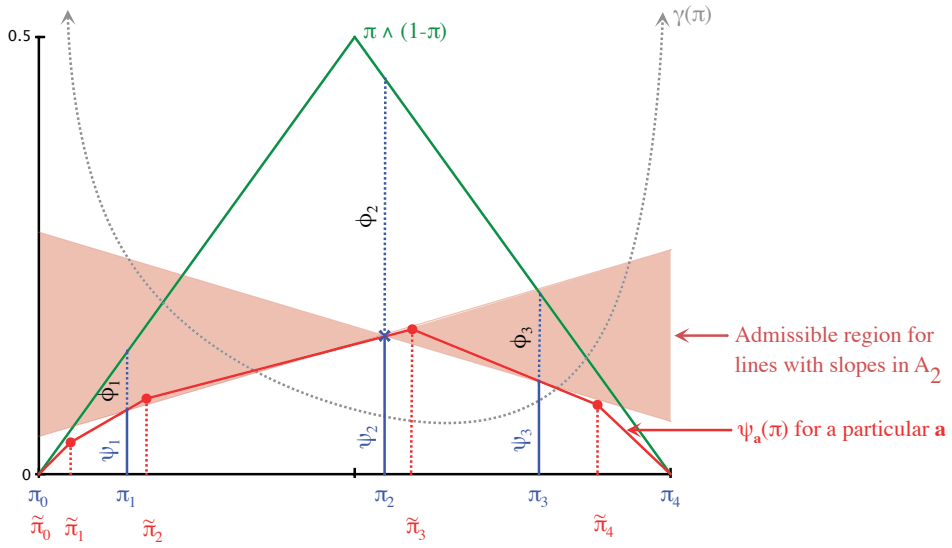


Figure 8: Illustration of construction of optimal $\psi(\pi) = \mathbb{L}(\pi, P, Q)$. The optimal ψ is piecewise linear such that $\psi(\pi_i) = \psi_i$, $i = 0, \ldots, n+1$.

Thus given any such $\tilde{\psi}$, one can always construct

$$\psi^*(\pi) = \min(g_1^{\tilde{\psi}}(\pi), \ldots, g_n^{\tilde{\psi}}(\pi)) \tag{76}$$

such that ψ^* is concave, satisfies (73) and $\psi^*(\pi) \geq \tilde{\psi}(\pi)$, for all $\pi \in [0,1]$. It remains to take account of (74). That is trivially done by setting

$$\psi(\pi) = \min(\psi^*(\pi), \pi \wedge (1-\pi)) \tag{77}$$

which remains concave and piecewise linear (although with potentially one additional linear segment). Finally, the pointwise smallest concave ψ satisfying (73) and (74) is the piecewise linear function connecting the points $(0,0), (\pi_1, \psi_1), (\pi_2, \psi_2), \ldots, (\pi_m, \psi_m), (1,0)$.

Let $g\colon [0,1] \to [0, \frac{1}{2}]$ be this function which can be written explicitly as

$$g(\pi) = \left(\psi_i + \frac{(\psi_{i+1} - \psi)(\pi - \pi_i)}{\pi_{i+1} - \pi_i} \right) \cdot [\![\pi \in [\pi_i, \pi_{i+1}]]\!], \quad i = 0, \ldots, n,$$

where we have defined $\pi_0 := 0, \psi_0 := 0, \pi_{n+1} := 1$ and $\psi_{n+1} := 0$.

We now explicitly parameterize this family of functions. Let $p_i\colon [0,1] \to \mathbb{R}$ denote the affine segment the graph of which passes through (π_i, ψ_i), $i = 0, \ldots, n+1$. Write $p_i(\pi) = a_i \pi + b_i$. We know that $p_i(\pi_i) = \psi_i$ and thus

$$b_i = \psi_i - a_i \pi_i, \quad i = 0, \ldots, n+1.$$

In order to determine the constraints on a_i, since g is concave and minorizes ψ, it suffices to only consider $(\pi_{i-1}, g(\pi_{i-1}))$ and $(\pi_{i+1}, g(\pi_{i+1}))$ for $i = 1, \ldots, n$. We have (for $i = 1, \ldots, n$)

$$\begin{aligned}
& p_i(\pi_{i-1}) && \geq g(\pi_{i-1}) \\
\Rightarrow \quad & a_i \pi_{i-1} + b_i && \geq \psi_{i-1} \\
\Rightarrow \quad & a_i \pi_{i-1} + \psi_i - a_i \pi_i && \geq \psi_{i-1} \\
\Rightarrow \quad & a_i \underbrace{(\pi_{i-1} - \pi_i)}_{<0} && \geq \psi_{i-1} - \psi_i \\
\Rightarrow \quad & a_i && \leq \frac{\psi_{i-1} - \psi_i}{\pi_{i-1} - \pi_i}.
\end{aligned}$$

Similarly we have (for $i = 1, \ldots, n$)

$$\begin{aligned}
& p_i(\pi_{i+1}) && \geq g(\pi_{i+1}) \\
\Rightarrow \quad & a_i \pi_{i+1} + b_i && \geq \psi_{i+1} \\
\Rightarrow \quad & a_i \pi_{i+1} + \psi_i - a_i \pi_i && \geq \psi_{i+1} \\
\Rightarrow \quad & a_i \underbrace{(\pi_{i+1} - \pi_i)}_{>0} && \geq \psi_{i+1} - \psi_i \\
\Rightarrow \quad & a_i && \geq \frac{\psi_{i+1} - \psi_i}{\pi_{i+1} - \pi_i}.
\end{aligned}$$

We now determine the points at which ψ defined by (76) and (77) change slope. That occurs at the points π when

$$\begin{aligned}
& p_i(\pi) && = p_{i+1}(\pi) \\
\Rightarrow \quad & a_i \pi + \psi_i - a_i \pi_i && = a_{i+1} \pi + \psi_{i+1} - a_{i+1} \pi_{i+1} \\
\Rightarrow \quad & (a_{i+1} - a_i) \pi && = \psi_i - \psi_{i+1} + a_{i+1} \pi_{i+1} - a_i \pi_i \\
\Rightarrow \quad & \pi && = \frac{\psi_i - \psi_{i+1} + a_{i+1} \pi_{i+1}}{a_{i+1} - a_i} \\
& && =: \tilde{\pi}_i
\end{aligned}$$

for $i = 0, \ldots, n$. Thus

$$\psi(\pi) = p_i(\pi), \quad \pi \in [\tilde{\pi}_{i-1}, \tilde{\pi}_i], \ i = 1, \ldots, n.$$

Let $\mathbf{a} = (a_1, \ldots, a_n)$. We explicitly denote the dependence of ψ on $\mathbf{a}$ by writing $\psi_{\mathbf{a}}$. Let

$$\begin{aligned}
\phi_{\mathbf{a}}(\pi) \quad &:= \quad \pi \wedge (1 - \pi) - \psi_{\mathbf{a}}(\pi) \\
&= \quad \alpha_{\mathbf{a},i} \pi + \beta_{\mathbf{a},i}, \quad \pi \in [\pi_{i-1}, \pi_i], \ i = 1, \ldots, n+1,
\end{aligned}$$

where $\mathbf{a} \in A_n$ (see (50)), π_i, $\alpha_{\mathbf{a},i}$ and $\beta_{\mathbf{a},i}$ are defined by (53), (54) and (55) respectively. The extra segment induced at index j (see (52)) is needed since $\pi \mapsto \pi \wedge (1 - \pi)$ has a slope change at $\pi = \frac{1}{2}$. Thus in general, $\phi_{\mathbf{a}}$ is piecewise linear with $n + 2$ segments (recall i ranges from 0 to $n + 2$); if $\tilde{\pi}_{k+1} = \frac{1}{2}$ for some $k \in \{1, \ldots, n\}$, then there will be only $n + 1$ non-trivial segments.

Thus

$$\left\{ \pi \mapsto \sum_{i=0}^{n} \phi_{\mathbf{a}}(\pi) \cdot [\![\pi \in [\pi_i, \pi_{i+1}]]\!] : \mathbf{a} \in A_n \right\}$$

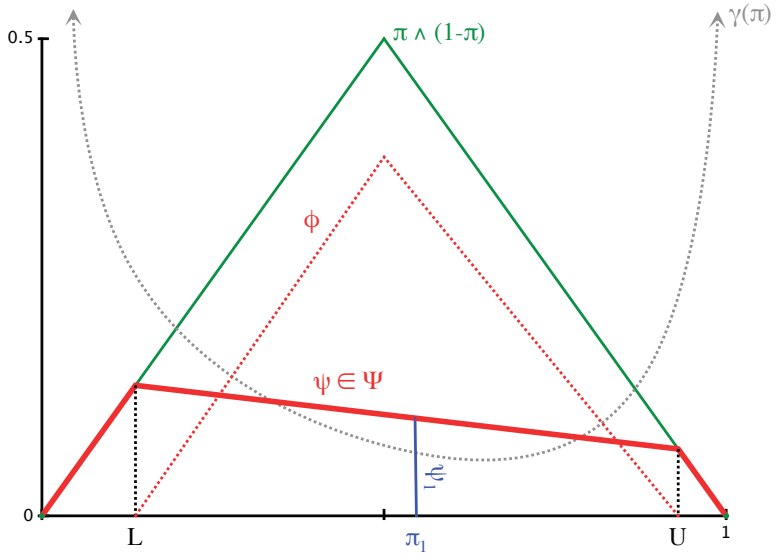


Figure 9: The optimisation problem when $n = 1$. Given ψ_1, there are many risk curves consistent with it. The optimisation problem involves finding the piecewise linear concave risk curve $\psi \in \Psi$ and the corresponding $\phi = \pi \wedge (1-\pi)$ that maximises $\mathbb{I}_f$. L and U are defined in the text.

is the set of ϕ consistent with the constraints and A_n is defined in (50). Thus substituting into (71), interchanging the order of summation and integration and optimizing we have shown (56). The tightness has already been argued: under the additional assumption on $\mathcal{X}$, since there is no slop in the argument above since every ϕ satisfying the constraints is the Bayes risk function for some (P,Q). ■

Proof (Theorem 31) In this case $n = 1$ and the optimal ψ function will be piecewise linear, concave, and its graph will pass through (π_1, ψ_1). Thus the optimal ϕ will be of the form

$$\phi(\pi) = \begin{cases} 0, & \pi \in [0, L] \cup [U, 1] \\ \pi - (a\pi + b), & \pi \in [L, \frac{1}{2}] \\ (1-\pi) - (a\pi + b), & \pi \in [\frac{1}{2}, U]. \end{cases}$$

where $a\pi_1 + b = \psi_1 \Rightarrow b = \psi_1 - a\pi_1$ and $a \in [-2\psi_1, 2\psi_1]$ (see Figure 9). For variational divergence, $\pi_1 = \frac{1}{2}$ and thus

$$\psi_1 = \pi_1 \wedge (1-\pi_1) - \frac{V}{4} = \frac{1}{2} - \frac{V}{4} \tag{78}$$

and so $\phi_1 = V/4$. We can thus determine L and U:

$$\begin{aligned} & aL+b & &= L \\ \Rightarrow \quad & aL+\psi_1-a\pi_1 & &= L \\ \Rightarrow \quad & L & &= \frac{a\pi_1-\psi_1}{a-1}. \end{aligned}$$

Similarly $aU+b = 1-U \Rightarrow U = \frac{1-\psi_1+a\pi_1}{a+1}$ and thus

$$\mathbb{I}_f(P,Q) \geq \min_{a\in[-2\psi_1,2\psi_1]} \int_{\frac{a\pi_1-\psi_1}{a-1}}^{\frac{1}{2}} [(1-a)\pi-\psi_1+a\pi_1]\gamma_f(\pi)d\pi + \int_{\frac{1}{2}}^{\frac{1-\psi_1+a\pi_1}{a+1}} [(-a-1)\pi-\psi_1+a\pi_1+1]\gamma_f(\pi)d\pi. \tag{79}$$

If γ_f is symmetric about $\pi = \frac{1}{2}$ (so by Corollary 13 $\mathbb{I}_f$ is symmetric) and convex and $\pi_1 = \frac{1}{2}$, then the optimal $a = 0$. Thus in that case,

$$\begin{aligned} \mathbb{I}_f(P,Q) &\geq 2\int_{\psi_1}^{\frac{1}{2}} (\pi-\psi_1)\gamma_f(\pi)d\pi \\ &= 2\left[(\tfrac{1}{2}-\psi_1)\Gamma_f(\tfrac{1}{2})+\Gamma_f(\psi_1)-\Gamma_f(\tfrac{1}{2})\right] \\ &= 2\left[\tfrac{V}{4}\Gamma_f(\tfrac{1}{2})+\Gamma_f\left(\tfrac{1}{2}-\tfrac{V}{4}\right)-\Gamma_f(\tfrac{1}{2})\right]. \end{aligned} \tag{80}$$

Appendix B. Examples of Generalised Pinsker Inequality

Combining the above with (78) leads to a range of Pinsker style bounds for symmetric $\mathbb{I}_f$:

Jeffrey's Divergence $J(P,Q) = \mathrm{KL}(P,Q)+\mathrm{KL}(Q,P)$. Thus $\gamma(\pi) = \frac{1}{\pi^2(1-\pi)} + \frac{1}{\pi(1-\pi)^2} = \frac{1}{\pi^2(1-\pi)^2}$. (As a check, $f(t) = (t-1)\ln(t)$, $f''(t) = \frac{t+1}{t^2}$ and so $\gamma_f(\pi) = \frac{1}{\pi^3} f'\left(\frac{1-\pi}{\pi}\right) = \frac{1}{\pi^2(1-\pi)^2}$.) Thus

$$\begin{aligned} J(P,Q) &\geq 2\int_{\psi_1}^{1/2} \frac{(\pi-\psi_1)}{\pi^2(1-\pi)^2}d\pi \\ &= (4\psi_1-2)(\ln(\psi_1)-\ln(1-\psi_1)). \end{aligned}$$

Substituting $\psi_1 = \frac{1}{2}-\frac{V}{4}$ gives

$$J(P,Q) \geq V\ln\left(\frac{2+V}{2-V}\right).$$

Observe that the above bound behaves like V^2 for small V, and $V\ln\left(\frac{2+V}{2-V}\right) \geq V^2$ for $V \in [0,2]$. Using the traditional Pinkser inequality ($\mathrm{KL}(P,Q) \geq V^2/2$) we have

$$\begin{aligned} J(P,Q) &= \mathrm{KL}(P,Q)+\mathrm{KL}(Q,P) \\ &\geq \frac{V^2}{2}+\frac{V^2}{2} \\ &= V^2 \end{aligned}$$

Jensen-Shannon Divergence Here $f(t) = \frac{t}{2}\ln t - \frac{(t+1)}{2}\ln(t+1) + \ln 2$ and thus the weight function $\gamma_f(\pi) = \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right) = \frac{1}{2\pi(1-\pi)}$. Thus

$$\begin{aligned} JS(P,Q) &= 2\int_{\psi_1}^{\frac{1}{2}} \frac{\pi-\psi_1}{2\pi(1-\pi)} d\pi \\ &= \ln(1-\psi_1) - \psi_1 \ln(1-\psi_1) + \psi_1 \ln\psi_1 + \ln(2). \end{aligned}$$

Substituting $\psi_1 = \frac{1}{2} - \frac{V}{4}$ leads to

$$JS(P,Q) \geq \left(\frac{1}{2} - \frac{V}{4}\right)\ln(2-V) + \left(\frac{1}{2} + \frac{V}{4}\right)\ln(2+V) - \ln(2).$$

Hellinger Divergence Here $f(t) = (\sqrt{t}-1)^2$. Consequently the weight function

$$\gamma_f(\pi) = \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right) = \frac{1}{\pi^3} \frac{1}{2\left((1-\pi)/\pi\right)^{3/2}} = \frac{1}{2[\pi(1-\pi)]^{3/2}}$$

and thus

$$\begin{aligned} h^2(P,Q) &\geq 2\int_{\psi_1}^{\frac{1}{2}} \frac{\pi-\psi_1}{2[\pi(1-\pi)]^{3/2}} d\pi \\ &= \frac{4\sqrt{\psi_1}(\psi_1-1) + 2\sqrt{1-\psi_1}}{\sqrt{1-\psi_1}} \\ &= \frac{4\sqrt{\frac{1}{2}-\frac{V}{4}}\left(\frac{1}{2}-\frac{V}{4}-1\right) + 2\sqrt{1-\frac{1}{2}+\frac{V}{4}}}{\sqrt{1-\frac{1}{2}+\frac{V}{4}}} \\ &= 2 - \frac{(2+V)\sqrt{2-V}}{\sqrt{2+V}} \\ &= 2 - \sqrt{4-V^2}. \end{aligned}$$

For small V, $2-\sqrt{4-V^2} \approx V^2/4$.

Arithmetic-Geometric Mean Divergence Here $f(t) = \frac{t+1}{2}\ln\left(\frac{t+1}{2\sqrt{t}}\right)$. Thus $f''(t) = \frac{t^2+1}{4t^2(t+1)}$ and hence $\gamma_f(\pi) = \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right) = \gamma_f(\pi) = \frac{2\pi^2-2\pi+1}{\pi^2(\pi-1)^2}$ and thus

$$\begin{aligned} T(P,Q) &\geq 2\int_{\psi_1}^{\frac{1}{2}} (\pi-\psi_1)\frac{2\pi^2-2\pi+1}{\pi^2(\pi-1)^2} d\pi \\ &= -\frac{1}{2}\ln(1-\psi) - \frac{1}{2}\ln(\psi) - \ln(2). \end{aligned}$$

Substituting $\psi_1 = \frac{1}{2} - \frac{V}{4}$ gives

$$\begin{aligned} T(P,Q) &\geq -\frac{1}{2}\ln\left(\frac{1}{2}+\frac{V}{4}\right) - \frac{1}{2}\ln\left(\frac{1}{2}-\frac{V}{4}\right) - \ln(2) \\ &= \ln\left(\frac{4}{\sqrt{4-V^2}}\right) - \ln(2). \end{aligned}$$

Symmetric χ^2-Divergence Here $\Psi(P,Q) = \chi^2(P,Q) + \chi^2(Q,P)$ and thus (see below) $\gamma_f(\pi) = \frac{2}{\pi^3} + \frac{2}{(1-\pi)^3}$. (As a check, from $f(t) = \frac{(t-1)^2(t+1)}{t}$ we have $f''(t) = \frac{2(t^3+1)}{t^3}$ and thus $\gamma_f(\pi) = \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right)$ gives the same result.)

$$\begin{aligned}\Psi(P,Q) &\geq 2\int_{\psi_1}^{\frac{1}{2}} (\pi - \psi_1)\left(\frac{2}{\pi^3} + \frac{2}{(1-\pi)^3}\right) d\pi \\ &= \frac{2(1+4\psi_1^2 - 4\psi_1)}{\psi_1(\psi_1 - 1)}.\end{aligned}$$

Substituting $\psi_1 = \frac{1}{2} - \frac{V}{4}$ gives $\Psi(P,Q) \geq \frac{8V^2}{4-V^2}$.

When γ_f is not symmetric, one needs to use (79) instead of the simpler (80). We consider two special cases.

χ^2-Divergence Here $f(t) = (t-1)^2$ and so $f''(t) = 2$ and hence $\gamma(\pi) = f''\left(\frac{1-\pi}{\pi}\right)/\pi^3 = \frac{2}{\pi^3}$ which is not symmetric. Upon substituting $2/\pi^3$ for $\gamma(\pi)$ in (79) and evaluating the integrals we obtain

$$\chi^2(P,Q) \geq 2 \min_{a \in [-2\psi_1, 2\psi_1]} \underbrace{\frac{1+4\psi_1^2-4\psi_1}{2\psi_1 - a} - \frac{1+4\psi_1^2-4\psi_1}{2\psi_1 - a - 2}}_{=:J(a,\psi_1)}.$$

One can then solve $\frac{\partial}{\partial a} J(a,\psi_1) = 0$ for a and one obtains $a^* = 2\psi_1 - 1$. Now $a^* > -2\psi_1$ only if $\psi_1 > \frac{1}{4}$. One can check that when $\psi_1 \leq \frac{1}{4}$, then $a \mapsto J(a,\psi_1)$ is monotonically increasing for $a \in [-2\psi_1, 2\psi_1]$ and hence the minimum occurs at $a^* = -2\psi_1$. Thus the value of a minimising $J(a,\psi_1)$ is

$$a^* = [\![\psi_1 > 1/4]\!](2\psi_1 - 1) + [\![\psi_1 \leq 1/4]\!](-2\psi_1).$$

Substituting the optimal value of a^* into $J(a,\psi_1)$ we obtain

$$J(a^*,\psi_1) = [\![\psi_1 > 1/4]\!](2 + 8\psi_1^2 - 8\psi_1) + [\![\psi_1 \leq 1/4]\!]\left(\frac{1+4\psi_1^2-4\psi}{4\psi} - \frac{1+4\psi_1^2-4\psi}{4\psi_1 - 2}\right).$$

Substituting $\psi_1 = \frac{1}{2} - \frac{V}{4}$ and observing that $V < 1 \Rightarrow \psi_1 > 1/4$ we obtain

$$\chi^2(P,Q) \geq [\![V < 1]\!] V^2 + [\![V \geq 1]\!] \frac{V}{(2-V)}.$$

Observe that the bound diverges to ∞ as $V \to 2$.

Kullback-Leibler Divergence In this case $f(t) = t\ln t$ and thus $f''(t) = 1/t$ and the weight function $\gamma_f(\pi) = \frac{1}{\pi^3} f''\left(\frac{1-\pi}{\pi}\right) = \frac{1}{\pi^2(1-\pi)}$ which is clearly not symmetric. From (79) we obtain

$$\mathrm{KL}(P,Q) \geq \min_{[-2\psi_1, 2\psi_1]} \left(1 - \frac{a}{2} - \psi_1\right) \ln\left(\frac{a + 2\psi_1 - 2}{a - 2\psi_1}\right) + \left(\frac{a}{2} + \psi_1\right) \ln\left(\frac{a + 2\psi_1}{a - 2\psi_1 + 2}\right).$$

Substituting $\psi_1 = \frac{1}{2} - \frac{V}{4}$ gives $\mathrm{KL}(P,Q) \geq \min_{a \in \left[\frac{V-2}{2}, \frac{2-V}{2}\right]} \delta_a(V)$, where

$$\delta_a(V) = \left(\frac{V + 2 - 2a}{4}\right) \ln\left(\frac{2a - 2 - V}{2a - 2 + V}\right) + \left(\frac{2a + 2 - V}{4}\right) \ln\left(\frac{2a + 2 - V}{2a + 2 + V}\right).$$

Set $\beta := 2a$ and we have (59).

■

Appendix C. Background and Prior Work

Specific prior results are referred to in the body of the paper. We now briefly indicate the broad sweep of prior work along the lines of the present paper.

The most important precursors and inspiration are the three nearly simultaneous[25] works by Buja et al. (2005), Liese and Vajda (2006) and Nguyen et al. (2005). The work by Dawid (2007) is also very similar in spirit to that presented here. A crucial difference is that he relies on a parametric viewpoint, and can use the machinery of Riemannian geometry. Zhang (2004a); Zhang and Matsuzoe (2009) have developed a number of connections between convex functions, the Bregman divergences they induce, and Riemannian geometry. All of the results in the present paper are, in contrast, "coordinate-free." The *motivation* of the present work is closely aligned with that of Hand (1994) whose avowed aim was to "stimulate debate about the need to formulate research questions sufficiently precisely that they may be unambiguously and correctly matched with statistical techniques." Hand and Vinciotti (2003) develop some refined machine learning tasks that can be viewed as weighted problems (in the sense of the weight functions we make extensive use of in this paper); confer Buja et al. (2005).

The paper presents a unification of sorts. This, in itself, is hardly new in machine learning. There are different approaches to unification. One distinction is between *Monistic* and *Pluralistic* approaches (James, 1909; Turkle and Papert, 1992); this corresponds to the hedgehog/fox distinction of Berlin (1953).

Monistic approaches aim for a single all encompassing theory.[26] A problem with most monistic approaches is that you have to accept it "all or nothing." There are many unifying approaches developed in Statistics and Machine learning that have left little trace; For example, Nelson's use of non-standard analysis (Nelson, 1987; Lutz and Musio, 2005) as the foundations for probability; Topsøe's (2006), Shafer and Vovk's (2001) game theory as a basis, and Le Cam's use of Riesz measures on a vector lattice to replace the traditional sample space (LeCam, 1964).

Pluralistic approaches are closer to what is proposed here (where, instead of searching for a single master representation, we study relationships and translations between a range of different representations). It resonates with Kiefer's assertion that "Statistics is too complex to be codified in terms of a simple prescription that is a panacea for all settings, and ...one must look as carefully as possible at a variety of possible procedures..." (Kiefer, 1977). Examples of existing pluralistic attempts include limited problem catalogs such as for different notions of *cost* (Turney, 2000) or a restricted set of problems (Raudys, 2001).

The decision theoretic approach (DeGroot, 1970; Berger, 1985; Kiefer, 1987) due to Wald (1950, 1949) is central to the present paper. The idea of seeking *primitives* for statistics dates back at least to the elementary experiments of Birnbaum (1961). The relationship between risks and Bregman divergences is studied by Grünwald and Dawid (2004) and Buja et al. (2005).

25. Nguyen et al. (2005) is dated 13 October, 2005, Liese and Vajda (2006) was received on 26 October 2005 and Buja et al. (2005) is dated 3 November 2005. Shen's PhD thesis (Shen, 2005), which contains most of the material in Buja et al. (2005), is dated 16 October 2005. The paper by Nguyen et al. (2005) has now appeared as Nguyen et al. (2009).

26. Monistic approaches can be categorised into at least four distinct categories. They are briefly summarised in Appendix C.1.

There are numerous possible definitions of information. Many of them are sterile; Csiszár (1978) and Aczél (1984) provide a critical analysis. Floridi (2004) discusses pluralistic versus monistic approach: is there one single definition of information, or should there be many different definitions depending on the particular problem? Our view, like Shannon (1948), is that there are many types. Shannon information was developed with communications problems in mind—there is no reason why it is the only notion of information that makes sense for learning and inference.

There are many known relationships between risks and divergences between distributions many of which we explicitly discuss later in the paper. General results include those due to Österreicher (2003), Österreicher and Vajda (1993), Gutenbrunner (1990), Liese and Vajda (2006), Goel and DeGroot (1979) and Golic (1987). Particular relations between risk in binary classification problems and f-divergences are not new (Poor and Thomas, 1977; Kailath, 1967). Some more general results that relate the choice of loss function in a binary learning problem to particular f-divergences between the class-conditional distributions have been (re)-discovered (Eguchi and Copas, 2001; Nguyen et al., 2005; Österreicher and Vajda, 1993). Known results relating different distances between probability distributions are summarised by Gibbs and Su (2002).

The idea of solving a machine learning problem by using a solution to some other learning problem is now called a *machine learning reduction* (Beygelzimer et al., 2008, 2005) The idea is not new. Equivalences are a natural structuring device and were explicit in Ashby's foundational work on cybernetics (Ashby, 1956), a precursor to Machine Learning. Ben-Bassat (1978) studied the concept of ε-equivalence, Conover and Iman (1981) showed how rank tests can be derived by applying nonparametric tests to order statistics, and Goldman et al. (1989) and Bartlett et al. (1996) used reductions for theoretical purposes. However recently there has been a large number of explicit constructions of reductions (Zadrozny et al., 2003; Langford, 2006; Beygelzimer et al., 2005; Langford and Beygelzimer, 2005; Langford and Zadrozny, 2005; Langford et al., 2006; Li and Lin, 2007; Beygelzimer et al., 2007; Langford, 2007; Scott and Davenport, 2007),or development of results which although not explicitly called reductions are effectively so (Brown et al., 2002; Brown and Low, 1996; Brown and Zhao, 2003; Chaudhuri and Loh, 2002; Cossock and Zhang, 2006; Cuevas and Fraiman, 1997; Domingos, 1999; Steinwart et al., 2005; Tasche, 2001). Two key differences between the recent machine learning reductions literature and the present paper is that our relationships between problems are (usually) exact (instead of approximate) and we work with the true underlying distributions (rather than finite sample distributions).

The theory of *Comparison of Experiments*, developed by Blackwell (1951, 1953), and significantly extended by LeCam (1964, 1986) is also related to the overall goal set out here. It has been used to define notions of isomorphism for statistical problems (Morse and Sacksteder, 1966; Sacksteder, 1967) and is the subject of three books (Strasser, 1985; Torgersen, 1991; Heyer, 1982) and a recent review (Goel and Ginebra, 2003). The key difference with the present work is that the comparison of experiments theory seeks results that hold for *all* loss functions rather than for a particular one; with a few exceptions (Torgersen, 1991, Chapter 10). Blackwell related comparisons to sufficient statistics and characterised comparisons. LeCam (1964) quantified comparisons in terms of the degree to which one experiment is "better than" another (the deficiency distance). There are very few known examples of deficiency distance (Carter, 2002). Furthermore LeCam's theory is formulated in a particularly abstract way to make its theorems elegant (Yang and Le Cam, 1999). Renowned probabilists concur that its arcane formulation has made it inaccessible (van der Vaart, 2002; Pollard, 2000; Strasser, 2000). Consequently the subject has had relatively limited impact.

Graphical representations have been used for a long while to better understand binary experiments. In the main body of the paper we develop connections between Receiver Operating Characteristic (ROC) curves, (Fawcett, 2006, 2004; Flach, 2003; Flach and Wu, 2005; Maxion and Roberts, 2004) the Area Under ROC Curve (AUC), (Cortes and Mohri, 2004; Hand, 2008; Hand and Till, 2001; Hanley and McNeil, 1982) and Cost Curves (Drummond and Holte, 2006; Torgersen, 1991). These can be seen as *representations* of Binary Experiments.

C.1 Summary of Previous "Monistic" Approaches to Unification

There are are range of different approaches to unifying machine learning from a monistic perspective:

Low level data interchange: There is a small amount of work on developing standards for interchanging data sets (Grossman et al., 2002; Carey et al., 2007; Wettschereck and Muller, 2001)—this is analogous to PDDL (Ghallab et al., 1998). There are also some limited higher level attempts such as ontologies (Soldatova and King, 2006) and general frameworks (Fayyad et al., 1996).

Modelling frameworks: To *solve* a machine learning problem, one needs models. There is a rich literature on graphical models (Jordan, 1999), factor graphs (Kschischang et al., 2001) and Markov logic networks (Domingos and Richardson, 2004; Richardson and Domingos, 2006) which have allowed the unification of sets of problems (Worthen and Stark, 2001), with a focus on the modelling and computational techniques for particular problems.

Comparison of frameworks: There are several philosophical frameworks/approaches to designing inference and learning algorithms. Barnett (1999), Bayarri and Berger (2004) and Berger (2003) compare and contrast these. They are effectively comparing different monistic frameworks, not comparing problems.

Overarching frameworks: These include frameworks such as Bayesian (Robert, 1994), information theoretic (Jenssen, 2005b; Harremoës, 1993), game-theoretic (Vovk et al., 2005; Grünwald and Dawid, 2004), MDL (Grünwald, 2007; Rissanen, 2007), regularised distance minimisation (Borwein and Lewis, 1991; Altun and Smola, 2006; Broniatowski, 2004), and more narrowly focussed "unifying frameworks" such as information geometry (Dawid, 2007; Eguchi, 2005), exponential families (Canu and Smola, 2006) and the information bottleneck (Tishby et al., 2000).

Appendix D. Examples and Prior Work on Surrogate Regret Bounds

Surrogate regret bounds have garnered interest in the machine learning community (Zhang, 2004b; Bartlett et al., 2006; Steinwart, 2007; Steinwart and Christmann, 2008). Steinwart and Christmann (2008, Chapter 3) have presented a good summary of recent work.

All of the recent work has been in terms of *margin losses* of the form

$$L^{\phi}(\eta,\hat{h}) = \eta\phi(\hat{h}) + (1-\eta)\phi(-\hat{h}).$$

As Buja et al. (2005) discuss, such margin losses can not capture the richness of all possible proper losses. Bartlett et al. (2006) prove that for any $\hat{h}$

$$\psi\left(L^{0-1}(\eta,\hat{h}) - \underline{L}^{0-1}(\eta)\right) \leq L^{\phi}(\eta,\hat{h}) - \underline{L}^{\phi}(\eta),$$

where $\psi = \tilde{\psi}^{\star\star}$ is the LF biconjugate of $\tilde{\psi}$,

$$\tilde{\psi}(\theta) = H^{-}\left(\frac{1+\theta}{2}\right) - H\left(\frac{1+\theta}{2}\right),$$

$H(\eta) = \underline{L}^{\phi}(\eta)$ and

$$H^{-}(\eta) = \inf_{\alpha:\, \alpha(2\eta-1)\leq 0} (\eta\phi(\alpha) + (1-\eta)\phi(-\alpha))$$

is the optimal conditional risk under the constraint that the sign of the argument α disagrees with $2\eta - 1$.

We will consider two examples presented by Bartlett et al. (2006) and show that the bounds we obtain with the above theorem match the results we obtain with Theorem 25.

Exponential Loss Consider the link $\hat{h} = \psi(\hat{\eta}) = \frac{1}{2}\ln\frac{\hat{\eta}}{1-\hat{\eta}}$ with corresponding inverse link $\hat{\eta} = \frac{1}{1+e^{-2\hat{h}}}$. Buja et al. (2005) showed that this link function combined with exponential margin loss $\phi(\gamma) = e^{-\gamma}$ results in a proper scoring rule

$$L(\eta, \hat{\eta}) = \eta\left(\frac{1-\hat{\eta}}{\hat{\eta}}\right)^{\frac{1}{2}} + (1-\eta)\left(\frac{\hat{\eta}}{1-\hat{\eta}}\right)^{\frac{1}{2}}.$$

From (32) we obtain

$$w(\eta) = \frac{1}{2[\eta(1-\eta)]^{\frac{3}{2}}}.$$

(Note Buja et al., 2005 have missed the factor of $\frac{1}{2}$.) Thus $W(\eta) = \frac{2\eta-1}{\sqrt{\eta(1-\eta)}}$ and $\overline{W}(\eta) = -2\sqrt{\eta(1-\eta)}$. Hence we obtain

$$\underline{L}(\eta) = 2\sqrt{\eta(1-\eta)} \tag{81}$$

and from (46) we obtain that if $B_{\frac{1}{2}}(\eta, \hat{\eta}) = \alpha$ then

$$B(\eta, \hat{\eta}) \geq 1 - \sqrt{1-4\alpha^2}. \tag{82}$$

Equations 81 and 82 match the results presented by Bartlett et al. (2006) upon noting that $B_{\frac{1}{2}}(\eta, \hat{\eta})$ measures the loss in terms of $\ell_{\frac{1}{2}}$ and Bartlett et al. (2006) used $\ell^{0-1} = 2\ell_{\frac{1}{2}}$.

Truncated Quadratic Loss Consider the margin loss $\phi(\hat{h}) = (1 + \hat{h} \vee 0)^2 = (2\hat{\eta} \vee 0)^2$ with link function $\hat{h}(\hat{\eta}) = 2\hat{\eta} - 1$. From (32) we obtain $\underline{L}(\eta) = 4\eta(1-\eta)$ and from (46) the regret bound $B(\eta, \hat{\eta}) \geq 4\alpha^2$. These match the results presented by Bartlett et al. (2006) when again it is noted we used $\ell_{\frac{1}{2}}$ and they used ℓ^{0-1}.

The above results are for $c_0 = \frac{1}{2}$. Generalisations of margin losses to the case of uneven weights are presented by Steinwart and Christmann (2008, Section 3.5). Nevertheless, since the same ϕ function is still used for both components of the loss (albeit with unequal weights) such a scheme can still not capture the full generality of all proper scoring rules in the manner achieved by the results in Section 7.1.

Appendix E. History of Pinsker Inequalities

Pinsker (1964) presented the first bound relating $\mathrm{KL}(P,Q)$ to $V(P,Q)$: $\mathrm{KL} \geq V^2/2$ and it is now known by his name or sometimes as the Pinsker-Csiszár-Kullback inequality since Csiszár (1967)

presented another version and Kullback (1967) showed $\mathrm{KL} \geq V^2/2 + V^4/36$. Much later Topsøe (2001) showed $\mathrm{KL} \geq V^2/2 + V^4/36 + V^6/270$. Non-polynomial bounds are due to Vajda (1970): $\mathrm{KL} \geq L_{\mathrm{Vajda}}(V) := \ln\left(\frac{2+V}{2-V}\right) - \frac{2V}{2+V}$ and Toussaint (1978) who showed $\mathrm{KL} \geq L_{\mathrm{Vajda}}(V) \vee (V^2/2 + V^4/36 + V^8/288)$.

Care needs to be taken when comparing results from the literature as different definitions for the divergences exist. For example Gibbs and Su (2002) use a definition of V that differs by a factor of 2 from ours. There are some isolated bounds relating V to some other divergences, analogous to the classical Pinkser bound; Kumar and Chhina (2005) have presented a summary as well as new bounds for a wide range of *symmetric* f-divergences by making assumptions on the likelihood ratio: $r \leq p(x)/q(x) \leq R < \infty$ for all $x \in \mathcal{X}$. This line of reasoning has also been developed by Dragomir et al. (2001) and Taneja (2005a,b). Topsøe (2000) has presented some infinite series representations for capacitory discrimination in terms of triangular discrimination which lead to inequalities between those two divergences. Liese and Miescke (2008, p.48) give the inequality $V \leq h\sqrt{4-h^2}$ (which seems to be originally due to LeCam, 1986) which when rearranged corresponds exactly to the bound for h^2 in theorem 31. Withers (1999) has also presented some inequalities between other (particular) pairs of divergences; his reasoning is also in terms of infinite series expansions.

Unterreiter et al. (2000) considered the case of $n = 1$ but arbitrary $\mathbb{I}_f$ (that is they bound an arbitrary f-divergence in terms of the variational divergence). Their argument is similar to the geometric proof of Theorem 30. They do not compute any of the explicit bounds in theorem 31 except they state (page 243) $\chi^2(P,Q) \geq V^2$ which is looser than (58).

Gilardoni (2006a) showed (via an intricate argument) that if $f'''(1)$ exists, then $\mathbb{I}_f \geq \frac{f''(1)V^2}{2}$. He also showed some fourth order inequalities of the form $\mathbb{I}_f \geq c_{2,f}V^2 + c_{4,f}V^4$ where the constants depend on the behaviour of f at 1 in a complex way. Gilardoni (2006b,c) presented a completely different approach which obtains many of the results of theorem 31.[27] Gilardoni (2006c) improved Vajda's bound slightly to $\mathrm{KL}(P,Q) \geq \ln\frac{2}{2-V} - \frac{2-V}{2}\ln\frac{2+V}{2}$.

Gilardoni (2006b,c) presented a general tight lower bound for $\mathbb{I}_f(P,Q)$ in terms of $V(P,Q)$ which is difficult to evaluate explicitly in general:

$$\mathbb{I}_f \geq \frac{V}{2}\left(\frac{f[g_R^{-1}(k(1/V))]}{g_R^{-1}(k(1/V))-1} + \frac{f[g_L^{-1}(k(1/V))]}{1-g_L^{-1}(k(1/V))}\right),$$

where $k^{-1}(t) = \frac{1}{2}\left(\frac{1}{1-g_L^{-1}(t)} + \frac{1}{g_R^{-1}(t)-1}\right)$, $g(u) = (u-1)f'(u) - f(u)$, $g_R^{-1}[g(u)] = u$ for $u \geq 1$ and $g_L^{-1}[g(u)] = u$ for $u \leq 1$. He presented a new parametric form for $\mathbb{I}_f = \mathrm{KL}$ in terms of Lambert's W function. In general, the result is analogous to that of Fedotov et al. (2003) in that it is in a parametric form which, if one wishes to evaluate for a particular V, one needs to do a one dimensional numerical search—as complex as (59). However, when f is such that $\mathbb{I}_f$ is symmetric, this simplifies to the elegant form $\mathbb{I}_f \geq \frac{2-V}{2}f\left(\frac{2+V}{2-V}\right) - f'(1)V$. He presented explicit special cases for h^2, J, Δ and I identical to the results in Theorem 31. It is not apparent to us how the approach of Gilardoni (2006b,c) could be extended to more general situations such as that in Theorem 30 (i.e., $n > 1$).

Finally Bolley and Villani (2005) have considered *weighted* versions of the Pinsker inequalities (bounds for a weighted generalisation of Variational divergence) in terms of KL-divergence that are related to transportation inequalities.

27. We were unaware of these two papers until completing the results presented in the main paper.

Appendix F. Variational Representation of $\mathbb{I}_f$ and its Generalizations

The variational representation of the Variational divergence (62) suggests the question of whether there is a variational representation for a general f-divergence. This has been considered previously. We briefly summarise the approach, and then explore some (new) implications of the representation.

One can obtain a variational representation for $\mathbb{I}_f$ by substituting a variational representation for f into the definition of $\mathbb{I}_f$ (Keziou, 2003a,b; Broniatowski, 2004; Broniatowski and Keziou, 2009). Let p and q denote the densities corresponding to P and Q and assume for now they exist. Recall from Section 2.2 above, that the Legendre-Fenchel conjugate of f is given by $f^\star(s) = \sup_{u\in \mathrm{Dom} f} us - f(u)$. In general $\mathrm{Ran}\, f^\star = \mathbb{R}^\star := \mathbb{R}\cup\{+\infty\}$. Since $f(u) = \sup_{\rho\in\mathbb{R}} u\rho - f^\star(\rho)$, we can write

$$\begin{aligned}
\mathbb{I}_f(P,Q) &= \int_{\mathcal{X}} q(x) \sup_{\rho\in\mathbb{R}} \left(\rho\frac{p(x)}{q(x)} - f^\star(\rho)\right) dx \\
&= \sup_{\rho\in\mathbb{R}^{\mathcal{X}}} \int_{\mathcal{X}} \rho(x)p(x) - f^\star(\rho(x))q(x)dx. \\
&= \sup_{\rho\in\mathbb{R}^{\mathcal{X}}} (\mathbb{E}_P\rho - \mathbb{E}_Q f^\star(\rho)). \qquad (83)
\end{aligned}$$

We make this concrete by considering the variational divergence. The corresponding f is given by $f(t) = |t-1|$ and (adopting the convention that $[\![\text{false}]\!]$ is a "very strong zero" so $[\![\text{false}]\!]\cdot\infty = 0$; confer Knuth, 1992)

$$f^\star(x) = [\![x \notin [-1,1]]\!]\infty + [\![x\in[-1,1]]\!]x.$$

Since the supremum in (83) will not be attained if the second term is infinite, one can restrict the supremum to be over $\mathcal{F} = \{\rho\in\mathbb{R}^{\mathcal{X}}\colon \|\rho\|_\infty \le 1\}$. Thus

$$\begin{aligned}
V(P,Q) &= \sup_{\rho\colon \|\rho\|_\infty\le 1} (\mathbb{E}_P\rho - \mathbb{E}_Q\rho) = \sup_{\rho\in\{-1,1\}^{\mathcal{X}}} (\mathbb{E}_P\rho - \mathbb{E}_Q\rho) \\
&= \sup_{\rho\in\{0,2\}^{\mathcal{X}}} (\mathbb{E}_P\rho - \mathbb{E}_Q\rho) = 2\sup_{\rho\in\{0,1\}^{\mathcal{X}}} (\mathbb{E}_P\rho - \mathbb{E}_Q\rho) \\
&= 2\sup_A |P(A) - Q(A)|,
\end{aligned}$$

since the supremum will be attained for functions ρ taking on values only in $\{-1,1\}$ and the remaining steps are simply a shift and rescaling (to $\{0,2\}$ by adding 1, and then to $\{0,1\}$).

The representation (83) suggests the generalisation

$$\begin{aligned}
\mathbb{I}_{f,\mathcal{F}}(P,Q) &:= \sup_{\rho\in\mathcal{F}\subseteq\mathbb{R}^{\mathcal{X}}} \int_{\mathcal{X}} \rho(x)p(x) - f^\star(\rho(x))q(x)dx \\
&= \sup_{\rho\in\mathcal{F}} (\mathbb{E}_P\rho - \mathbb{E}_Q f^\star(\rho)).
\end{aligned}$$

Observing this is not symmetric in p and q suggests a further generalisation:

$$\begin{aligned}
\mathbb{I}_{f,g,\mathcal{F}}(P,Q) &:= \sup_{\rho\in\mathcal{F}\subseteq\mathbb{R}^{\mathcal{X}}} \int_{\mathcal{X}} -g^\star(\rho(x))p(x) - f^\star(\rho(x))q(x)dx \\
&= \sup_{\rho\in\mathcal{F}} (-\mathbb{E}_P g^\star(\rho) - \mathbb{E}_Q f^\star(\rho)).
\end{aligned}$$

Here $g^\star$ is the $\mathbb{R}^\star$-valued LF conjugate of a convex function g. Set $\mathbb{I}_{f,g} := \mathbb{I}_{f,g,\mathbb{R}^\mathcal{X}}$.

An alternative generalisation of $\mathbb{I}_f$ is

$$\tilde{\mathbb{I}}_{f,g,\mathcal{F}}(P,Q) := \sup_{\rho\in\mathcal{F}} (\mathbb{E}_P g^\star(\rho) - \mathbb{E}_Q f^\star(\rho))$$

which is identical to (84) except for removal of the minus sign preceding $g^\star$. Set $\tilde{\mathbb{I}}_{f,g} := \tilde{\mathbb{I}}_{f,g,\mathbb{R}^\mathcal{X}}$. If $\rho \in \mathcal{F}$ are such that $\|\rho\|_\infty$ is unbounded, then in general $\tilde{I}_{f,g,\mathcal{F}}(P,Q)$ will be infinite. Properties of the alternative definition relate to the extended infimal convolution between two convex functions.

Definition 35 *Suppose $f,g\colon \mathbb{R}^+ \to \mathbb{R}^*$ are convex. The extended infimal convolution is*

$$(f\square g)(\tau) := \inf_{x\in\mathbb{R}^+} f(x) + \tau g(x/\tau), \quad \tau \in \mathbb{R}^+.$$

Note that the second term in this convolution is the perspective function (Section 2.1) applied to g, that is, $I_g(x,\tau)$.

Theorem 36 *Suppose $f,g\colon \mathbb{R}^+ \to \mathbb{R}^*$ are convex. Then*

1. $\mathbb{I}_f(P,Q) = \mathbb{I}_{f,\mathbb{R}^\mathcal{X}}(P,Q)$, $\tilde{\mathbb{I}}_{f,\mathrm{id},\mathcal{F}}(P,Q) = \mathbb{I}_{f,\mathcal{F}}(P,Q)$, *and*
$$\mathbb{I}_{t\mapsto|t-1|,\mathcal{F}}(P,Q) = 2V_{\mathcal{F},\frac{1}{2}}(P,Q).$$
2. $\tilde{\mathbb{I}}_{f_1,g_1,\mathcal{F}} = \mathbb{I}_{f_2,g_2,\mathcal{F}}$ *only if* $f_1 - f_2 = f_a$ *and* $g_1 - g_2 = g_a$ *and* $f_1, f_2, f_a, g_1, g_2, g_a$ *are affine.*
3. $\mathbb{I}_{f,f,\mathcal{F}} = \mathbb{I}_{\mathrm{id},\mathrm{id},f^\star(\mathcal{F})}(P,Q)$.
4. $\tilde{\mathbb{I}}_{f,f,\mathcal{F}} = \tilde{\mathbb{I}}_{\mathrm{id},\mathrm{id},f^\star(\mathcal{F})}(P,Q) = 2V_{f^\star(\mathcal{F})}(P,Q)$.
5. $\mathbb{I}_{f,g} = \mathbb{I}_{f\square g}$.

Proof Part 1 follows immediately from the various definitions. Since affine functions are the only functions that are simultaneously convex and concave, $\tilde{\mathbb{I}}_{f_1,g_1,\mathcal{F}} = \mathbb{I}_{f_2,g_2,\mathcal{F}}$ only if f_1, f_2 (resp. g_1, g_2) are affine and their differences are affine (since an affine offset will not change $\tilde{\mathbb{I}}$). This proves part 2.

We have by change of variables

$$\tilde{\mathbb{I}}_{f,f,\mathcal{F}}(P,Q) = \sup_{\rho\in\mathcal{F}}(\mathbb{E}_P f^\star(\rho) - \mathbb{E}_Q f^\star(\rho)) = \sup_{\psi\in f^\star(\mathcal{F})} (\mathbb{E}_P\psi - \mathbb{E}_Q\psi) = \tilde{\mathbb{I}}_{\mathrm{id},\mathrm{id},f^\star(\mathcal{F})}(P,Q),$$

where $f^\star(\mathcal{F}) := \{f^\star\circ\rho\colon \rho\in\mathcal{F}\}$. (The same argument applies to $\mathbb{I}_{f,f,\mathcal{F}}$ although $\sup_{\psi\in g^\star(\mathcal{F})}(-\mathbb{E}_P\psi - \mathbb{E}_Q\psi)$ does not correspond to a generalised variational divergence.) This proves parts 3 and 4. ∎

In order to prove 5 we need the following lemma.

Lemma 37 *Let $f\colon \mathbb{R}\to\mathbb{R}$ and $K\colon \mathbb{R}\times\mathbb{R}\to\mathbb{R}$ be convex and bounded from below. Then the extended infimal convolution*

$$(f\square K)(x) = \inf_{y\in\mathbb{R}} f(y) + K(x,y), \quad x\in\mathbb{R}$$

is convex in $x\in\mathbb{R}$.

Observe that if $K(x,y) = g(x-y)$ for convex g, then $f \square K = f \oplus g$, the standard infimal convolution (Hiriart-Urruty and Lemaréchal, 1993b). This extended infimal convolution seems little studied with the exception owith the exception of Cepedello-Boiso (1998).

Proof Let $\tilde{f}(x,y) := f(y)$, $x \in \mathbb{R}$. Clearly $\tilde{f}$ is convex on $\mathbb{R} \times \mathbb{R}$. Let $\tilde{h}(x,y) = \tilde{f}(x,y) + K(x,y)$. Hiriart-Urruty and Lemaréchal (1993b, Proposition 2.1.1) show that $\tilde{h}$ is convex on $\mathbb{R} \times \mathbb{R}$. Observe that $(f \square K)(x) = \inf\{\tilde{h}(x,y) \colon y \in \mathbb{R}\}$, that is, the *marginal* function of $\tilde{h}$. Since by construction $\tilde{h}$ is bounded from below, using the result of Hiriart-Urruty and Lemaréchal (1993b, p.169) proves the result. ■

Corollary 38 *For any convex f and g, $f \square g$ is convex.*

Proof Observe that $(f \square g)(x) = \inf_{y \in \mathbb{R}^+} f(y) + xg(y/x) = \inf_{y \in \mathbb{R}^+} f(y) + I_g(x,y)$, $x \in \mathbb{R}^+$, where I_g is the perspective function (1). Hiriart-Urruty and Lemaréchal (1993b, Proposition 2.2.1) show that if $g \colon \mathbb{R}^n \to \mathbb{R}$ is convex then the perspective I_g is convex on $\mathbb{R}^{n+1}$. The corollary then follows from the lemma. ■

Proof (part 5 of Theorem 36) Observe that if $h(x) = t\phi(x)$ then the LF conjugate $h^*(s) = t\phi(s/t)$. Thus using the Fenchel duality theorem (Rockafellar, 1970) we have, using (Rockafellar and Wets, 2004, Theorem 14.60) to justify the swapping the order of the supremum and integration,

$$\begin{aligned}
\mathbb{I}_{f,g}(P,Q) &= \sup_{\rho \in \mathbb{R}^{\mathcal{X}}} \int_{\mathcal{X}} -g^\star(\rho(x))p(x) - f^\star(\rho(x))q(x)dx \\
&= \int_{\mathcal{X}} \sup_{\rho \in \mathbb{R}} -g^\star(\rho)p(x) - f^\star(\rho)q(x)dx \\
&= \int_{\mathcal{X}} \inf_{\rho \in \mathbb{R}} f\left(\frac{\rho}{q(x)}\right) + g\left(\frac{\rho}{p(x)}\right) dx \\
&= \int_{\mathcal{X}} \inf_{\rho \in \mathbb{R}} q(x) f\left(\frac{\rho}{q(x)}\right) + p(x) g\left(\frac{\rho}{p(x)}\right) dx \\
&= \int_{\mathcal{X}} i_{f,g}(p,q)(x)dx,
\end{aligned}$$

where

$$i_{f,g}(p,q)(\cdot) := \inf_{\rho \in \mathbb{R}} q(\cdot) f\left(\frac{\rho}{q(\cdot)}\right) + p(\cdot) g\left(\frac{\rho}{p(\cdot)}\right).$$

Let $x := \frac{\rho}{q} \in \mathbb{R}^+$. Thus $\rho = xq$ and

$$i_{f,g}(p,q) = \inf_{x \in \mathbb{R}^+} qf(x) + pg(xq/p).$$

Let $\tau = \frac{p}{q} \in \mathbb{R}^+$. Thus

$$\begin{aligned}
i_{f,g}(p,q)(\tau) &= \inf_{x \in \mathbb{R}^+} qf(x) + pg(x/\tau) \\
&= q\left[\inf_{x \in \mathbb{R}^+} f(x) + \tau g(x/\tau)\right] \\
&= q \cdot (f \square g)(\tau). \qquad (84)
\end{aligned}$$

Let $h := f \square g$. Observe from (84) that $i_{f,g}(p,q) = qh(p/q)$ and thus

$$\mathbb{I}_{f,g}(p,q) = \int_{\mathcal{X}} q(x) h\left(\frac{p(x)}{q(x)}\right) dx = \mathbb{I}_h(p,q)$$

if h is convex, which we know to be the case from Corollary 38. ■

It suggests the question: given a suitable convex f, does there always exist g such that $f = g \square g$? This is analogous to the question of spectral factorisation (Sayed and Kailath, 2001) for ordinary linear convolution. We do not know the answer to this question, but have collected a few examples in Appendix G that demonstrates it is certainly true for *some* f. There does not appear to be a result analogous to part 5 of Theorem 36 for $\tilde{\mathbb{I}}_{f,g}$.

We have seen how f-divergences are related to integral probability metrics $V_{\mathcal{F}}$. It turns out that the variational divergence is special in being both. Many integral probability metrics are true metrics (Müller, 1997a,b). The only f-divergence that is a metric is the variational divergence. Whether there exist $\mathcal{F}$ such that $V_{\mathcal{F}}(\cdot,\cdot)$ is not a metric but equals $\mathbb{I}_f(\cdot,\cdot)$ for some $f \neq t \mapsto |t-1|$ (or affine transformation thereof) is left as an open problem.[28]

We end with another open problem. We have seen how $\underline{\mathbb{L}}_{\mathcal{F}}$ and $V_{\mathcal{F}}$ are related. This begs the question whether there is a representation of the form

$$\mathbb{I}_{f,\mathcal{F}}(P,Q) \stackrel{?}{=} \int_0^1 \Delta \underline{\mathbb{L}}_{\mathcal{F}}^{0-1}(\pi,P,Q)\gamma_f(\pi)d\pi.$$

Appendix G. Examples of Extended Convolution Factorisation

In this section we present three examples of f which can be written as $f = g \square g$.

If $g(t) = (t-1)^2$ (corresponding to Pearson χ^2 divergence), $(g \square g)(\tau) = \inf_{x \in \mathbb{R}^+} (x-1)^2 + \tau(x/\tau - 1)^2$. Differentiating the right-hand side with respect to x, setting to zero and solving for x gives $x = \frac{4}{2(1+1/\tau)}$. Substituting we obtain $(g \square g)(\tau) = \frac{(\tau-1)^2}{\tau-1}$ which is the f for $\Delta(P,Q)$, the triangular discrimination.

If $g(t) = t\ln(t)$, a similar straightforward calculation yields $(g \square g)(\tau) = \frac{-2\sqrt{\tau}}{e}$.

If $g(t) = (\sqrt{t}-1)^2$ (corresponding to Hellinger divergence) then a similar calculation yields $(g \square g)(\tau) = \frac{1}{2}(\sqrt{\tau}-1)^2 = g(\tau)/2$. Thus this g plays a role analogous to a gaussian kernel in ordinary convolution. The significance of this is unclear.

We summarise the results (and the associated $g^\star$) in the following table.

$g(t)$	$(g \square g)(\tau)$	$g^\star(s)$
$(t-1)^2$	$\frac{(\tau-1)^2}{\tau-1}$	$\frac{s^2}{4} + s$
$t\ln t$	$\frac{-2\sqrt{\tau}}{e}$	e^{s-1}
$(\sqrt{t}-1)^2$	$\frac{1}{2}(\sqrt{\tau}-1)^2$	$\frac{s}{1-s}[\![s<1]\!] + \infty[\![s \geq 1]\!]$

28. This has in fact been solved by Sriperumbudur et al. (2009) since an earlier version of the present paper was published as an ArXiV preprint.

Whilst it is indeed straightforward to compute $(g\Box g)$ given g (although a simple closed form is not always possible), it is far from obvious how to go from a given f to a g such that $f = g\Box g$.

Hiriart-Urruty and Lemaréchal (1993a, page 69) show that for f convex on $\mathbb{R}^+$, g convex and increasing on $\mathbb{R}^+$,

$$(g \circ f)^\star(s) = \inf_{\alpha>0} \alpha f^\star(\tfrac{s}{\alpha}) + g^\star(\alpha) = f^\star \Box g^\star.$$

This illuminates the difficulty of the above "factorisation problem". It is equivalent to: given a convex increasing $f^\star$, find a convex increasing $g^\star$ such that $f^\star = g^\star \circ g^\star$.

Appendix H. Empirical Estimators of $V_{B_{\mathcal{H}},\frac{1}{2}}(P,Q)$ and SVMs

This appendix further develops the observations made in Section 8.2 regarding the relationship between divergence and risk when $\mathcal{R} = B_{\mathcal{H}}$, a unit ball in a reproducing kernel Hilbert space $\mathcal{H}$. In contrast to the rest of the paper (which focussed on relationships involving the underlying distributions), in this appendix we will consider the practical situation where there is only an empirical sample. We will see how the general results have interesting implications for sample based machine learning algorithms.

If we require an empirical estimate of $V_{\mathcal{R},\pi}(P,Q)$ we can replace P and Q by empirical distributions. We will use *weighted* empirical distributions. Given an independent identically distributed sample $\mathbf{w} = (w_1,\ldots,w_m) \in \mathcal{X}^m$ the $\boldsymbol{\alpha}$-weighted empirical distribution $\hat{P}_{\mathbf{w}}^{\boldsymbol{\alpha}}$ with respect to $\mathbf{w}$ is defined by

$$d\hat{P}_{\mathbf{w}}^{\boldsymbol{\alpha}} := \sum_{i=1}^m \alpha_i \delta(\cdot - w_i)$$

where $\boldsymbol{\alpha} = (\alpha_1,\ldots,\alpha_m)$, $\alpha_i \geq 0$, $i = 1,\ldots,m$ and $\sum_{i=1}^m \alpha_i = 1$. We will write $\hat{\mathbb{E}}_{\mathbf{w}}^{\boldsymbol{\alpha}}\phi := \mathbb{E}_{\hat{P}_{\mathbf{w}}^{\boldsymbol{\alpha}}}\phi = \sum_{i=1}^m \alpha_i \phi(w_i)$. Thus

$$V^2_{\mathcal{R},\frac{1}{2}}(\hat{P}_{\mathbf{w}}^{\boldsymbol{\alpha}}, \hat{P}_{\mathbf{z}}^{\boldsymbol{\beta}}) = \frac{1}{2}\|\hat{\mathbb{E}}_{\mathbf{w}}^{\boldsymbol{\alpha}}\phi - \hat{\mathbb{E}}_{\mathbf{z}}^{\boldsymbol{\beta}}\|^2_{\mathcal{H}}.$$

Suppose now that P and Q correspond to the positive and negative class conditional distributions. Let $\mathbf{x} := (x_1,\ldots,x_m)$ be a sample drawn from $M = \pi P + (1-\pi)Q$ with corresponding label vector $\mathbf{y} = (y_1,\ldots,y_m)$. Let $I := \{1,\ldots,m\}$, $I^+ := \{i \in I\colon y_i = 1\}$, $I^- := \{i \in I\colon y_i = -1\}$. Consider a weight vector $\boldsymbol{\alpha} = (\alpha_1,\ldots,\alpha_m)$ over the whole sample. Thus

$$\hat{\mathbb{E}}_P\phi = \sum_{i\in I^+} \alpha_i \phi(x_i) \quad \text{and} \quad \hat{\mathbb{E}}_Q\phi = \sum_{i\in I^-} \alpha_i \phi(x_i)$$

where we also require

$$\sum_{i\in I^+} \alpha_i = \frac{m^+}{m} \quad \text{and} \quad \sum_{i\in I^-} \alpha_i = \frac{m^-}{m}$$

and hence

$$\sum_{i\in I} \alpha_i y_i = \frac{m^+ - m^-}{m}.$$

Substituting into (65) we have

$$\begin{aligned}
2V_{B_{\mathcal{H}},\frac{1}{2}}(\hat{P},\hat{Q}) &= \left\langle \hat{\mathbb{E}}_P\phi - \hat{\mathbb{E}}_Q\phi, \hat{\mathbb{E}}_P\phi - \hat{\mathbb{E}}_Q\phi \right\rangle \\
&= \left\langle \sum_{i\in I^+} \alpha_i\phi(x_i) - \sum_{i\in I^-} \alpha_i\phi(x_i), \sum_{j\in I^+} \alpha_j\phi(x_j) - \sum_{j\in I^-} \alpha_j \right\rangle \\
&= \left\langle \sum_{i\in I} \alpha_i y_i\phi(x_i), \sum_{j\in I} \alpha_j y_j\phi(x_j) \right\rangle \\
&= \sum_{i\in I}\sum_{j\in I} \alpha_i\alpha_j y_i y_j \langle \phi(x_i), \phi(x_j)\rangle \\
&= \sum_{i\in I}\sum_{j\in I} \alpha_i\alpha_j y_i y_j k(x_i,x_j) =: J(\boldsymbol{\alpha},\mathbf{x}). \qquad (85)
\end{aligned}$$

We now consider three different choices of $\boldsymbol{\alpha}$.

Uniform weighting If we set $\alpha_i = \frac{1}{m}$, $i = 1,\ldots,m$, then (85) becomes

$$\frac{1}{m^2}\sum_{i,j\in I} y_i y_j k(x_i,x_j) = \mathrm{MMD}_b^2[B_{\mathcal{H}}, \boldsymbol{x}^+, \boldsymbol{x}^-]$$

where $\boldsymbol{x}^+ := (x_i)_{i\in I^+}$, $\boldsymbol{x}^- := (x_i)_{i\in I^-}$ and MMD_b is the biased estimator of the *Maximum Mean Discrepancy* (Gretton et al., 2008), an alternate name for $V_{\mathcal{R}}$. Observe that from theorem 34, this case corresponds to using a Fisher linear discriminant in feature space (Devroye et al., 1996) when it is assumed that the within-class covariance matrices are both the identity matrix. This follows by observing that the constructed hypothesis is identical in both cases.

Pessimistic Weighting Instead of weighting each sample equally, one can optimise over $\boldsymbol{\alpha}$. By theorem 34, minimizing $J(\boldsymbol{\alpha},\mathbf{x})$ over $\boldsymbol{\alpha}$ will maximize $\underline{\mathbb{L}}^{\mathrm{lin}}$ and is thus the most pessimistic choice. Explicitly, we have

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^m\sum_{i=1}^m \alpha_i\alpha_j y_i y_j k(x_i,x_j) && (86)\\
\text{s.t.} \quad & \alpha_i \geq 0, \;\; i = 1,\ldots,m && (87)\\
& \sum_{i=1}^m \alpha_i y_i = \frac{m^+ - m^-}{m} && (88)\\
& \sum_{i=1}^m \alpha_i = 1 && (89)
\end{aligned}$$

which can be recognized as the support vector machine (Cortes and Vapnik, 1995). The SVM uses the sign of the "witness" (Gretton et al., 2008), $x \mapsto \sum_{i=1}^m \alpha_i y_i k(x_i,x)$ as its predictor.

Interpolation between above two cases A parameterized interpolation between the above two cases can be constructed by the addition of the constraints

$$\alpha_i \leq \frac{1}{\nu m}, \;\; i = 1,\ldots,m, \qquad (90)$$

where $\nu \in (0,1]$ is an adjustable parameter. Observe that ν controls the sparsity of $\boldsymbol{\alpha}$ since (90), (87) and (89) together imply that $|\{i \in I\colon \alpha_i \neq 0\}| \geq \nu m$. Crisp and Burges (2000) have shown that (86),...,(90) is equivalent to the ν-SVM algorithm (Schölkopf et al., 2000).

While "information-theoretic" approaches to the SVM and weighted kernel representations are hardly new,[29] the results presented here are novel and provide a simple and direct derivation of the SVM via the generalised variational divergence.

If $V_{B_{\mathcal{H}},\frac{1}{2}}(\hat{P}_{\boldsymbol{w}},\hat{Q}_{\boldsymbol{z}})$ is used as a test statistic to infer whether two samples $\boldsymbol{w}$ and $\boldsymbol{z}$ are drawn from the same distribution (as Gretton et al., 2008 do), then when the distributions from which $\boldsymbol{w}$ and $\boldsymbol{z}$ are drawn are close, the classification performance of the corresponding classifier (i.e., the classifier that uses the sign of the witness function) will be close to the worst possible. Thus one will be operating in a regime distinct from the normal situation, where the risk is typically small.

Finally observe that the derivation of the SVM presented here could be viewed as an application of an alternate "inductive principle"—a general recipe for constructing learning algorithms from learning task specification (Vapnik, 1989, 2006). The traditional Empirical Risk Minimization principle entails replacing (P,Q) with $(\hat{P}_{\boldsymbol{x}^+},\hat{Q}_{\boldsymbol{x}^-})$ in the definition of $\underline{\mathbb{L}}(\pi,P,Q)$. Then, in order to not overfit, one restricts the class of functions from which hypotheses are drawn. That is, there are two approximations:

$$\underline{\mathbb{L}}(\pi,P,Q) \xrightarrow{\text{Empirical Approximation (uniform)}} \underline{\mathbb{L}}(\pi,\hat{P}_{\boldsymbol{x}^+},\hat{Q}_{\boldsymbol{x}^-}) \xrightarrow{\text{Restrict Class}} \underline{\mathbb{L}}_{\mathcal{R}}(\pi,\hat{P}_{\boldsymbol{x}^+},\hat{Q}_{\boldsymbol{x}^-}).$$

Upon setting $\boldsymbol{\alpha}^+ = (\alpha_i)_{i\in I^+}$ and $\boldsymbol{\alpha}^- = (\alpha_i)_{i\in I^-}$, the derivation presented above, in contrast, can be summarised schematically by

$$\text{"}\underline{\mathbb{L}}(\pi,P,Q)\text{"} \xrightarrow{\text{Restrict Class}} \underline{\mathbb{L}}_{\mathcal{R}}(\pi,P,Q) \xrightarrow{\text{Empirical Approximation } (\boldsymbol{\alpha}\text{-weighted})} \underline{\mathbb{L}}_{\mathcal{R}}(\pi,\hat{P}^{\boldsymbol{\alpha}^+}_{\boldsymbol{x}^+},\hat{Q}^{\boldsymbol{\alpha}^-}_{\boldsymbol{x}^-}),$$

where a different loss (the "linear" loss) was used at the start. With that loss function, reversing the order of the two approximations would not work, and is (thus) not equivalent to the ERM inductive principle. The first step makes $\underline{\mathbb{L}}$ well defined—with no restriction it is not, hence the quotes; and will avoid overfitting in any case. The second step is the more general ($\boldsymbol{\alpha}$-weighted) empirical approximation.

We believe that this alternate derivation of the SVM is of interest because it is simpler (avoids the need to introduces margins) and it elucidates the connection between the kernel methods for

29. The use of kernel representations for classification is of course not new: from the classical kernel classifier (where $\alpha_i = 1/m$ for all $i \in I$) (Devroye et al., 1996, Chapter 10) to the Generalised Portrait (Aizerman et al., 1964), the Generalised Discriminant (Baudat and Anouar, 2000) and the panoply of techniques inspired by Support Vector Machines (Schölkopf and Smola, 2002; Herbrich, 2002). None of these techniques is designed from the perspective of minimising a f-divergence.

Principe et al. (2000a) have developed an approach to machine learning problems based on information theoretic criteria (Principe et al., 2000b; Jenssen et al., 2004; Xu et al., 2005; Jenssen, 2005a; Jenssen et al., 2006; Pavia et al., 2006). Jenssen et al. (2004, 2006) considered kernel methods from the perspective of Renyi's quadratic entropy. They do not exploit the formal relationship between maximising divergence and minimising risk. They interpret the SVM as being constructed from weighted Parzen windows density estimates. Gretton et al. (2008) explained the relationship between their MMD estimators and those derived from (unweighted) Parzen windows estimates of the class-conditional distributions. Weighted Parzen windows estimates were used as a basis for building a classifier by Babich and Camps (1996). Weighted empirical distributions are widely used in particle filtering (Crisan and Doucet, 2002).

McDermott and Katagiri (2002) considered the direct optimisation of a classifier built on top of Parzen windows density estimates. They showed that the minimum classification error criterion is equivalent to a Parzen windows estimate of the theoretical Bayes risk. They re-derive the traditional approach of minimising an estimate of the expected loss. McDermott and Katagiri (2003) extended their approach to the multi-class setting in a way that takes account of all the "other" classes better in estimating the probability of error of a given class.

classification and MMD—indeed MMD is nothing but the Fisher linear discriminant applied to a binary problem induced by the given distributions P and Q.

References

H. Abelson, G.J. Sussman, and J. Sussman. *Structure and Interpretation of Computer Programs*. MIT Press, 1996.

J. Aczél. Measuring information beyond communication theory — why some generalized information measures maybe be useful, others not. *Aequationes Mathematicae*, 27:1–19, 1984.

M. A. Aizerman, É. M. Braverman, and L. I. Rozonoér. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

S.M. Ali and S.D. Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society. Series B (Methodological)*, 28(1):131–142, 1966.

Y. Altun and A. Smola. Unifying divergence minimization and statistical inference via convex duality. In *Proceedings of the Annual Conference on Learning Theory (COLT)*, pages 139–153, 2006.

P. Antosik, J. Mikusinski, and R. Sikorski. *Theory of Distributions: The Sequential Approach*. American Elsevier, 1973.

W. R. Ashby. *An Introduction to Cybernetics*. Chapman and Hall, 1956.

G.A. Babich and O.I. Camps. Weighted Parzen windows for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):567–570, 1996.

F.R. Bach, D. Heckerman, and E. Horvitz. Considering cost asymmetry in learning classifiers. *Journal of Machine Learning Research*, 7:1713–1741, 2006.

C.Y. Baldwin and K.B. Clark. Modularity in the design of complex engineering systems. In Dan Braha Ali Minai and Yaneer Bar Yam, editors, *Complex Engineered Systems: Science Meets Technology*. Springer, 2006a.

C.Y. Baldwin and K.B. Clark. Between 'knowledge' and 'the economy': Notes on the scientific study of designs. In D. Forey and B. Kahin, editors, *Advancing Knowledge and the Knowledge Economy*, pages 299–328, Cambridge, Mass., 2006b. MIT Press.

A. Banerjee, X. Guo, and H. Wang. On the optimality of conditional expectation as a Bregman predictor. *IEEE Transactions on Information Theory*, 51(7):2664–2669, 2005a.

A. Banerjee, S. Merugu, I.S. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *The Journal of Machine Learning Research*, 6:1705–1749, 2005b.

V. Barnett. *Comparative Statistical Inference*. John Wiley and Sons, Chichester, 3rd edition, 1999.

P.L. Bartlett, P.M. Long, and R.C. Williamson. Fat-shattering and the learnability of real-valued functions. *Journal of Computer and System Sciences*, 52(3):434–452, 1996.

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, March 2006.

G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

H. H. Bauschke, R. Goebel, Y. Lucet, and X. Wang. The proximal average: Basic theory. *SIAM Journal on Optimization*, 19:766–785, 2008.

M.J. Bayarri and J.O. Berger. The interplay of Bayesian and frequentist analysis. *Statistical Science*, 19(1):58–80, 2004.

M. Ben-Bassat. epsilon-equivalence of feature selection rules. *IEEE Transactions on Information Theory*, 24(6):769–772, 1978.

S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman-Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, 2010.

J.O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer, New York, 1985.

J.O. Berger. Could Fisher, Jeffreys and Neyman have agreed on testing? *Statistical Science*, 18(1): 1–32, 2003.

I. Berlin. *The Hedgehog and the Fox: An Essay on Tolstoy's view of History*. Weidenfeld and Nicolson, London, 1953.

A. Beygelzimer, V. Dani, T. Hayes, J. Langford, and B. Zadrozny. Error limiting reductions between classification tasks. In *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, 2005.

A. Beygelzimer, J. Langford, and P. Ravikumar. Multiclass classification with filter trees. Preprint, June 2007. URL `http://hunch.net/~jl/projects/reductions/mc_to_b/invertedTree.pdf`.

A. Beygelzimer, J. Langford, and B. Zadrozny. Machine learning techniques — reductions between prediction quality metrics. In Zhen Liu and Cathy H. Xia, editors, *Performance Modeling and Engineering*, pages 3–28. Springer US, April 2008. URL `http://hunch.net/~jl/projects/reductions/tutorial/paper/chapter.pdf`.

P. Bickel and K. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*, volume 1. Prentice-Hall, 2nd edition, 2001.

A. Birnbaum. On the foundations of statistical inference: Binary experiments. *The Annals of Mathematical Statistics*, 32(2):414–435, June 1961.

D. Blackwell. Comparison of experiments. In J. Neyman, editor, *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*, pages 93–102, Berkeley and Los Angeles, 31 July – 12 August 1951. University of California Press.

D. Blackwell. Equivalent comparisons of experiments. *The Annals of Mathematical Statistics*, 24 (2):265–272, 1953.

A. Blass and Y. Gurevich. Algorithms: A quest for absolute definitions. *Bulletin of the European Association for Theoretical Computer Science*, 81:195–225, 2003.

A. Blass, N. Dershowitz, and Y. Gurevich. When are two algorithms the same? *Bull. Symbolic Logic*, 15(2):145–168, 2009.

F. Bolley and C. Villani. Weighted Csiszár-Kullback-Pinsker inequalities and applications to transportation inequalities. *Annales de la Faculte des Sciences de Toulouse*, 14(3):331–352, 2005.

K.M. Borgwardt, A. Gretton, M.J. Rasch, H.P. Kriegel, B. Schölkopf, and A.J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14), 2006.

J.M. Borwein and A.S. Lewis. Duality relationships for entropy-like minimization problems. *SIAM Journal of Control and Optimization*, 29(2):325–338, 1991.

O. Bousquet. Making machine learning more scientific. Blog Post., 2006. URL `http://ml.typepad.com/machine_learning_thoughts/2006/06/making_machine_.html`.

G.C. Bowker. *Memory practices in the sciences*. MIT Press, 2005.

G.C. Bowker and S.L. Star. *Sorting Things Out: Classification and its Consequences*. MIT Press, Cambridge, Mass., 1999.

L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

M. Broniatowski. Minimum divergence in inference and testing. In *Atti della XLII Riunione Scientifica*, Università di Bari, 2004. Società Italiana Statistica.

M. Broniatowski and A. Keziou. Parametric estimation and tests through divergences and the duality technique. *Journal of Multivariate Analysis*, 100(1):16–36, 2009.

L.D. Brown and M.G. Low. Asymptotic equivalence of nonparametric regression and white noise. *Annals of Statistics*, 24(6):2384–2398, 1996.

L.D. Brown and L. Zhao. Direct asymptotic equivalence of nonparametric regression and the infinite dimensional location problem. Preprint, 2003. URL `http://www-stat.wharton.upenn.edu/~lzhao/papers/`.

L.D. Brown, T.T. Cai, M.G. Low, and C.H. Zhang. Asymptotic equivalence theory for nonparametric regression with random design. *Annals of Statistics*, 30(3):688–707, 2002.

H.D. Brunk, G.M. Ewing, and W.R. Utz. Minimizing integrals in certain classes of monotone functions. *Pacific Journal of Mathematics*, 7(1):833–847, 1957.

A. Buja, W. Stuetzle, and Y. Shen. Loss functions for binary class probability estimation and classification: Structure and applications. Technical report, University of Pennsylvania, November 2005.

S. Canu and A. Smola. Kernel methods and the exponential family. *Neurocomputing*, 69(7-9): 714–720, 2006.

V.J. Carey, J. Mar, and R. Gentleman. MLInterfaces: Towards uniform behaviour of machine learning tools in R. Technical report, Harvard University, 2007.

A.V. Carter. Deficiency distance between multinomial and multivariate normal experiments. *Annals of Statistics*, 30(3):708–730, 2002.

Y. Censor and S.A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.

M. Cepedello-Boiso. On regularization in superreflexive Banach spaces by infimal convolution formulas. *Studia Mathematica*, 129(3):265–284, 1998.

P. Chaudhuri and W.Y. Loh. Nonparametric estimation of conditional quantiles using quantile regression trees. *Bernoulli*, 8(5):561–576, 2002.

G. Choquet. Theory of capacities. *Annales de l'institut Fourier*, 5(54):131–295, 1953.

W.J. Conover and R.L. Iman. Rank transformations as a bridge between parametric and nonparametric statistics. *The American Statistician*, 35(3):124–129, 1981.

C. Cortes and M. Mohri. AUC optimization vs. error rate minimization. *Advances in Neural Information Processing Systems*, 16, 2004.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

D. Cossock and T. Zhang. Subset ranking using regression. In G. Lugosi and H.U. Simon, editors, *Proceedings of Conference on Learning Theory (COLT) 2006*, LNAI 4005, pages 605–619, 2006.

D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing,*, 50(3):736–746, 2002.

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.

I. Csiszár. Information-type measures of difference of probability distributions and indirect observations. *Studia Scientiarum Mathematicarum Hungarica*, 2:299–318, 1967.

I. Csiszár. Information measures: A critical survey. In *Transactions of the Seventh Prague Conference on Information Theory, Statistical Decision Functions, Random Processes*, pages 73–86, Dordrecht, 1978. D. Riedel.

I. Csiszár. Generalized projections for non-negative functions. *Acta Mathematica Hungarica*, 68 (1):161–186, March 1995.

A. Cuevas and R. Fraiman. A plug-in approach to support estimation. *The Annals of Statistics*, 25 (6):2300–2312, 1997.

A.P. Dawid. The geometry of proper scoring rules. *Annals of the Institute of Statistical Mathematics*, 59(1):77–93, March 2007.

M.H. DeGroot. Uncertainty, information, and sequential experiments. *The Annals of Mathematical Statistics*, 33(2):404–419, 1962.

M.H. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill Book Company, New York, 1970.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Approach to Pattern Recognition*. Springer, New York, 1996.

P. Domingos. MetaCost: a general method for making classifiers cost-sensitive. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 155–164, New York, NY, USA, 1999. ACM Press.

P. Domingos and M. Richardson. Markov logic: A unifying framework for statistical relational learning. In *Proceedings of the ICML-2004 Workshop on Statistical Relational Learning and its Connections to Other Fields*, pages 49–54, 2004.

S.S. Dragomir, V. Gluščević, and C.E.M. Pearce. Csiszár f-divergence, Ostrowski's inequality and mutual information. *Nonlinear Analysis*, 47:2375–2386, 2001.

C. Drummond. Discriminative vs. generative classifiers for cost sensitive learning. In *Proceedings of the Nineteenth Canadian Conference on Artificial Intelligence*, LNAI 4013, pages 479–490, Quebec, Canada, 2006.

C. Drummond and R.C. Holte. Cost curves: An improved method for visualizing classifier performance. *Machine Learning*, 65(1):95–130, 2006.

R. M. Dudley. *Real Analysis and Probability*. Cambridge University Press, 2nd edition, 2003.

S. Eguchi. Information geometry and statistical pattern recognition. To appear in *Sugaku Exposition*, American Mathematical Society, 2005.

S. Eguchi and J. Copas. Recent developments in discriminant analysis from an information geometric point of view. *Journal of the Korean Statistical Society*, 30:247–264, 2001.

T. Fawcett. ROC graphs: Notes and practical considerations for researchers. *Machine Learning*, 31, 2004.

T. Fawcett. ROC graphs with instance-varying costs. *Pattern Recognition Letters*, 27(8):882–891, 2006.

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge discovery and data mining: Towards a unifying framework. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 82–88. AAAI Press, 1996.

A.A. Fedotov, P. Harremoës, and F. Topsøe. Refinements of Pinsker's inequality. *IEEE Transactions on Information Theory*, 49(6):1491–1498, June 2003.

D. Feldman and F. Österreicher. A note on f-divergences. *Studia Scientiarum Mathematicarum Hungarica*, 24:191–200, 1989.

P.A. Flach. The geometry of ROC space: understanding machine learning metrics through ROC isometrics. In *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*, pages 194–201, 2003.

P.A. Flach and S. Wu. Repairing concavities in ROC curves. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI-2005)*, page 702, 2005.

L. Floridi. Open problems in the philosophy of information. *Metaphilosophy*, 35(4):554–582, 2004.

F.G. Friedlander. *Introduction to the Theory of Distributions*. Cambridge University Press, 1982.

M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. WH Freeman & Co. New York, NY, USA, 1979.

J. K. Gershenson, G. J. Prasad, and Y. Zhang. Product modularity: definitions and benefits. *Journal of Engineering Design*, 14:295–313, 2003.

M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL specification, 1998. AIPS-98 Planning Committee.

A.L. Gibbs and F.E. Su. On choosing and bounding probability metrics. *International Statistical Review*, 70:419–435, 2002.

G. L. Gilardoni. On Pinsker's type inequalities and Csiszár's f-divergences. part i: Second and fourth-order inequalities. arXiv:cs/0603097v2, April 2006a.

G.L. Gilardoni. On the minimum f-divergence for a given total variation. *Comptes Rendus Académie des sciences, Paris, Series 1*, 343, 2006b.

G.L. Gilardoni. On the relationship between symmetric f-divergence and total variation and an improved Vajda's inequality. Preprint, Departamento de Estatística, Universidade de Brasília, April 2006c. URL `http://www.unb.br/ie/est/docentes/curriculo/gustavo_homepage/artigos/improved.pdf`.

T. Gneiting and A.E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, March 2007.

P.K. Goel and M.H. DeGroot. Comparison of Experiments and Information Measures. *The Annals of Statistics*, 7(5):1066–1077, 1979.

P.K. Goel and J. Ginebra. When is one experiment 'always better than' another? *Journal of the Royal Statistical Society Series D (The Statistician)*, 52(4):515–537, 2003.

P.W. Goldberg. When can two unsupervised learners achieve PAC separation? *Proceedings of the 14th Annual Conference on Computational Learning Theory (COLT)*, pages 303–319, 2001.

S.A. Goldman, R.L. Rivest, and R.E. Schapire. Learning binary relations and total orders. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 46–51, 1989.

J.D.J. Golic. On the relationship between the information measures and the Bayes probability of error. *IEEE Transactions on Information Theory*, 33(5):681–693, 1987.

A. Gretton, K.M. Borgwardt, M. Rasch, B. Schölkopf, and A.J. Smola. A kernel method for the two-sample-problem. *Journal of Machine Learning Research*, 2008. submitted.

R.L. Grossman, M.F. Hornick, and G. Meyer. Data mining standards initiatives. *Commun. ACM*, 45(8):59–61, 2002. ISSN 0001-0782.

P.D. Grünwald. *The Minimum Description Length Principle*. MIT Press, Cambridge, MA, 2007.

P.D. Grünwald and A.P. Dawid. Game theory, maximum entropy, minimum discrepancy and robust bayesian decision theory. *The Annals of Statistics*, 32(4):1367–1433, 2004.

C. Gutenbrunner. On applications of the representation of f-divergences as averaged minimal Bayesian risk. In *Transactions of the 11th Prague Conference on Information Theory, Statistical Decision Functions and Random Processes*, pages 449–456, Dordrecht; Boston, 1990. Kluwer Academic Publishers.

D.J. Hand. Deconstructing statistical questions. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 157(3):317–356, 1994.

D.J. Hand. Comment on "The skill-plot: A graphical technique for evaluating continuous diagnostic tests". *Biometrics*, 63:259, 2008.

D.J. Hand and R.J. Till. A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine Learning*, 45(2):171–186, 2001.

D.J. Hand and V. Vinciotti. Local versus global models for classification problems: Fitting models where it matters. *The American Statistician*, 57(2):124–131, 2003.

J.A. Hanley and B.J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36, 1982.

P. Harremoës. *Time and Conditional Independence*. Ph.d. thesis, Roskilde University, 1993. Original in Danish entitled Tid og Betinget Uafhængighed. English translation partially available at `http://www.math.ku.dk/~moes/index.html`.

R. Herbrich. *Learning Kernel Classifiers*. MIT Press, Cambridge MA, 2002.

H. Heyer. *Theory of Statistical Experiments*. Springer, 1982.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms: Part II: Advanced Theory and Bundle Methods*. Springer, Berlin, 1993a.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms: Part I: Fundamentals*. Springer, Berlin, 1993b.

J-B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer, Berlin, 2001.

J.-B. Hiriart-Urruty and J.-E. Martínez-Legaz. Convex solutions of a functional equation arising in information theory. *Journal of Mathematical Analysis and Applications*, 328:1309–1320, 2007.

W. James. *A Pluralistic Universe*. Longmans, Green, and Co., London, 1909. Hibbert Lectures at Manchester College on the Present Situation in Philosophy.

R. Jenssen. An information theoretic approach to machine learning. Doctor Scientiarum Thesis, Department of Physics, Faculty of Science, University of Tromsø, 2005a.

R. Jenssen. *An Information Theoretic Approach to Machine Learning*. PhD thesis, Department of Physics, University of Tromsø, Norway, May 2005b.

R. Jenssen, D. Erdogmus, J.C. Principe, and T. Eltoft. Towards a unification of information theoretic learning and kernel methods. In *Proc. IEEE Workshop on Machine Learning for Signal Processing (MLSP2004)*, pages 93–102, 2004.

R. Jenssen, D. Erdogmus, J.C. Príncipe, and T. Eltoft. Some equivalences between kernel methods and information theoretic methods. *Journal of VLSI Signal Processing*, 45(1):49–65, 2006. (Paper 3 of Jenssen (2005a)).

D.S. Johnson. NP-Completeness Columns. *Journal of Algorithms; ACM Transactions on Algorithms*, 2–13; 1–3, 1982–1992; 2005–2007. URL `http://www.research.att.com/~dsj/columns/`.

M.I. Jordan. *Learning in Graphical Models*. MIT Press, 1999.

T. Kailath. The divergence and Bhattacharyya distance measures in signal selection. *IEEE Transactions on Communications*, 15(1):52–60, 1967.

A. Keziou. Dual representations of ϕ-divergences and applications. *Comptes Rendus Académie des sciences, Paris, Series 1*, 336:857–862, 2003a.

A. Keziou. *Utilisation des Divergences entre Mesures en Statistique Inférentielle*. PhD thesis, Université Paris 6, 2003b.

M. Khosravifard, D. Fooladivanda, and T.A. Gulliver. Confliction of the convexity and metric properties in f-divergences. *IEICE Transactions on Fundamentals of Electronics, Communication and Computer Sciences*, E90-A(9):1848–1853, 2007.

J. Kiefer. The foundations of statistics — are there any? *Synthese*, 36:161–176, 1977.

J.C. Kiefer. *Introduction to Statistical Inference*. Springer-Verlag, New York, 1987.

D.E. Knuth. Two notes on notation. *American Mathematical Monthly*, pages 403–422, 1992.

F.R. Kschischang, B.J. Frey, and H.A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

S. Kullback. Lower bound for discrimination information in terms of variation. *IEEE Transactions on Information Theory*, 13:126–127, 1967. Correction, volume 16, p. 652, September 1970.

P. Kumar and S. Chhina. A symmetric information divergence measure of the Csiszár's f-divergence class and its bounds. *Computers and Mathematics with Applications*, 49:575–588, 2005.

N. Lambert, J. Langford, J. Wortman, Y. Chen, D. Reeves, Y. Shoham, and D. Pennock. Self-financed wagering mechanisms for forecasting. In Lance Fortnow, John Riedl, and Tuomas Sandholm, editors, *Proceedings of the ACM Conference on Electronic Commerce*, pages 170–179, 2008.

J. Langford. Machine learning reductions tutorial. Slides presented at the *Machine Learning Summer School*, July 2006.

J. Langford. Alternative machine learning reductions definitions. Machine Learning (Theory), March 2007. URL `http://hunch.net/?p=255`.

J. Langford and A. Beygelzimer. Sensitive error correcting output codes. In *Proceedings of the 18th Annual Conference on Learning Theory (COLT-2005)*, pages 158–172. Springer, 2005.

J. Langford and B. Zadrozny. Estimating class membership probabilities using classifier learners. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTAT'05)*, 2005.

J. Langford, R. Oliveira, and B. Zadrozny. Predicting conditional quantiles via reduction to classification. In *Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence*. AUAI Press, 2006.

J.A. Lasserre, C.M. Bishop, and T.P. Minka. Principled hybrids of generative and discriminative models. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition – Volume 1*, pages 87–94. IEEE Computer Society Washington, DC, USA, 2006.

L. LeCam. Sufficiency and Approximate Sufficiency. *The Annals of Mathematical Statistics*, 35(4): 1419–1455, 1964.

L. LeCam. *Asymptotic Methods in Statistical Decision Theory*. Springer, 1986.

L. Li and H.-T. Lin. Ordinal regression by extended binary classification. In *Advances in Neural Information Processing Systems 19*, pages 865–873, 2007.

F. Liese and K.-J. Miescke. *Statistical Decision Theory*. Springer, New York, 2008.

F. Liese and I. Vajda. On divergences and informations in statistics and information theory. *IEEE Transactions on Information Theory*, 52(10):4394–4412, 2006.

D.V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

J. Lindström. On the origin and early history of functional analysis. Technical Report U.U.D.M. Projrect Report 2008:1, Uppsala Universitet, January 2008.

J. Locke. *An Essay Concerning Human Understanding*. Thomas Basset, London, 1690.

P.M. Long and R.A. Servedio. Discriminative learning can succeed where generative learning fails. In G. Lugosi and H.U. Simon, editors, *Proceedings of the Conference on Learning Theory (COLT)*, LNAI 4005, pages 319–334, Berlin, 2006. Springer-Verlag.

P.M. Long, R.A. Servedio, and H.U. Simon. Discriminative learning can succeed where generative learning fails. Preprint, correction to Long and Servedio (2006), November 2006.

R. Lutz and M. Musio. An alternative mathematical foundation for statistics. *Acta Applicandae Mathematicae: An International Survey Journal on Applying Mathematics and Mathematical Applications*, 89(1):217–249, 2005.

C. Lyell. *Principles of Geology*. John Murray, London, 1830.

R. A. Maxion and R. R. Roberts. Proper use of ROC curves in intrusion/anomaly detection. Technical Report CS-TR-871, School of Computing Science, University of Newcastle upon Tyne, November 2004.

E. McDermott and S. Katagiri. A Parzen window based derivation of minimum classification error from the theoretical Bayes classification risk. In *Proceedings of International Conference on Spoken Language Processing*, pages 2465–2468, 2002.

E. McDermott and S. Katagiri. A new formalization of minimum classification error using a Parzen estimate of classification chance. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 2003. Paper 4406.

T. Minka. Discriminative models, not discriminative training. Technical Report MSR-TR-2005-144, Microsoft Research, Cambridge, October 2005.

J. Mokyr. *The Lever of Riches: Technological Creativity and Economic Progress*. Oxford University Press, USA, 1992.

N. Morse and R. Sacksteder. Statistical isomorphism. *The Annals of Mathematical Statistics*, 37(1): 203–214, 1966.

A. Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29:429–443, 1997a.

A. Müller. Stochastic orders generated by integrals: A unified study. *Advances in Applied Probability*, 29:414–428, 1997b.

E. Nelson. *Radically Elementary Probability Theory*. Princeton University Press, 1987.

J. Neyman and E. S. Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Math. or Phys. Character (1896-1934)*, 231:289–337, 1933. URL `http://dx.doi.org/10.1098/rsta.1933.0009`.

A. Y. Ng and M.I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naïve Bayes. In *Advances in Neural Information Processing Systems (NIPS)*, volume 15, 2002.

X. Nguyen, M.J. Wainwright, and M.I. Jordan. On distance measures, surrogate loss functions, and distributed detection. Technical Report 695, Department of Statistics, University of California, Berkeley, October 2005.

X. Nguyen, M.J. Wainwright, and M.I. Jordan. On surrogate loss functions and f-divergences. *Annals of Statistics*, 37:876–904, 2009.

D. Noble. *The Music of Life: Biology Beyond the Genome*. Oxford University Press, 2006.

F. Österreicher. f-divergences — representation theorem and metrizability. Technical Report, Institute of Mathematics, University of Salzburg, September 2003.

F. Österreicher and D. Feldman. Divergenzen von Wahrscheinlichkeitsverteilungen — Integralgeometrisch Betrachtet. *Acta Mathematica Academaiae Scientiarum Hungarica*, 37(4):329–337, 1981.

F. Österreicher and I. Vajda. Statistical information and discrimination. *IEEE Transactions on Information Theory*, 39(3):1036–1039, 1993.

N. Palmer and P.W. Goldberg. PAC classification based on PAC estimates of label class distributions. *Arxiv preprint cs.LG/0607047*, 2006.

A.R.C. Pavia, J.-.W Xu, and J.C. Príncipe. Kernel principal components are maximum entropy projections. In *ICA 2006. Springer Lecture Notes in Computer Science 3889*, pages 846–853, 2006.

R.R. Phelps. *Lectures on Choquet's Theorem*, volume 1757 of *Lecture Notes in Mathematics*. Springer, 2nd edition, 2001.

M.S. Pinsker. *Information and Information Stability of Random Variables and Processes*. Holden-Day, 1964.

D. Pollard. Some thoughts on LeCam's statistical decision theory. Preprint, Statistics Department, Yale University, May 2000. URL `http://www.stat.yale.edu/~pollard/Papers/thoughts.pdf`.

H. Vincent Poor and John B. Thomas. Applications of Ali-Silvey distance measures in the design of generalized quantizers for binary decision systems. *IEEE Transactions on Communications*, 25(9):893–900, September 1977.

J.C. Principe, D. Xu, and J. Fisher. Information theoretic learning. In Simon Haykin, editor, *Unsupervised Adaptive Filtering: Volume 1, Blind Source Seperation*, pages 265–319, New York, 2000a. Wiley.

J.C. Principe, D. Xu, Q. Zhao, and J.W. Fisher III. Learning from examples with information theoretic criteria. *The Journal of VLSI Signal Processing*, 26(1–2):61–77, 2000b.

S.T. Rachev. *Probability metrics and the stability of stochastic models*. Wiley series in probability and mathematical statistics, Chichester, 1991.

S. Raudys. *Statistical and Neural Classifiers: An Integrated Approach to Design*, chapter Taxonomy of Pattern Classification Algorithms. Springer, London, 2001.

M.D. Reid and R.C. Williamson. Surrogate regret bounds for proper losses. In *Proceedings of the International Conference on Machine Learning*, pages 897–904, 2009.

M.D. Reid and R.C. Williamson. Composite binary losses. *Journal of Machine Learning Research*, 11:2387–2422, 2010.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1):107–136, 2006.

J. Rissanen. *Information and Complexity in Statistical Modelling*. Springer, New York, 2007.

C.P. Robert. *The Bayesian Choice: A Decision-Theoretic Motivation*. Springer, New York, 1994.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

R.T. Rockafellar and R.J-B. Wets. *Variational Analysis*. Springer-Verlag, Berlin, 2004.

Y.D. Rubinstein and T. Hastie. Discriminative vs informative learning. In *Knowledge Discovery and Data Mining*, pages 49–53. AAAI Press, 1997.

R. Sacksteder. A note on statistical equivalence. *The Annals of Mathematical Statistics*, 38(3): 787–794, 1967.

L.J. Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66(336):783–801, 1971.

A.H. Sayed and T. Kailath. A survey of spectral factorization methods. *Numerical Linear Algebra with Applications*, 8(6-7):467–496, 2001.

M.J. Schervish. A general method for comparing probability assessors. *The Annals of Statistics*, 17 (4):1856–1879, 1989.

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

B. Schölkopf, A. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.

C. Scott. Calibrated surrogate losses for classification with label-dependent costs. arXiv:1009:2718v1, September 2010.

C. Scott and M. Davenport. Regression level set estimation via cost-sensitive classification. *IEEE Transactions on Signal Processing*, 55(6 Part 1):2752–2757, 2007.

G. Shafer and V. Vovk. *Probability and Finance: It's Only a Game!* J. Wiley & Sons, 2001.

C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423; 623–656, 1948.

Y. Shen. *Loss Functions for Binary Classification and Class Probability Estimation*. PhD thesis, Department of Statistics, University of Pennsylvania, October 2005.

E. Shuford, A. Albert, and H.E. Massengill. Admissible probability measurement procedures. *Psychometrika*, 31(2):125–145, June 1966.

L.N. Soldatova and R.D. King. An ontology of scientific experiments. *Interface: The Journal of The Royal Society*, 3(11):795–803, 2006.

L. Song, M.D. Reid, A.J. Smola, and R.C. Williamson. Discriminative estimation of f-divergence. Submitted to AISTATS09, October 2008.

B.K. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. On integral probability metrics, ϕ-divergences and binary classification. Arxiv preprint arXiv:0901.2698v4, October 2009.

I. Steinwart. How to compare different loss functions and their risks. Preprint, Modeling, Algorithms and Informatics Group, CCS-3, Los Alamos National Laboratory, September 2006.

I. Steinwart. How to compare different loss functions and their risks. *Constructive Approximation*, 26(2):225–287, August 2007.

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

I. Steinwart, D. Hush, and C. Scovel. Density level detection is classification. *Advances in Neural Information Processing Systems*, 17, 2005.

H. Strasser. *Mathematical Theory of Statistics*. Walter de Gruyter, Berlin, 1985.

H. Strasser. Reduction of complexity. In Josef A. Mazanec and Helmut Strasser, editors, *A Nonparametric Approach to Perceptions-Based Market Segmentation: Foundations*, pages 99–140, Wien, 2000. Springer.

I.J. Taneja. Refinement inequalities among symmetric divergence measures. arXiv:math/0501303v2, April 2005a.

I.J. Taneja. Bounds on non-symmetric divergence measures in terms of symmetric divergence measures. arXiv:math.PR/0506256v1, 2005b.

D. Tasche. Conditional expectation as quantile derivative. Arxiv preprint math.PR/0104190, 2001.

W. B. Temple. Stieltjes integral representation of convex functions. *Duke Mathematical Journal*, 21:527–531, 1954.

N. Tishby, F.C. Pereira, and W. Bialek. The information bottleneck method. Arxiv preprint physics/0004057, 2000.

S. Tong and D. Koller. Restricted Bayes optimal classifiers. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, pages 658–664, 2000.

F. Topsøe. Some inequalities for information divergence and related measures of discrimination. *IEEE Transactions on Information Theory*, 46(4):1602–1609, 2000.

F. Topsøe. Bounds for entropy and divergence for distributions over a two-element set. *J. Ineq. Pure & Appl. Math*, 2(2), 2001.

F. Topsøe. Between truth and description. Presented at Prague Stochastics 2006, June 2006.

E.N. Torgersen. Measures of information based on comparison with total information and with total ignorance. *The Annals of Statistics*, 9(3):638–657, 1981.

E.N. Torgersen. *Comparison of Statistical Experiments*. Cambridge University Press, 1991.

G.T. Toussaint. Probability of error, expected divergence and the affinity of several distributions. *IEEE Transactions on Systems, Man and Cybernetics*, 8:482–485, 1978.

S. Turkle and S. Papert. Epistemological pluralism and the revaluation of the concrete. *Journal of Mathematical Behavior*, 11(1):3–33, March 1992.

P. Turney. Types of cost in inductive concept learning. In *Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning*, pages 15–21, 2000.

A. Unterreiter, A. Arnold, P. Markowich, and G. Toscani. On generalized Csiszár-Kullback inequalities. *Monatshefte für Mathematik*, 131:235–253, 2000.

I. Vajda. Note on discrimination and variation. *IEEE Transactions on Information Theory*, 16: 771–773, 1970.

A. van der Vaart. The statistical work of Lucien Le Cam. *Annals of Statistics*, 30(3):631–682, 2002.

V. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, New York, 2006. 2nd edition.

V. N. Vapnik. Inductive principles of the search for empirical dependences (methods based on weak convergence of probability measures). In *COLT '89: Proceedings of the second annual workshop on computational learning theory*, pages 3–21, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

H.R. Varian. Innovation, components and complements. University of California, Berkeley, October 2003. URL `http://www.almaden.ibm.com/coevolution/pdf/varian_paper.pdf`.

V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic learning in a random world*. Springer, 2005.

A. Wald. Statistical decision functions. *The Annals of Mathematical Statistics*, 20(2):165–205, June 1949.

A. Wald. *Statistical Decision Functions*. John Wiley & Sons, New York, 1950.

M.L. Weitzman. Recombinant growth. *Quarterly Journal of Economics*, 113(2):331–360, 1998.

D. Wettschereck and S. Muller. Exchanging data mining models with the predictive modelling markup language. *International Workshop on Integration and Collaboration Aspects of Data Mining, Decision Support and Meta-Learning*, 2001.

Wikipedia. Grothendieck's relative point of view. Wikipedia, September 2007. URL `http://en.wikipedia.org/wiki/Grothendieck's_relative_point_of_view`. Accessed 17/09/2007.

R. Winkler, J. Muñoz, J. Cervera, J. Bernardo, G. Blattenberger, J. Kadane, D. Lindley, A. Murphy, R. Oliver, and D. Ríos-Insua. Scoring rules and the evaluation of probabilities. *TEST*, 5(1):1–60, March 1990.

L. Withers. Some inequalities relating different measures of divergence between two probability distributions. *IEEE Transactions on Information Theory*, 45(5):1728–1735, 1999.

A.P. Worthen and W.E. Stark. Unified design of iterative receivers using factor graphs. *IEEE Transactions on Information Theory*, 47(2):843–849, 2001.

J. Xie and C.E. Priebe. A weighted generalization of the Mann-Whitney-Wilcoxon statistic. *Journal of Statistical Planning and Inference*, 102(2):441–466, 2002.

J.-W. Xu, D. Erdogmus, R. Jenssen, and J.C. Príncipe. An information-theoretic perspective to kernel independent component analysis. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP'05)*, volume 5, pages 249–252, 2005.

G.L. Yang and L. Le Cam. A conversation with Lucien Le Cam. *Statistical Science*, 14(2):223–241, 1999.

B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Third IEEE International Conference on Data Mining*, pages 435–442, 2003.

J. Zhang. Divergence function, duality, and convex analysis. *Neural Computation*, 16(1):159–195, 2004a.

J. Zhang and H. Matsuzoe. Dualistic riemannian manifold structure induced from convex functions. *Advances in Mechanics and Mathematics*., 17:437–464, 2009.

T. Zhang. Statistical behaviour and consistency of classification methods based on convex risk minimization. *Annals of Mathematical Statistics*, 32:56–134, 2004b.

V.M. Zolotarev. Probability metrics. *Theory of Probability and its Applications*, 38(2):278–302, 1984.

Journal of Machine Learning Research 12 (2011) 819-862 Submitted 7/09; Revised 5/10; Published 3/11

Learning Transformation Models for Ranking and Survival Analysis

Vanya Van Belle VANYA.VANBELLE@ESAT.KULEUVEN.BE
Katholieke Universiteit Leuven, ESAT-SCD
Kasteelpark Arenberg 10
B-3001 Leuven, Belgium

Kristiaan Pelckmans KRISTIAAN.PELCKMANS@IT.UU.SE
Uppsala University
Department of Information Technology
SysCon Polacksbacken
SE-751 05 Uppsala, Sweden

Johan A. K. Suykens JOHAN.SUYKENS@ESAT.KULEUVEN.BE
Sabine Van Huffel SABINE.VANHUFFEL@ESAT.KULEUVEN.BE
Katholieke Universiteit Leuven, ESAT-SCD
Kasteelpark Arenberg 10
B-3001 Leuven, Belgium

Editor: Nicolas Vayatis

Abstract

This paper studies the task of learning transformation models for ranking problems, ordinal regression and survival analysis. The present contribution describes a machine learning approach termed MINLIP. The key insight is to relate ranking criteria as the Area Under the Curve to monotone transformation functions. Consequently, the notion of a Lipschitz smoothness constant is found to be useful for complexity control for learning transformation models, much in a similar vein as the 'margin' is for Support Vector Machines for classification. The use of this model structure in the context of high dimensional data, as well as for estimating non-linear, and additive models based on primal-dual kernel machines, and for sparse models is indicated. Given n observations, the present method solves a quadratic program existing of $O(n)$ constraints and $O(n)$ unknowns, where most existing risk minimization approaches to ranking problems typically result in algorithms with $O(n^2)$ constraints or unknowns. We specify the MINLIP method for three different cases: the first one concerns the preference learning problem. Secondly it is specified how to adapt the method to ordinal regression with a finite set of ordered outcomes. Finally, it is shown how the method can be used in the context of survival analysis where one models failure times, typically subject to censoring. The current approach is found to be particularly useful in this context as it can handle, in contrast with the standard statistical model for analyzing survival data, all types of censoring in a straightforward way, and because of the explicit relation with the Proportional Hazard and Accelerated Failure Time models. The advantage of the current method is illustrated on different benchmark data sets, as well as for estimating a model for cancer survival based on different micro-array and clinical data sets.

Keywords: support vector machines, preference learning, ranking models, ordinal regression, survival analysis

©2011 Vanya Van Belle, Kristiaan Pelckmans, Johan A. K. Suykens and Sabine Van Huffel.

1. Introduction

Methods based on ranking continue to challenge researchers in different scientific areas, see, for example, Clémençon et al. (2005), Herbrich, Graepel, and Obermayer (2000) and the references therein. Learning ranking functions offers a solution to different types of problems including ordinal regression, bipartite ranking and discounted cumulative gain ranking (DCG, see Clémençon and Vayatis, 2007), studied frequently in research on information retrieval. These cases distinguish themselves in the definition (of the cardinality k) of the output domain and the chosen loss function. This paper deals with the general problem where the output domain can be arbitrary (with possibly infinite members $k = \infty$), but possesses a natural ordering relation between the members. Examples in which $k = \infty$ are found in survival analysis and preference learning in cases where the number of classes is not known in advance.

Earlier approaches to learning preference functions reduce the ranking problem to pairwise classification problems. This reasoning was followed in Ailon and Mohri (2008) and Fürnkranz and Hüllermeier (2003), Herbrich et al. (1998) and references therein. However, functions having high pairwise margins might still be bad approximations to real ranking problems. This is certainly the case in the (general) preference learning problem where possibly $k = \infty$: here a nonzero pairwise margin would need unnecessarily large parameters of the model. In this paper we address this issue by presenting a conceptual different approach: we adopt a smoothness condition on the ranking function to structure the space of ranking functions, and claim that this structure aligns in many applications better with the learning problems. This reasoning is motivated from relating a pairwise ranking criterion to a monotone *transformation* function. Besides empirical validation of this claim, we present formal relationships to other (statistical) models used for such tasks.

Figure 1 summarizes the ideas exposed in this work. First we describe the class of transformation models which contains two different components. The first component of a transformation model consists of a function $u : \mathbb{R}^d \to \mathbb{R}$ mapping the covariates $X \in \mathbb{R}^d$ to a value in $\mathbb{R}$ such that the natural order on $\mathbb{R}$ induces the ranking (approximately). Different names for such a function are found in literature depending on the problem setting, including a scoring, ranking, utility or health function. In this paper we will refer to this as to the utility function. The second component of the model maps this utility to an outcome in $\mathbb{R}$ by a transformation function $h : \mathbb{R} \to \mathbb{R}$. This is a univariate monotonically increasing function, basically capturing the scale of the output. The central observation now is that when one knows the ordinal relations between instances, one can estimate a transformation function mapping the instances to their utility value $\hat{u}(X)$. Depending on the problem at hand one is interested in the results of the first or second component of the transformation model. For ranking and survival analysis one typically ignores the second phase, whereas in ordinal regression a prediction of the output level is found by combining the first and the second components.

Transformation models are especially appropriate when considering data arising from a survival study. Survival analysis concerns data which represent a time-to-event, as, for example, a patient relapsing after surgery, or the time till a part of a mechanical device breaks down, see Kalbfleisch and Prentice (2002) for a broad survey of this field. The goal in survival analysis is often to relate time-to-event of an instance to a corresponding set of covariates. While practice and theoretical results here continue to have a strong impact in most quantitative scientific areas, survival analysis has been studied only sporadically in a context of machine learning, and such studies are mostly found in the field of artificial neural networks, see, for example, Biganzoli et al. (1998) and Kattan

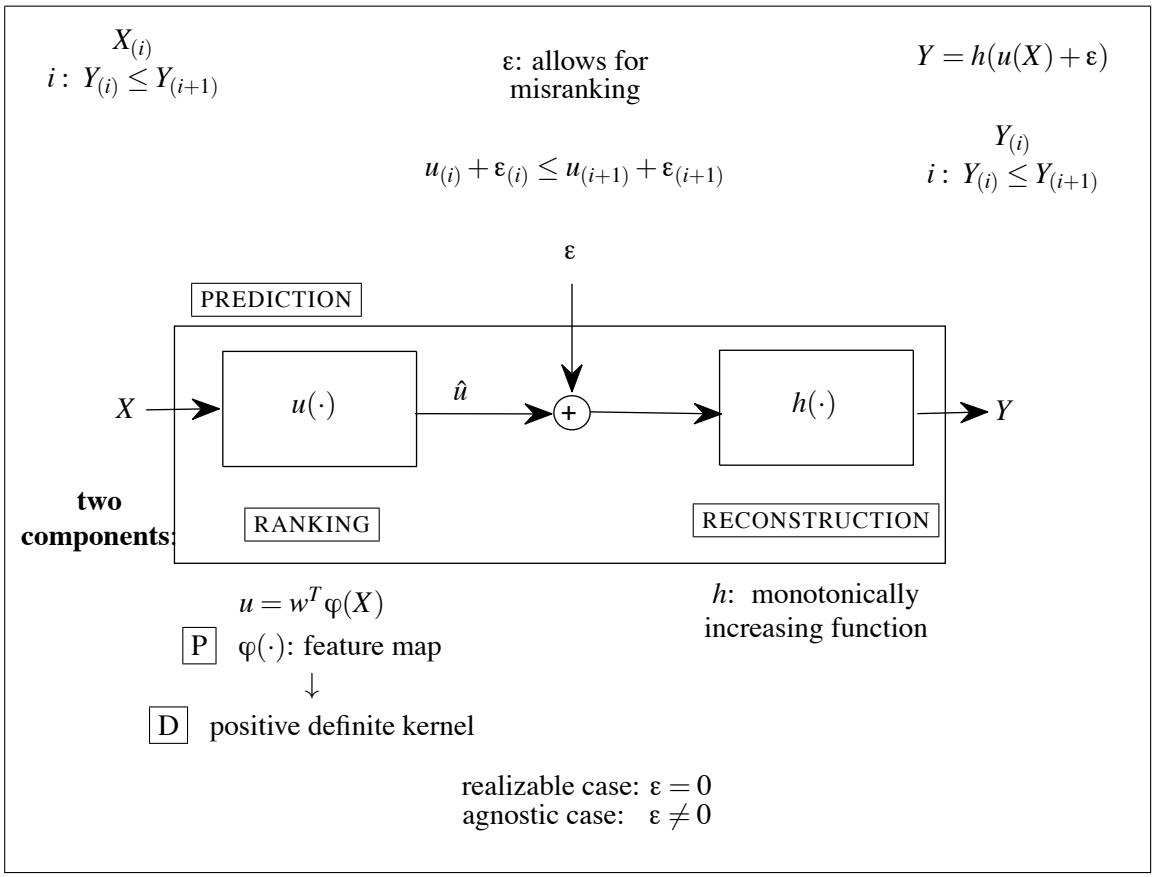


Figure 1: Overview: Transformation models consist of two components, the *utility function u* and a *transformation function h*. Given a data set $\mathcal{D} = \{(X_{(i)}, Y_{(i)})\}_{i=1}^n$ where the instances are sorted such that $Y_{(i)} \leq Y_{(i+1)}$, a utility function $u(X) = w^T \varphi(X)$ is trained such that the ranking on the evaluations of this function is representative for the ranking on the outcome. In the *realizable case* the ordering in utility will exactly coincide with the ordering in observed outcome $\{Y_{(i)}\}_i$. In the *agnostic case* however, the ordering will only be exact up to appropriate (nonzero) error variables $\{\varepsilon_i\}_i$. The modelling procedure will also be performed in two steps. The first step recovers u ('ranking'), while the second step is concerned with learning an explicit representation of the transformation function ('reconstruction'). In practice (depending on the problem at hand) one is mostly interested in implementing the first step only.

et al. (1997). However, we claim that there is a large potential for such studies: as (i) the approach of classical likelihood-based approaches have their intrinsic limitations, especially when a realistic underlying model cannot be assumed. A distribution-free approach is more appropriate here; (ii) A risk-based approach is often easier as one does not care about recovering the exact parameters describing the process of interest, but one is only interested in making good predictions, or exploring structure in the data; (iii) Computational issues for the classical statistical approach persist, and the

question how to solve the estimation equations numerically is often approached in an ad hoc way (if at all, see Kalbfleisch and Prentice, 2002 and references).

We find that the class of transformation models is a powerful tool to model data arising from survival studies for different reasons. The first reason being that they separate nicely the model for the time-scale (via the transformation function), and the qualitative characterization of an instance (via the utility function). We will furthermore argue on the close relationship with existing techniques as Cox' proportional hazard and accelerated failure time (AFT) models, see, for example, Dabrowska and Doksum, 1988, Koenker and Geling, 2001, Cheng, Wei, and Ying, 1997 and citations. In the following, we will relate the transformation function to ranking criteria as Kendall's τ or area under the curve (AUC), hence outlining a unified framework to study survival models as used in a statistical context and machine learning techniques for learning ranking functions. This relation indicates how one may apply the method of structural risk minimization (SRM, see Vapnik, 1998) here. The immediate consequence is the possibility to apply learning theory, with the capabilities to explain good performances in modelling high-dimensional data sets as well as for non-linear models (see Vapnik, 1998). Thirdly, in studies of failure time data, *censoring* is omnipresent. Censoring prohibits that one observes the actual event of interest fully, but gives partial information on the outcome instead. The prototypical case is 'a patient hasn't suffered the event as yet, but may experience an event in the future', but many other examples are studied. We will see how the proposed approach can handle censored observations conveniently.

The computational merit of this paper is then how one can fit such a model efficiently to data. Therefore, we consider an appropriate class of utility functions, either linear functions, or kernel based models. Secondly, instead of restricting attention to a parameterized class of transformation functions, we let the transformation function of interest be unspecified as one does for partial likelihood approaches, see Kalbfleisch and Prentice (2002). Especially, we define the appropriate transformation function only on the observed samples, by inferring an appropriate set of ordinal relations between them. Then we observe that the Lipschitz smoothness constant associated to such a transformation function can also be evaluated based on the samples only. Consequently, our fitting strategy called MINLIP finds the maximally smooth (implicitly defined) transformation function fitting the data samples. This is the *realizable case* where we can make the assumption of the existence of such a transformation model. In case we allow for misfit, we extend the model using slack-variables. It is then found that this problem can be solved as a convex Quadratic Program (QP), for which highly efficient software is readily available. In the case of utility functions which are kernel based models, we indicate how one can represent the solution as a sum of positive definite kernels, and the Lagrange dual problem again solves the corresponding problem as a convex QP. For the case linear utility functions are considered, we suggest how one can obtain zero parameters ('sparseness') suggesting structure in the data using an 1-norm regularization mechanism (see also Tibshirani, 1996).

Besides the conceptual and computational discussion, this paper gives empirical evidence for the approach. We consider empirical studies of ordinal regression and survival analysis. Performance of MINLIP on ordinal regression is analyzed using the ordinal data compiled by Chu and Keerthi (2005). MINLIP is applied on two different survival studies. A first study involves micro-array data sets: two breast cancer data sets (Sørlie et al., 2003; van Houwelingen et al., 2006) and one data set concerning diffuse large B-cell carcinoma (Rosenwald et al., 2002). In a last study, concerning a clinical breast cancer survival study (Schumacher et al., 1994), we investigate the estimation of

non-linear covariate effects and compare results obtained with MINLIP with Cox regression with penalized smoothing splines.

In Van Belle et al. (2007) we proposed a modification to standard SVMs to handle censored data. A computationally less demanding algorithm was presented in Van Belle et al. (2008). Starting from this latter model, we replaced the maximal margin strategy with the minimal Lipschitz smoothness strategy as presented in Van Belle et al. (2009). This work extends considerably the results of this short paper. Most notably, this paper additionally elaborates on the case of survival analysis and a number of new case studies. The different application areas in which the proposed method can be applied are summarized in Table 1. In addition, it is stated how the model needs to be used and which equations need to be solved to obtain the solution.

This paper is organized as follows. The following Section discusses in some detail the use of transformation models and its relation with ranking methods. Section 3 studies the estimator in a context of ranking. Section 4 specifies how MINLIP is to be used in a context of ordinal regression, where only k different output levels are possible. Section 5 discusses the use of MINLIP in the presence of censoring. In Section 6 experiments illustrate the use of the MINLIP method.

2. Transformation Models and Ranking Methods

In this paper we work in a stochastic context, so we denote random variables as capital letters, for example, $X, Y, \ldots$, which follow an appropriate stochastic law $P_X, P_Y, \ldots$, abbreviated (generically) as P. Deterministic quantities as constants and functions are represented in lower case letters (e.g., $d, h, u, \ldots$). Matrices are denoted as boldface capital letters (e.g., $\mathbf{X}, \mathbf{D}, \ldots$). Ordered sets will be denoted as $\{S_{(i)}\}$, indicating that $S_{(i)} \leq S_{(i+1)}$. Before the relation between transformation models and ranking methods can be explored, some terminology needs to be defined.

Definition 1 (Lipschitz smoothness) *A univariate function $h(Z)$ has a Lipschitz constant $L \geq 0$ if*

$$|h(Z) - h(Z')| \leq L|Z - Z'|, \ \forall\, Z, Z' \in \mathbb{R}.$$

A *transformation model* is then defined as follows:

Definition 2 (Transformation Model) *Let $h : \mathbb{R} \to \mathbb{R}$ be a* strictly *increasing function with Lipschitz constant $L < \infty$, and let $u : \mathbb{R}^d \to \mathbb{R}$ be a function of the covariates $X \in \mathbb{R}^d$. Let ε be a random variable ('noise') independent of X, with cumulative distribution function $F_\varepsilon(e) = P(\varepsilon \leq e)$ for any $e \in \mathbb{R}$. Then a Noisy Transformation Model (*NTM*) takes the form*

$$Y = h(u(X) + \varepsilon). \tag{1}$$

In the remainder of the paper, we will use Z to denote $u(X) + \varepsilon$ for notational convenience. Now the problem is reduced to estimating a utility function $u : \mathbb{R}^d \to \mathbb{R}$ and a transformation function h from a set of i.i.d. observations $\{(X_i, Y_i)\}_{i=1}^n$ without imposing any distributional (parametric) assumptions on the noise terms $\{\varepsilon_i\}$. Note that without structural assumptions, the utility can not uniquely be defined. Later on, we will specify similar assumptions as in the maximal margin strategy of Vapnik when introducing support vector machines, to find a unique solution for the utility function.

Task	Subtasks	algorithm	necessary training data for algorithm	result of algorithm	equation	comment
ranking	ranking	MINLIP	$\{(X_i, Y_i)\}$	$\hat{u}(X)$	(8)	Use in combination with (10)* to obtain sparse models
ordinal regression	ranking	MINLIP	$\{(X_i, Y_i)\}, B$	$\hat{u}(X), \hat{v}$	(11)	Use in combination with (10)* to obtain sparse models
	reconstruction	comparison with thresholds v	$\{\hat{u}(X_i)\}, \hat{v}$	$\hat{Y}$	Figure 5	The goal is to predict class label
survival analysis	ranking	MINLIP	$\{(X_i, Y_i)\}$	$\hat{u}(X)$	(16)	Use in combination with (10)* to obtain sparse models
	reconstruction	monotonic regression after replication in consecutive time intervals	$\{(\hat{u}(X_i), Y_i)\}$	$\{\hat{Y}_{il}\}$		The goal is to predict hazard and/or survival function

*Equation (10) can only be used in combination with a linear kernel

Table 1: Overview of methods and applications proposed in the paper. Depending on the problem at hand, a different version of the proposed model needs to be applied. Depending on the subtasks, different training data (variables X_i, target value Y_i, utility $u(X_i)$, dummy responses $B, \ldots$) need to be given to the algorithm to obtain the desired response (utility $u(X_i)$, transformation function $h(u(X))$, prediction $\hat{Y}_i = h(u(X_i))$, threshold values $\hat{v}$, risk on event within the l^{th} interval $\hat{Y}_{il}, \ldots$).

Kalbfleisch and Prentice (2002) considered transformation models for failure time models. The transformation models discussed in Cheng, Wei, and Ying (1997), Dabrowska and Doksum (1988) and Koenker and Geling (2001) differ from the above definition in the transformation function h. They define the model as $h^-(Y) = u(X) + \varepsilon$, which is equivalent to (1) if $h^-(h(Z)) = h(h^-(Z)) = Z$ for all Z.

To relate transformation models with ranking functions, we reason as follows. To express the performance of a ranking function one can use Kendall's τ, area under the curve (AUC) or a related measure. In this paper we will work with the *concordance* of a function $u : \mathbb{R}^d \to \mathbb{R}$ respective to the outcome. The concordance is defined as the probability that the order in outcome of two i.i.d. observations (X,Y) and (X',Y') is preserved in the utility u:

$$C(u) = P((u(X) - u(X'))(Y - Y') > 0). \tag{2}$$

Given a set of n i.i.d. observations $\{(X_i, Y_i)\}_{i=1}^n$, the empirical concordance index is then calculated as

$$C_n(u) = \frac{2}{n(n-1)} \sum_{i<j} I[(u(X_i) - u(X_j))(Y_i - Y_j) > 0],$$

where the indicator function $I(z)$ equals 1 if $z > 0$, and equals zero otherwise. Equivalently, the risk is defined as follows.

Definition 3 (Risk of (h,u)) *The risk associated with a monotonically increasing function penalizes discordant samples $h(u(X))$ and $h(u(X'))$ as*

$$\mathcal{R}(u) = P((h(u(X)) - h(u(X')))(Y - Y') < 0).$$

Or, since h is monotonically increasing, the risk is expressed as

$$\mathcal{R}(u) = P((u(X) - u(X'))(Y - Y') < 0).$$

Its empirical counterpart then becomes

$$\mathcal{R}_u(u) = 1 - C_n(u).$$

Empirical Risk Minimization (ERM) is then performed by solving

$$\hat{u} = \arg\min_{u \in \mathcal{U}} \mathcal{R}_u(u) = \arg\max_{u \in \mathcal{U}} C_n(u), \tag{3}$$

where $\mathcal{U} \subset \{u : \mathbb{R}^d \to \mathbb{R}\}$ is an appropriate subset of ranking functions, see, for example, Clémençon et al. (2005) and citations. However, this approach results in combinatorial optimization problems. One therefore majorizes the discontinuous indicator function by the Hinge loss, that is, $\ell(z) \leq \max(0, 1 - z)$ yielding rankSVM (Herbrich, Graepel, and Obermayer, 2000). The disadvantage of this solution is that it leads to $O(n^2)$ number of constraints or unknowns, often making it difficult to apply to real life problems. A solution to this problem is found in relating transformation models with Equation (3): if a function $u : \mathbb{R}^d \to \mathbb{R}$ exists such that $C_n(u) = 1$, one describes implicitly a transformation function (see Figure 2). If two variables u and y are perfectly concordant, then there exists a monotonically increasing function h such that $h(u)$ and y are perfectly concordant. Moreover, there exists such a function h, with Lipschitz constant L, mapping u to y such that $y = h(u)$. Or more formally:

Lemma 1 (Existence of a Transformation Function) *Given a collection of pairs* $\{(Z_{(i)},Y_{(i)})\}_{i=1}^n$, *enumerated such that* $Y_{(i)} \leq Y_{(j)}$ *if and only if* $i \leq j$, *and considering the conditions on the observations for* $L < \infty$*:*

$$0 \leq Y_{(i)} - Y_{(j)} \leq L\left(Z_{(i)} - Z_{(j)}\right), \ \forall\, i < j = 1,\ldots,n, \tag{4}$$

we state that:

1. *If one has for a finite value* $L \geq 0$ *that (4) holds, then there exists a monotonically increasing function* $h : \mathbb{R} \to \mathbb{R}$ *with Lipschitz constant L interpolating the data points.*

2. *If for all admissible* $(Z,Y) \in \mathbb{R} \times \mathbb{R}$ *one has that* $Y = h(Z)$ *for an (unknown) continuous, (finite) differentiable and monotonically increasing function* $h : \mathbb{R} \to \mathbb{R}$, *then there is a value* $L < \infty$ *such that (4) holds.*

Proof To prove 1, consider the linear interpolation function $h_n : \mathbb{R} \to \mathbb{R}$, defined as

$$h_n(Z) = \frac{Z - Z_{\underline{z}(Z)}}{Z_{\overline{z}(Z)} - Z_{\underline{z}(Z)}}\left(Y_{\overline{z}(Z)} - Y_{\underline{z}(Z)}\right) + Y_{\underline{z}(Z)},$$

where we define $\overline{z}(Z) = \arg\min_{i\in\{1,\ldots,n\}} (Z_i : Z_i > Z)$ and $\underline{z}(Z) = \arg\max_{i\in\{1,\ldots,n\}} (Z_i : Z_i \leq Z)$. Direct manipulation shows that this function is monotonically increasing and continuous. Now take $Z < Z' \in \mathbb{R}$, then we have to show that $h_n(Z') - h_n(Z) \leq L(Z' - Z)$. For notational convenience define $l = \underline{z}(Z)$, $u = \overline{z}(Z)$, $l' = \underline{z}(Z')$ and $u' = \overline{z}(Z')$, then

$$\begin{aligned} h_n(Z') - h_n(Z) &= \frac{Z' - Z_{l'}}{Z_{u'} - Z_{l'}}(Y_{u'} - Y_{l'}) + Y_{l'} - \frac{Z - Z_l}{Z_u - Z_l}(Y_u - Y_l) - Y_l \\ &\leq L(Z' - Z_{l'}) - L(Z - Z_l) + L(Z_{l'} - Z_l) \\ &= L(Z' - Z), \end{aligned}$$

where we use that $Y_{l'} - Y_l \leq L(Z_{l'} - Z_l)$.

Item 2 is proven as follows. Let such an h exist, then the mean value theorem asserts that for any two samples (Z_i,Y_i) and (Z_j,Y_j) for which $Z_i \leq Z_j$, there exists a Z within the interval $[Z_i,Z_j] \subset \mathbb{R}$ such that

$$(Y_j - Y_i) = (Z_j - Z_i)h'(Z) \leq L(Z_i - Z_j),$$

where $L = \sup_Z h'(Z)$.
Note that Equation (4) implies that $C_n(Z) = 1$. ■

3. MINLIP: A Convex Approach to Learning a Transformation Model

This Section describes how transformation models can be learned by means of a convex approach. The Section starts with a discussion of the realizable case and extends this model formulation towards the agnostic case and non-linearities using Mercer kernels.

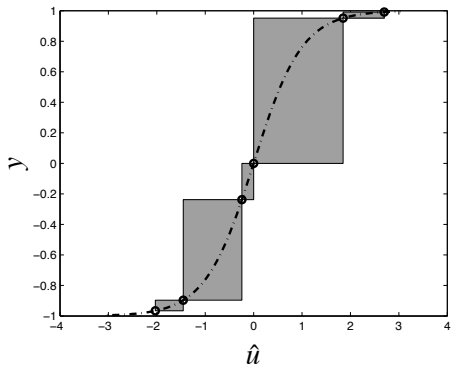


Figure 2: Relation between ranking and transformation models: if two variables u and y are perfectly concordant, they describe a monotonically increasing function $y = h(u)$. The dots represent u and outcome y for training points. In these observations the value of the function h is known exactly. To predict the $y-$value of the test observations, the function h needs to be approximated between the training points (grey area). All functions $\hat{h}$ which are monotonically increasing and lie within the grey zones are valid prediction rules.

3.1 The Realizable Case

The realizable case refers to the situation where there exists a function $u(X)$ such that the ranking of $u(X)$ perfectly reflects the ranking of Y. Otherwise stated, there exists a function $u(X)$ such that $\mathcal{C}(u) = 1$. Lemma 1 describes the existence of h, but since this transformation function is only known at the training points, it is not unique. Figure 2 illustrates that all monotonically increasing functions lying within the grey bounds satisfy the conditions. Therefore, the Lipschitz constant is used to control the complexity of the transformation function. Transformation functions with a smaller Lipschitz constant will be preferred. For notational convenience, we will assume no coinciding outcomes (ties). Let h be a monotonically increasing function with Lipschitz constant $L < \infty$, such that $h(Z) - h(Z') \leq L(Z - Z')$ for all $Z \geq Z'$. Restricting attention to the observations $\{(X_{(i)}, Y_{(i)})\}_{i=1}^{n}$, one has the necessary conditions

$$h\left(u(X_{(i)})\right) - h\left(u(X_{(i-1)})\right) \leq L\left(u(X_{(i)}) - u(X_{(i-1)})\right) ,$$

for all $i = 2, \ldots, n$. Here, we assume that the data obey a noiseless transformation model ($\varepsilon = 0$ in (1)). For now, linear utility functions defined as

$$u(X) = w^T X ,$$

are considered. Extensions towards non-linear utility functions using Mercer kernels are handled in Subsection 3.3. Since the function $u(X) = w^T X$ can be arbitrary rescaled such that the corresponding transformation function has an arbitrary Lipschitz constant (i.e., for any $\alpha > 0$, one has $h(u(X)) = \tilde{h}(\tilde{u}(X))$ where $\tilde{h}(Z) \triangleq h(\alpha^{-1} Z)$ and $\tilde{u}(X) = \alpha u(X)$), we fix the norm $w^T w$ and try to find $u(X) = v^T X$ with $v^T v = 1$. Hence learning a transformation model with minimal Lipschitz

constant of h can be written as

$$\begin{array}{ll} \min\limits_{v,L} & \frac{1}{2}L^2 \\ \text{s.t.} & \begin{cases} \|v\|_2 = 1 \\ Y_{(i)} - Y_{(i-1)} \leq L\left(v^T X_{(i)} - v^T X_{(i-1)}\right), \ \forall\, i = 2,\ldots,n. \end{cases} \end{array}$$

Substituting $w = Lv$ we get equivalently:

$$\begin{array}{ll} \min\limits_{w} & \frac{1}{2}w^T w \\ \text{s.t.} & Y_{(i)} - Y_{(i-1)} \leq w^T X_{(i)} - w^T X_{(i-1)}, \ \forall\, i = 2,\ldots,n, \end{array} \tag{5}$$

which goes along similar lines as the hard margin SVM (see, e.g., Shawe-Taylor and Cristianini, 2004) and ranking SVM (Freund et al., 2004), where the threshold value 1 is replaced by $Y_{(i)} - Y_{(i-1)}$. Note that an intercept term is not needed since differences in utility are used. Observe that this problem has $n-1$ linear constraints. We will refer to this model as MINLIP.

Problem (5) can be compactly rewritten as

$$\begin{array}{ll} \min\limits_{w} & \frac{1}{2}w^T w \\ \text{s.t.} & \mathbf{D}\mathbf{X}w \geq \mathbf{D}\mathbf{Y}, \end{array}$$

where $\mathbf{X} \in \mathbb{R}^{n\times d}$ is a matrix with each row containing one observation, that is, $\mathbf{X}_i = X_{(i)} \in \mathbb{R}^d$ and $\mathbf{Y} = [Y_{(1)} \cdots Y_{(n)}]^T$, a vector with the corresponding outcomes. The matrix $\mathbf{D} \in \{-1,0,1\}^{(n-1)\times n}$

$$\mathbf{D} = \begin{bmatrix} -1 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & -1 & 1 & 0 & \ldots & 0 & 0 \\ \vdots & & & & & & \vdots \\ 0 & \ldots & 0 & 0 & 0 & -1 & 1 \end{bmatrix},$$

gives the first order differences of a vector, that is, assuming no ties in the output, $\mathbf{D}_i\mathbf{Y} = Y_{(i+1)} - Y_{(i)}$ for all $i = 1,\ldots,n-1$, with $\mathbf{D}_i$ the i^{th} row of $\mathbf{D}$.

In the presence of ties, $Y_{(i+1)}$ is replaced by $Y_{(j)}$, with j the smallest output value with $Y_{(j)} > Y_{(i)}$. See Section 4 for more details. Solving this problem as a convex QP can be done efficiently with standard mathematical solvers as implemented in MOSEK[1] or R-quadprog.[2] The following proposition states when the MINLIP model is valid.

Proposition 1 (Validity of MINLIP) *Assume that $(X,Y) \in \mathbb{R}^d \times \mathbb{R}$ would obey the relation*

$$Y = h_0(w_0^T X), \tag{6}$$

where we refer to the (fixed but unknown) vector $w_0 \in \mathbb{R}^d$ as to the 'true' parameters, and to the (fixed but unknown) monotonically increasing function $h_0 : \mathbb{R} \to \mathbb{R}$ as the 'true' transformation function. Let for each couple (X,Y) and (X',Y') where $Y \neq Y'$ the constant $L' > 0$ be defined as

$$\frac{1}{L'} = \frac{w_0^T(X - X')}{Y - Y'},$$

1. MOSEK can be found at `http://www.mosek.org`.
2. R-quadprog can be found at `http://cran.r-project.org/web/packages/quadprog/index.html`.

where $L' = \infty$ if $w_0^T(X - X') = 0$. By construction we have that $L' \leq L_0$ and that the constant exists everywhere. The result of the MINLIP *model then becomes:*

$$\pounds = \max_{\|w\|_2=1} \min_{Y>Y'} \frac{w^T(X-X')}{Y-Y'} \quad = \quad \max_{\|w\|_2=1} \min_{Y>Y'} \frac{w^T(X-X')}{L' w_0^T(X-X')}. \tag{7}$$

We then state that MINLIP *yields a good approximation of the parameter vector w_0 in the noiseless case as long as there are enough observations (X,Y) such that $w_0^T(X-X') \approx 1$ and $L' \approx L_0$.*

Proof Let the unit-length vector $\overline{(X-X')} \in \mathbb{R}^d$ be defined as $X - X' = \overline{(X-X')}\|X-X'\|_2$, then we can write (7) as

$$\max_{\|w\|_2=1} \min_{Y>Y'} \frac{w^T \overline{(X-X')}}{L' w_0^T \overline{(X-X')}}.$$

Let us now focus attention on the set $\mathcal{S} = \{(X-X', Y-Y') : (X,Y),(X',Y') \in \mathcal{D} = \{X_i, Y_i\}_{i=1}^n\}$, where $\pounds = w^T(X-X')/(Y-Y')$ for which this value $\pounds$ is actually achieved. It is seen that the estimate w lies in the span of this set $\mathcal{S}$ as otherwise the maximum value could be increased. When we assume that the data set contains enough observations (X,Y) such that $w_0^T(X-X') \approx 1$ and $L' \approx L_0$, they will end up in the set $\mathcal{S}$, and as a result we have that $w^T w_0 \approx 1$. As the optimal solution is fully determined by the terms $w_0^T \overline{(X-X')} \approx 1$ and $L' \approx L_0$ (cfr. duality results in convex optimization), one should also have that $w \approx w_0$. ■

Formally, consistency of MINLIP in the asymptotic case under a sufficient condition of the data being non-degenerate is derived in Appendix A.

3.2 The Agnostic Case

In case it is impossible to find a utility function $u : \mathbb{R}^d \to \mathbb{R}$ extracting the ranking perfectly, a noisy transformation model is considered:

$$Y = h(w^T X + \varepsilon),$$

where $u = w^T X$. The introduction of the error variable asks for an adaptation of the Lipschitz-based complexity control. As a loss function $\ell : \mathbb{R} \to \mathbb{R}$ we choose the absolute value loss $\ell(\varepsilon) = |\varepsilon|$ for three reasons: (i) It is known that this loss function is more robust to misspecification of the model and outliers than, for example, the squared loss $\ell(\varepsilon) = \varepsilon^2$; (ii) The use of the absolute value loss will result in sparse solutions with many error terms equal to zero; (iii) In binary classification this norm is well performing in SVMs. However, the choice of the loss remains arbitrary. Incorporation of the errors (slack variables) leads to the following model formulation:

$$\begin{array}{ll} \min\limits_{w,\varepsilon} & \frac{1}{2} w^T w + \gamma \|\varepsilon\|_1 \\ \text{s.t.} & \mathbf{D}(\mathbf{X}w + \varepsilon) \geq \mathbf{DY}, \end{array} \tag{8}$$

where $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)^T \in \mathbb{R}^n$ represents the errors, $\|\varepsilon\|_1 = \sum_{i=1}^n |\varepsilon_i|$ and $\gamma > 0$ is a regularization constant, making a trade-off between model complexity and error. This problem can again be solved as a convex quadratic program.

3.3 A Non-linear Extension using Mercer Kernels

Let $\varphi : \mathbb{R}^d \to \mathbb{R}^{d_\varphi}$ be a feature map mapping the data to a high dimensional feature space (of dimension d_φ, possibly infinite). A non-linear utility function can then be defined as

$$u(X) = w^T \varphi(X),$$

with $w \in \mathbb{R}^{d_\varphi}$ a vector of unknowns (possibly infinite dimensional). Take $\mathbf{\Phi} = [\varphi(X_{(1)}), \ldots, \varphi(X_{(n)})]^T \in \mathbb{R}^{n \times d_\varphi}$. The realizable learning problem can then be represented as:

$$\begin{array}{ll} \min\limits_{w} & \frac{1}{2} w^T w \\ \text{s.t.} & \mathbf{D\Phi} w \geq \mathbf{DY}, \end{array}$$

with the matrix $\mathbf{D}$ defined as before. The Lagrange dual problem becomes

$$\begin{array}{ll} \min\limits_{\alpha} & \frac{1}{2} \alpha^T \mathbf{DKD}^T \alpha - \alpha^T \mathbf{DY} \\ \text{s.t.} & \alpha \geq 0_{n-1}, \end{array}$$

where the kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ contains the kernel evaluations such that $\mathbf{K}_{ij} = \varphi(X_i)^T \varphi(X_j)$ for all $i, j = 1, \ldots, n$. The estimated utility $\hat{u}$ can be evaluated at any point $X^* \in \mathbb{R}^d$ as

$$\hat{u}(X^*) = \hat{\alpha}^T \mathbf{DK}_n(X^*), \tag{9}$$

where $\mathbf{K}_n(X^*) = [K(X_1, X^*), \ldots, K(X_n, X^*)]^T \in \mathbb{R}^n$. The dual (Shawe-Taylor and Cristianini, 2004; Suykens, Gestel, Brabanter, Moor, and Vandewalle, 2002; Vapnik, 1998) of the agnostic learning machine of Subsection 3.2 is obtained analogously:

$$\begin{array}{ll} \min\limits_{\alpha} & \frac{1}{2} \alpha^T \mathbf{DKD}^T \alpha - \alpha^T \mathbf{DY} \\ \text{s.t.} & \begin{cases} -\gamma 1_n \leq \mathbf{D}^T \alpha \leq \gamma 1_n \\ \alpha \geq 0_{n-1}, \end{cases} \end{array}$$

with $\mathbf{K}$ as above and the resulting estimate can be evaluated as in (9) without computing explicitly $\hat{w}$ nor φ. We refer to Appendix B for a detailed derivation. Typical choices for kernel functions are:

$$\begin{aligned} K(X, X_i) &= X_i^T X \text{ (linear kernel)} \\ K(X, X_i) &= (\tau + X_i^T X)^d, \ \tau \geq 0 \text{ (polynomial kernel of degree d)} \\ K(X, X_i) &= \exp\left(-\frac{||X - X_i||_2^2}{\sigma^2} \right) \text{ (RBF kernel)}. \end{aligned}$$

In cases where one is interested in the modelling of covariate effects, one could use an additive utility function:

$$u(X) = \sum_{p=1}^{d} u^p(X^p),$$

where X^p represents the p^{th} covariate of datapoint X. Using Equation (9) this can be written as:

$$\begin{aligned}\hat{u}(X) &= \sum_{p=1}^{d} \alpha^T \mathbf{D}\mathbf{K}^p(X^p) \\ &= \alpha^T \mathbf{D} \sum_{p=1}^{d} \mathbf{K}^p(X^p),\end{aligned}$$

where the kernel matrix $\mathbf{K}^p \in \mathbb{R}^{n\times n}$ contains the kernel evaluations such that $\mathbf{K}^p_{ij} = \varphi(X^p_i)^T\varphi(X^p_j)$ for all $i,\ j = 1,\ldots,n$. As a result, componentwise kernels (Pelckmans et al., 2005b):

$$K(X,X_i) = \sum_{p=1}^{d} K^p(X^p,X^p_i),$$

which can be seen as a special case of ANOVA kernels (Vapnik, 1998), can be used. The use of such componentwise kernels allows for interpreting the non-linear effects of the covariates.

3.4 Prediction with Transformation Models

Prediction of the outcome using transformation models is a two-step approach (see Figure 1). First, the utility $u(X)$ is estimated, giving an ordering relation between the observations. When interested in an outcome prediction, the transformation function h has to be estimated. The prediction step is a univariate regression problem, which can be solved using monotonic regression models. Remark that in the ranking setting, one is not interested in the estimation of the transformation function since the goal is to find the ranking. Estimation of the transformation function for ordinal regression and survival analysis will be illustrated later.

3.5 Toward Sparse Solutions using $\|w\|_1$

This subsection describes an extensions to the above model. Specifically, we will be interested in the case where d is large compared to n. Consequently, we will be interested in computational methods which reveal the relevant input variables of use in the learned prediction rule. We restrict ourselves to the primal case where $u(X) = w^TX$ for the linear case and an unknown monotonically increasing function $h : \mathbb{R} \to \mathbb{R}$. In this extension an l_1 penalty (Tibshirani, 1996) is used instead of the term w^Tw. We shall refer to this model as MINLIP$_{L1}$:

$$\begin{array}{ll}\min\limits_{w,\varepsilon} & \|w\|_1 + \gamma\|\varepsilon\|_1 \\ \text{s.t.} & \mathbf{D}(\mathbf{X}w + \varepsilon) \geq \mathbf{DY},\end{array} \tag{10}$$

where $\|w\|_1 = \sum_{p=1}^{d} |w_p|$. This linear programming problem (LP) can be solved efficiently with standard mathematical solvers. This formulation does not allow for a straightforward dual derivation.

Figure 3 illustrates the possible advantage of the sparse alternative over the standard MINLIP formulation. We created 100 artificial data sets, each containing 150 observations with 200 covariates. 100 observations were used for training, the remaining for testing. A varying number of $d = 100, 110, \ldots, 200$ covariates were used to build the outcome, all other features being irrelevant. All covariates were drawn from a normal distribution with zero mean and standard deviation 1. The outcome was obtained as a weighted sum of the relevant covariates, where the weights were drawn from a standard normal distribution. The test error of the MINLIP$_{L1}$ model was lower than for the standard model.

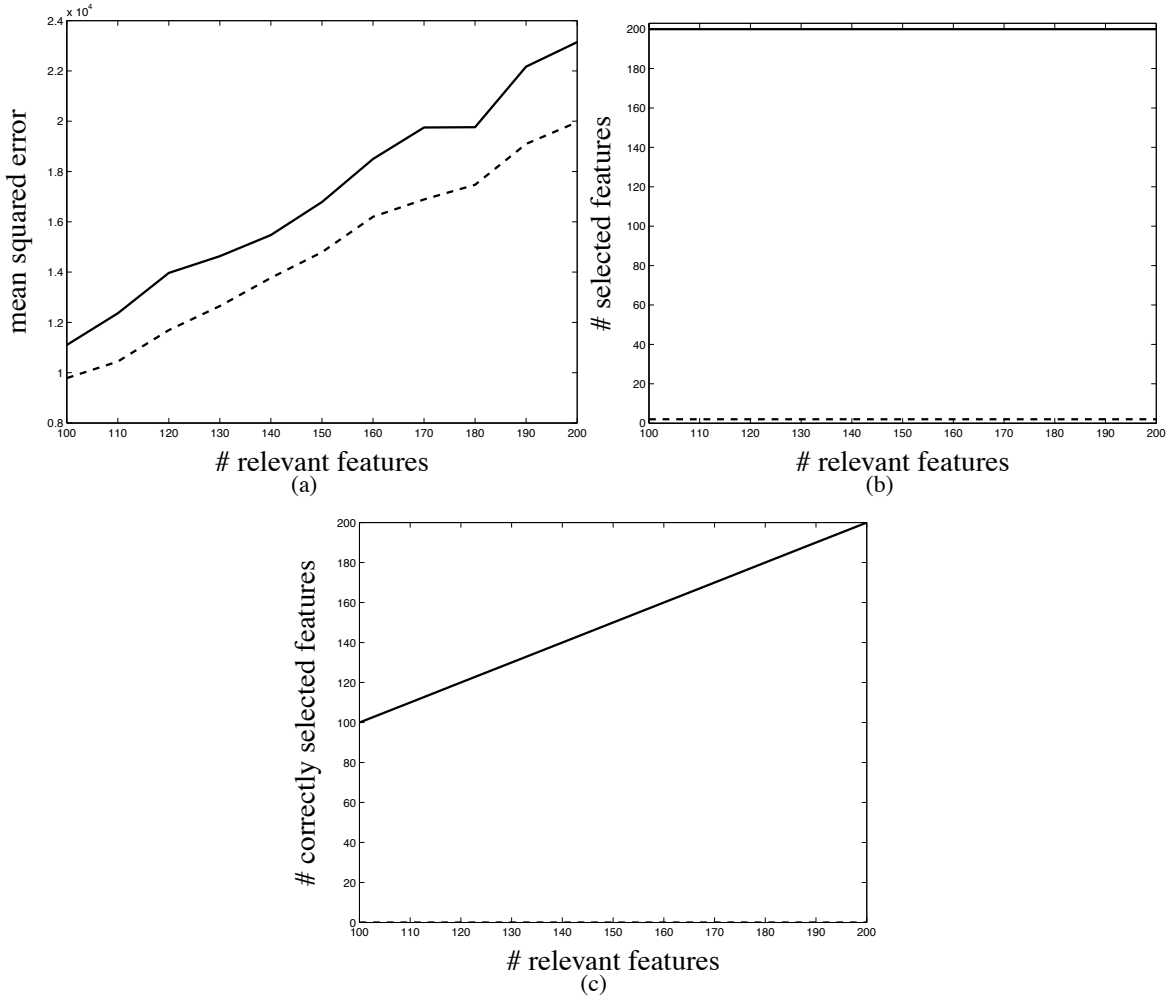


Figure 3: Performance and feature selection ability of MINLIP (solid) and $\text{MINLIP}_{\text{L1}}$ (dashed) on an artificial data set (n=100 for training, n_{test}=50 for testing). 200 $\mathcal{N}(0,1)$ distributed covariates were generated, a varying number $d = 100, 110, \ldots, 200$ of which were used to generate the outcome ($Y = \sum_{p=1}^{d} w^p X^p$, with w drawn form a standard normal distribution). The results are averaged over 100 data sets. (a) Median mean squared error on the test sets: $\text{MINLIP}_{\text{L1}}$ performs better than MINLIP. (b-c) Number of selected (absolute value of estimated weight$> 10^{-8}$) and correctly selected variables versus number of relevant variables. The MINLIP method selects all variables, a lot of them not being relevant. The $\text{MINLIP}_{\text{L1}}$ model selects very few variables, but those which are selected are also relevant.

3.6 Comparison with Other Methods

An approach often seen within preference ranking problems is the reformulation of the ranking problem as a classification problem. Examples of this strategy can be found in Ailon and Mohri (2008), Fürnkranz and Hüllermeier (2003) and Herbrich et al. (1998). However, transforming ranking to classification deflects attention from the underlying problem within ranking problems. In contrast with these methods, the MINLIP approach concentrates on the ranking problem by use of the transformation model.

Currently used ranking methods include rankSVM (Herbrich, Graepel, and Obermayer, 2000) and RankBoost (Freund et al., 2004). Although the method proposed here and rankSVM are both based on SVMs, two differences can be noted: (i) firstly, the rankSVM uses all pairs of data points for training, which results in $O(n^2)$ comparisons, where MINLIP has a complexity of $O(n)$. This reduction in complexity makes the model more applicable to large data sets; (ii) Secondly, the complexity control, being the margin and the Lipschitz constant, is different in both methods. In rankSVM all margins are equal and the model is tuned to maximize this margin. In MINLIP the margins differ corresponding to the difference in the output levels.

4. Learning for Ordinal Regression

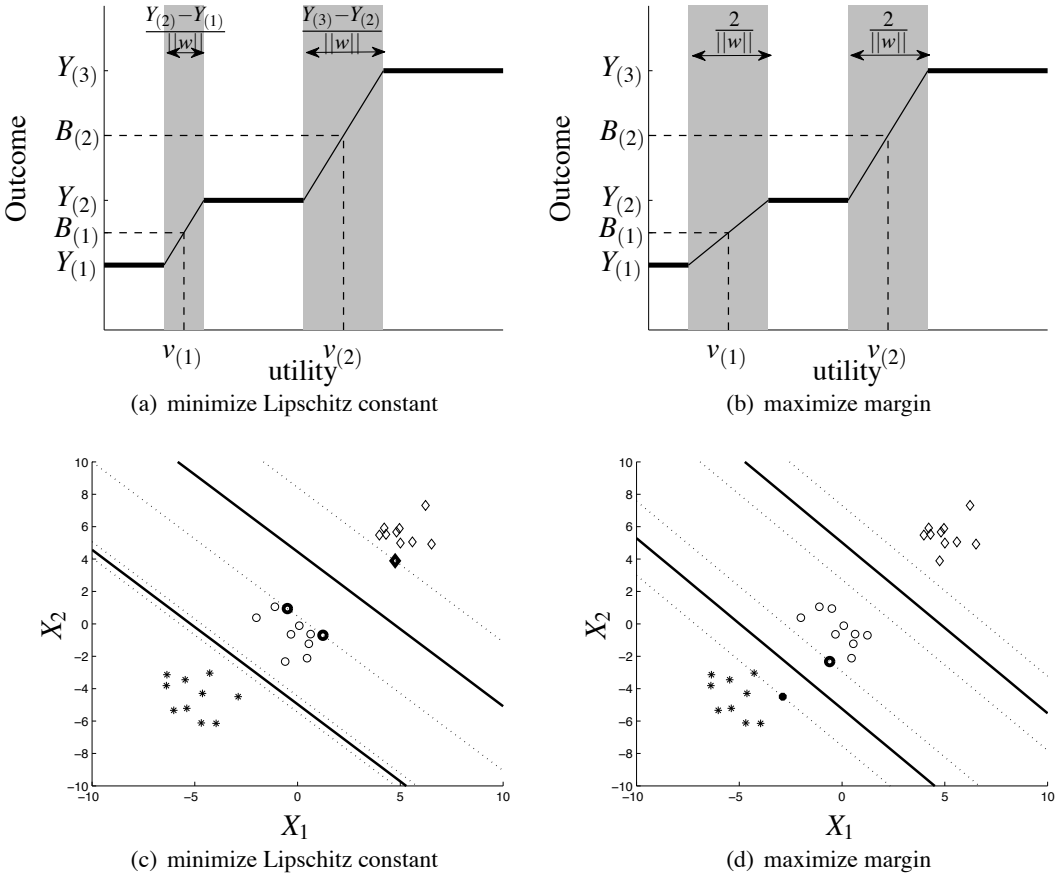
Consider now the situation where the output takes a finite number of values - say $k \in \mathbb{N}$ - and where the k different classes possess a natural ordering relation. In this case the outcome Y is an element of the finite ordered set $\{Y_{(1)}, \ldots, Y_{(k)}\}$.

4.1 A Modification to MINLIP

In Section 3.1 it is mentioned that comparisons are made between points (i) and (j) where $Y_{(j)}$ is the first ordered value bigger than $Y_{(i)}$. Applying this methodology in the ordinal setting would lead to as many comparisons with point (i) from class k_i as there are observations in class $k_i + 1$. To cope with this issue, we add dummy observations (X, B) in between two consecutive ordinal classes with levels $Y_{(i)} < Y_{(i+1)}$ such that $B_{(i)} = \frac{1}{2}(Y_{(i+1)} + Y_{(i)})$ (see Figure 4) and leaving their covariates and utility function unspecified. This implies that one has to compare each observation only twice, once with the dummy observation in between the previous and the current ordinal class and once with the dummy observation in between the current and the next class, restricting the number of constraints to $O(n)$. The solution of this problem can be found implicitly by extending the $\mathbf{Y} \in \mathbb{R}^n$ and $\mathbf{X} \in \mathbb{R}^{n \times d}$ matrices as follows:

$$\mathbf{Y} = \left[\begin{array}{c} \mathbf{Y} \\ \hline B_{(1)} \\ B_{(2)} \\ \cdots \\ B_{(k-1)} \end{array}\right] \quad \text{and} \quad \mathbf{X} = \left[\begin{array}{c|c} \mathbf{X} & 0 \\ \hline 0 & I_{k-1} \end{array}\right],$$

where $\mathbf{X} \in R^{(n+k-1)\times(d+k-1)}$ and $\mathbf{Y} \in \mathbb{R}^{n+k-1}$ and I_{k-1} represents the identity matrix of dimension $k-1$. The problem is then formulated as in Equation (8) after replacing $\mathbf{X}$ by $\mathbf{X}$ and $\mathbf{Y}$ by $\mathbf{Y}$ and results in the parameter vector $w = [w; v]$.



(a) minimize Lipschitz constant

(b) maximize margin

(c) minimize Lipschitz constant

(d) maximize margin

Figure 4: Adaptation of the MINLIP algorithm to ordinal regression: (a) Inclusion of dummy data points with output values B_i intermediate to the observed output values and undefined covariates and utility v_i. All data points are compared with two dummy data points; (b) Comparison with the maximal margin strategy used in standard SVM where the margin is equal between all classes; (c-d) Example with 3 linearly separable cases with outcomes equal to 1 (stars), 2 (circles) and 10 (diamond) respectively. The bold symbols represent the support vectors. In the maximal margin strategy there exists one margin, equal between every two successive classes, which results in a different Lipschitz constant. Using the MINLIP strategy, the Lipschitz smoothness is optimized, resulting in margins which are proportional to the difference in the class labels. Support vectors of the latter method are therefore more likely to be observations of two classes for which the output labels differ the most.

Using the above formulation, the thresholds v as well as the weights w are regularized. Since the motivation for this regularization scheme is not clear, one can formulate the problem explicitly as:

$$\begin{aligned} \min_{w,e,e^*,v} \quad & \|w\|_2 + \gamma \textstyle\sum_{i=1}^{n} 1_n^T (e + e^*) \\ \text{s.t.} \quad & \begin{cases} \mathbf{X}w - \mathbf{Q}v + e \geq \mathbf{Y} - \mathbf{QB} \\ -\mathbf{X}w + \mathbf{Q}^* v + e^* \geq -\mathbf{Y} + \mathbf{Q}^* \mathbf{B} \\ e \geq 0 \\ e^* \geq 0 \\ \mathbf{M}v \leq 0\,, \end{cases} \end{aligned} \tag{11}$$

with γ a positive regularization constant, $\mathbf{Q}$ and $\mathbf{Q}^* \in \mathbb{R}^{n \times (k-1)}$ matrices with all elements equal to zero except for positions $\{(i, k_i - 1)\}_{k_i=2}^{k}$ and $\{(i, k_i)\}_{k_i=1}^{k-1}$ respectively (where k_i represents the index of the output level of observation i), which contain ones. These positions correspond to the dummy data points with which one wishes to compare data points i. Vector $\mathbf{B} \in \mathbb{R}^{k-1}$ contains outcomes corresponding to the thresholds: $\mathbf{B} = [B_{(1)}, \cdots, B_{(k-1)}]^T$. Vector v contains all unknown utility function values for the dummy data points $v = [v_{(1)}, \cdots, v_{(k-1)}]^T$, and $\mathbf{M} \in \mathbb{R}^{(k-1) \times k}$ gives the first order differences of a vector and is defined as:

$$\mathbf{M} = \begin{bmatrix} 1 & -1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & -1 & 0 & \dots & 0 & 0 \\ \vdots & & & & & & \vdots \\ 0 & \dots & 0 & 0 & 0 & 1 & -1 \end{bmatrix}.$$

The Lagrange dual problem becomes

$$\begin{aligned} \min_{\alpha,\beta} \quad & \tfrac{1}{2}\alpha^T \mathbf{K}\alpha + \tfrac{1}{2}\beta^T \mathbf{K}\beta - \alpha^T \mathbf{K}\beta - \alpha^T (\mathbf{Y} - \mathbf{B}^T \mathbf{Q}) + \beta^T (\mathbf{Y} - \mathbf{B}^T \mathbf{Q}^*) \\ \text{s.t.} \quad & \begin{cases} 0_n \leq \alpha \leq \gamma 1_n \\ 0_n \leq \beta \leq \gamma 1_n \\ 0_{k-2} \leq \nu \\ \mathbf{Q}^T \alpha - \mathbf{Q}^{*T} \beta + \mathbf{M}^T \nu = 0_{k-1}\,, \end{cases} \end{aligned}$$

where 1_n and 0_n represent column vectors of size n with all elements equal to 1 and 0 respectively. Solving this explicit formulation is computationally less demanding and faster than solving the implicit problem formulation. We refer to Appendix C for a detailed derivation. The estimated $\hat{u}$ can be evaluated at any point $X^* \in \mathbb{R}^d$ as

$$\hat{u}(X^*) = (\hat{\alpha}^T - \hat{\beta}^T)\mathbf{K}_n(X^*)\,,$$

with $\mathbf{K}_n(X^*)$ defined as before.

4.2 Prediction for Ordinal Regression

A clear advantage of the approach which includes unknown thresholds is that the prediction step becomes very simple. As illustrated in Figure 5, the predictions can be easily obtained from the value of the utility function in comparison with the different threshold values.

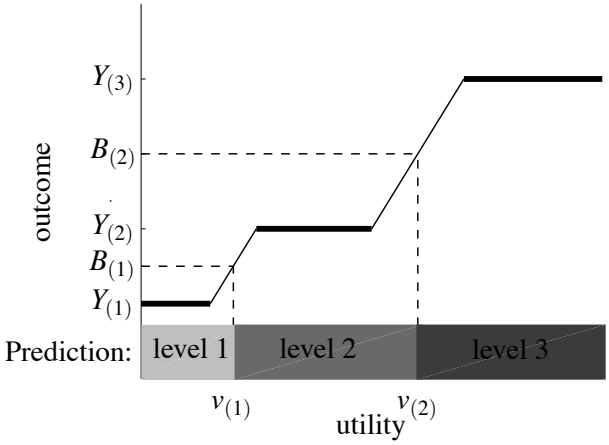


Figure 5: Prediction for ordinal regression. MINLIP for ordinal regression, including unknown thresholds, has the advantage to reduce the prediction step to a simple comparison between the utility of a new observation and the utility of the thresholds. If the utility has a value between threshold $j-1$ and j, the predicted outcome equals the j^{th} output level.

4.3 Difference with Other Methods

Chu and Keerthi (2005) proposed two SVM based models for ordinal regression. Both methods introduce $k-1$ thresholds with k the number of ordinal levels in the data. As with SVM classifiers, the margin between two ordinal levels is set to $\frac{2}{||w||_2}$. In their first method (EXC) a data point X_i belonging to class Y_i has two slack variables: one relating to the threshold between classes k_i-1 and k_i and a second relating to the threshold between classes k_i and k_i+1. To ensure that the threshold between classes k_i-1 and k_i is smaller than the threshold between classes k_i and k_i+1, $k-1$ additional constraints are explicitly included. The problem can be written as:

$$\begin{aligned}
\min_{w,e,e^*,v} \quad & \|w\|_2 + \gamma \textstyle\sum_{i=1}^{n}(e_i+e_i^*) \\
\text{s.t.} \quad & \begin{cases}
w^T X_i - v_j + e_i \geq 1 & \forall\, i=1,\ldots,n;\ j=\arg\max_j(T_i > v_j) \\
-w^T X_i + v_j + e_i^* \geq 1 & \forall\, i=1,\ldots,n;\ j=\arg\min_j(T_i < v_j) \\
e_i \geq 0 & \forall\, i=1,\ldots,n \\
e_i^* \geq 0 & \forall\, i=1,\ldots,n \\
v_j \leq v_{j+1} & \forall\, j=1,\ldots,k-2\,.
\end{cases}
\end{aligned}$$

In their second approach (IMC) the constraints on the thresholds are added implicitly by adding $k-1$ slack variables, one for each threshold, for every datapoint. The problem can be formulated as follows:

$$\min_{w,e,e^*,v} \quad \|w\|_2 + \gamma \sum_{i=1}^{n} (e_i + e_i^*)$$
$$\text{s.t.} \quad \begin{cases} w^T X_i - v_j + e_i \geq 1 & \forall\, i = 1,\ldots,n;\ \forall\, j:\ T_i > v_j \\ -w^T X_i + v_j + e_i^* \geq 1 & \forall\, i = 1,\ldots,n;\ \forall\, j:\ T_i < v_j \\ e_i \geq 0 & \forall\, i = 1,\ldots,n \\ e_i^* \geq 0 & \forall\, i = 1,\ldots,n\,. \end{cases}$$

In our method, we adopt the approach of the EXC method concerning slack variables, the method differing in the definition of the margin. Instead of defining an equal margin at every border, the margin between classes k_i and $k_i + 1$ is defined as $\frac{|Y_{(i+1)} - Y_{(i)}|}{||w||_2}$.

Remark the similarity between these models and the standard SVMs (Vapnik, 1998) in the binary classification problem (with two classes $\mathcal{C}_1$ and $\mathcal{C}_2$):

$$\min_{w,e,e^*,b} \quad \|w\|_2 + \gamma \sum_{i=1}^{n} (e_i + e_i^*)$$
$$\text{s.t.} \quad \begin{cases} w^T X_i + b + e_i \geq 1 & \forall\, i \in \mathcal{C}_1 \\ -w^T X_i - b + e_i^* \geq 1 & \forall\, i \in \mathcal{C}_2 \\ e_i \geq 0 & \forall\, i \in \mathcal{C}_1 \\ e_i^* \geq 0 & \forall\, i \in \mathcal{C}_2\,. \end{cases} \tag{12}$$

In case $k = 2$, both EXC and IMC reduce to the model:

$$\min_{w,e,e^*,v} \quad \|w\|_2 + \gamma \sum_{i=1}^{n} (e_i + e_i^*)$$
$$\text{s.t.} \quad \begin{cases} w^T X_i - v + e_i \geq 1 & \forall\, i \in \mathcal{C}_1 \\ -w^T X_i + v + e_i^* \geq 1 & \forall\, i \in \mathcal{C}_2 \\ e_i \geq 0 & \forall\, i \in \mathcal{C}_1 \\ e_i^* \geq 0 & \forall\, i \in \mathcal{C}_2\,, \end{cases} \tag{13}$$

which equals the model in Equation (12) when the threshold v (note that there is only one threshold in this case) is considered as the constant term. The MINLIP model reduces to:

$$\min_{w,e,e^*,v} \quad \|w\|_2 + \gamma \sum_{i=1}^{n} (e_i + e_i^*)$$
$$\text{s.t.} \quad \begin{cases} w^T X_i - v + e_i \geq Y_i - B & \forall\, i \in \mathcal{C}_1 \\ -w^T X_i + v + e_i^* \geq B - Y_i & \forall\, i \in \mathcal{C}_2 \\ e_i \geq 0 & \forall\, i \in \mathcal{C}_1 \\ e_i^* \geq 0 & \forall\, i \in \mathcal{C}_2\,, \end{cases} \tag{14}$$

where only one dummy observation (v, B) needs to be introduced. The difference between Equations (12, 13) and Equation (14) lies in the right hand side of the two first inequalities, which is a consequence of the used complexity control. Models (13) and (14) are equivalent up to the choice of the regularization constant.

Chu and Ghahramani (2005) proposed a probabilistic approach to ordinal regression in Gaussian processes (GPOR). They impose a Gaussian process prior distribution on the utility function (called latent function in their work) and employ an appropriate likelihood function for ordinal variables.

Experiments will compare our methods with the Bayesian inference technique of MacKay (1992), using the Laplacian approximation to implement model adaptation. The GPOR approach differs from ours since it uses a Bayesian framework.

5. Transformation Models for Failure Time Data

We now turn our attention to the case where the data originate from a survival study, that is, the dependent variable is essentially a time-to-failure and typically requires specific models and tools to capture its behavior. We will adopt a classical statistical setup, and will show how the techniques as described in Section 3 provide a powerful alternative to the classical statistical (semi-parametric) toolbox.

5.1 Survival Data

The observations are assumed to fit in the following statistical setup, see, for example, Kalbfleisch and Prentice (2002) for a more elaborate introduction. Let $T \in \mathbb{R}^+$ and $X \in \mathbb{R}^d$ be a random variable and random vector respectively, jointly following a probability law characterized by P as classical. The former variable T describes the *time to the event* of interest, and the random vector X taking values in $\mathbb{R}^d$ describes d covariates. Note that in this Section T has the same role as Y in the previous Sections. We assume that no ties will occur in the data in order to keep the explanations as simple as possible. We will consider predictive models where the covariates come in through a linear combination with weights $w \in \mathbb{R}^d$ as before, or $\mathcal{U} = \left\{u : \mathbb{R}^d \to \mathbb{R} : u(X) = w^T X,\ \forall\, X \in \mathbb{R}^d\right\}$. A key quantity in survival analysis is the *conditional survival function* $S(t|u(X)) : \mathbb{R}^+ \to [0,1]$ defined as

$$S(t|u(X)) = P\left(T > t \middle|\ u(X)\right),$$

denoting the probability of the event occurring past t given the value of the utility function $u(X) = w^T X$. A related quantity to the conditional survival function is the *conditional hazard function* $\lambda : \mathbb{R} \to \mathbb{R}^+$ defined as

$$\begin{aligned}\lambda(t|u(X)) &= \lim_{\Delta t \to 0} \frac{P\left(t \le T < t + \Delta t \middle| u(X), T \ge t\right)}{\Delta t} \\ &= \lim_{\Delta t \to 0} \frac{P\left(t \le T < t + \Delta t \middle| u(X)\right)}{S\left(t \middle|\ u(X)\right)}.\end{aligned}$$

If the derivative $s : \mathbb{R}^+ \to \mathbb{R}$ with $s(t|u(X)) = \dfrac{\partial S(t|u(X))}{\partial t}$ exists, one can write $\lambda(t|u(X)) = \frac{s(t|u(X))}{S(t|u(X)}$. The conditional hazard function reflects the instantaneous probability that the event will occur given that the subject already survived beyond time t. Finally, one can make the relation between the hazard λ and the survival function S even more explicit by introducing the *conditional cumulative hazard function* $\Lambda(t|u(X)) = \int_0^t \lambda(r|u(X))dr$ for $t \ge 0$ such that

$$\Lambda(t|u(X)) = -\ln\left(S(t \mid u(X))\right).$$

The following Subsection enumerates some commonly used (semi-)parametric methods for modelling the survival and hazard functions.

5.2 Transformation Models for Survival Analysis

The Transformation model (see Definition 1) encompasses a broad class of models, including the following classical survival models.

1. Cox' *proportional hazard model* is recovered when one defines $g = h^{-1}$ (if it exists) as $g(z) = \ln(-\ln(z))$. Under the Cox model, the value of the survival function at $t = T$ is

$$S(T,X) = [S_0(T)]^{\exp(-\beta^T X)},$$

where $S_0(t) = \exp(-\Lambda_0(t))$ is called the baseline survival function. Taking $\ln(-\ln(\cdot))$ of both sides in (1) leads to

$$\begin{aligned} \ln(-\ln(S(T,X)) &= \ln(-\ln(S_0(T))) - \beta^T X \\ &= \ln(\Lambda_0(T)) - \beta^T X \\ \Rightarrow\ \varepsilon &= g(T) - u(X) \\ \Rightarrow T &= h(u(X) + \varepsilon). \end{aligned}$$

Remark that the last transition is only possible if $g(t)$ is invertible. The 'noise terms' are i.i.d. observations from the extreme value distribution $F_\varepsilon(z) = 1 - \exp(-\exp(z))$.

2. The *proportional odds model* is defined as

$$\ln\left(\frac{F(t|X)}{1-F(t|X)}\right) = \alpha(t) + \beta^T X, \tag{15}$$

with $F(t|X)$ the conditional cumulative distribution function and $\alpha(t)$ a monotonically increasing function. In general the survival function equals $S(t) = 1 - F(t)$, leading together with Equation (15) to

$$\begin{aligned} \ln\left(\frac{1-S(T|X)}{S(T|X)}\right) &= \alpha(T) + \beta^T X \\ \Rightarrow\ \varepsilon &= \alpha(T) + u(X) \\ \Rightarrow T &= h(-u(X) + \varepsilon). \end{aligned}$$

Remark that the last transition is only possible if $\alpha(T)$ is invertible.

3. The *accelerated failure time (AFT)* is given when $h(z) = \ln(z)$.

For an extended discussion on the use of the class of transformation models and specific parameterizations of the functions h or g, see, for example, Dabrowska and Doksum (1988), Koenker and Geling (2001), Cheng, Wei, and Ying (1997) and citations.

5.3 Censoring

A typical property of failure time data is the occurrence of censoring. A failure time is called censored when the exact time of failure is not observed. Despite this, censored times do provide relevant information. Define $\mathcal{T}_i = (T_i, \delta_i)$ with δ_i the censoring indicator, capturing all censoring

information: $\delta = 0$ indicates the occurrence of an event at a known failure time (uncensored data point); right, left and interval censoring are indicated by $\delta = 1, \delta = 2$ and $\delta = 3$ respectively. Without censoring all possible pairs of datapoints $\{(\mathcal{T}_i, \mathcal{T}_j)\}_{i \neq j}$ can be used for comparison in Equation (5). The presence of censoring leads to a lack of comparability between certain data points. Let $\Delta(\mathcal{T}_i, \mathcal{T}_j)$ be a comparability indicator, indicating whether the datapoints i and j are comparable:

$$\Delta(\mathcal{T}_i, \mathcal{T}_j) = \begin{cases} 0 \text{ if } \mathcal{T}_i \text{ and } \mathcal{T}_j \text{ are not comparable} \\ 1 \text{ if } \mathcal{T}_i \text{ and } \mathcal{T}_j \text{ are comparable}. \end{cases}$$

This indicator is defined depending on the censoring types present in the data:

Right censoring occurs when the event of interest did not occur until the last follow-up time. This type of censoring typically occurs at the end of the study period. Although the exact failure time is not known in this case, the failure time is known to be later than the date of last follow-up. In case of right censoring the comparability indicator Δ takes the value 1 for two observations i and j when the observation with the earliest failure time is observed, and zero otherwise:

$$\Delta(\mathcal{T}_i, \mathcal{T}_j) = \begin{cases} 1 & \text{if } (T_i < T_j \text{ and } \delta_i = 0) \text{ or } (T_j < T_i \text{ and } \delta_j = 0) \\ 0 & \text{otherwise.} \end{cases}$$

Left censoring deals with the case when the failure is known to have happened before a certain time. An example of left censoring arises in case a variable can only be measured when its value is above a certain level. For left censoring, two observations i and j are comparable when the observation with the highest failure time is non-censored and zero otherwise:

$$\Delta(\mathcal{T}_i, \mathcal{T}_j) = \begin{cases} 1 & \text{if } (T_i < T_j \text{ and } \delta_j = 0) \text{ or } (T_j < T_i \text{ and } \delta_i = 0) \\ 0 & \text{otherwise.} \end{cases}$$

Interval censoring is a combination of the previous two censoring types. In this case the failure time is not known exactly, instead an interval including the failure time is indicated. This type of censoring is often found in medical studies where the patients are subject to regular check up times (Finkelstein, 1986). Whether two observations are comparable or not in case of interval censoring depends on the censoring times $\underline{T}_i$ and $\overline{T}_i$ defining the failure interval for each observation i: $T_i \in [\underline{T}_i, \overline{T}_i]$. For uncensored observations, the failure interval reduces to one time, namely the failure time $T_i = \underline{T}_i = \overline{T}_i$. The comparability indicator is defined as:

$$\Delta(\mathcal{T}_i, \mathcal{T}_j) = \begin{cases} 1 & \text{if } \overline{T}_i < \underline{T}_j \text{ or } \overline{T}_j < \underline{T}_i \\ 0 & \text{otherwise.} \end{cases}$$

In case the data consists of data points with different types of censoring, the comparability indicator is defined as follows. In the most general case, the failure time T_i is considered to be an element of the interval $[\underline{T}_i, \overline{T}_i]$. For right censored data points the right edge of the interval equals infinity, whereas for left censored observation the left edge of the interval equals zero. The comparability indicator is then defined as:

$$\Delta(\mathcal{T}_i, \mathcal{T}_j) = \begin{cases} 1 & \text{if } \overline{T}_i < \underline{T}_j \text{ or } \overline{T}_j < \underline{T}_i \\ 0 & \text{otherwise.} \end{cases}$$

More information on censoring can be found in Andersen et al. (1993), Elandt-Johnson and Johnson (1980), Harrell (2001), Kalbfleisch and Prentice (2002) and Miller (1981).

Standard statistical methods for modelling survival data obtain parameter estimates by maximizing a (partial) likelihood with regard to these parameters. This likelihood depends on the ranking of the failure times. In the presence of right censoring, this ranking can uniquely be defined and estimates for the parameters can be obtained. However, in the presence of interval censoring, a unique ranking of the failure of all instances is not always possible. Peto (1972) and Satten (1996) among others, suggested extensions of the proportional hazard model where censoring is not restricted to right censoring. However, estimation of the parameters in these cases remain difficult. In the next section we will illustrate that MINLIP can be easily adapted for right, left, interval censoring and combined censoring schemes. However, we first need an appropriate measure of concordance equivalent to Equation (3). Therefore, we resort to the concordance index as described by Harrell et al. (1984) and Harrell (2001).

Definition 4 (Concordance Index) *The concordance index (c-index) is a measure of association between the predicted and observed failures in case of censored data. The c-index equals the ratio of concordant to comparable pairs of data points. Two observations i and j are comparable if their relative order in survival time is known. A pair of observations i and j is concordant if they are comparable and the observation with the lowest failure time also has the lowest score for the utility function $u(X)$. Formally, the observation based c-index of a model generating predictions $u(X_i)$ for data X_i from a data set $\mathcal{D} = \{(X_i, Y_i, \delta_i)\}_{i=1}^n$ can be expressed as*

$$\mathcal{C}_n(u) \quad = \quad \frac{\sum_{i \neq j} \Delta(\mathcal{T}_i, \mathcal{T}_j) I[(u(X_j) - u(X_i))(Y_j - Y_i) > 0]}{\sum_{i \neq j} \Delta(\mathcal{T}_i, \mathcal{T}_j)}.$$

This index is an estimate probability of concordance between predicted and observed survival, with c-index $= 0.5$ for random predictions and c-index $= 1$ for a perfectly concordant model. Without censoring, this definition is exactly equal to the concordance as defined in Equation (2).

5.4 Modifications to MINLIP

This section describes how the standard MINLIP model can be extended towards failure time data including the handling of censored data. Therefore, Equation (8) is adapted to include censored data. In particular, the matrix $\mathbf{D}$ needs to be changed in order to allow for pairs of data points not to be comparable. Let $\mathbf{R} \in \mathbb{R}^{(n-1)\times(n-1)}$ be defined as the diagonal matrix with $\mathbf{R}_{ii} = \Delta(Z_i, Z_{i+1}), \forall\, i = 1, \ldots, n-1$. The matrix $\mathbf{D}$, representing the datapoints to be compared, is adapted for censoring according to:

$$\mathbf{D}_c = \mathbf{RD},$$

where Δ is defined as in Section 5.3, resulting in multiple rows with only zero entries in the matrix $\mathbf{D}_c$. For computational convenience these rows can be left out. It is seen that issues concerning the

type(s) of censoring in the data are easily dealt with by using the comparability indicator. In the remainder of this paper we will restrict our attention to right censored data.

The learning objective can now be formalized as

$$\begin{array}{ll} \min\limits_{w,e} & \frac{1}{2}w^T w + \gamma \|e\|_1 \\ \text{s.t.} & \mathbf{D}_c(\Phi w + e) \geq \mathbf{D}_c \mathbf{T}, \end{array} \tag{16}$$

where $\|e\|_1 = \sum_{i=1}^n |e_i|$ and $\mathbf{T} = [T_{(1)}, T_{(2)}, \ldots, T_{(n)}]^T$ is a vector containing all failure times, censored or not. As in Section 3, the dual of this optimization problem becomes

$$\begin{array}{ll} \min\limits_{\alpha} & \frac{1}{2}\alpha^T \mathbf{D}_c \mathbf{K} \mathbf{D}_c^T \alpha - \alpha^T \mathbf{D}_c \mathbf{T} \\ \text{s.t.} & \begin{cases} -\gamma 1_n \leq \mathbf{D}_c^T \alpha \leq \gamma 1_n \\ \alpha \geq 0_{n-1}, \end{cases} \end{array}$$

Given the solution $\hat{\alpha}$, the predicted utility can be calculated for a new point X^* as

$$u(X^*) = \hat{\alpha}^T \mathbf{D}_c \mathbf{K}_n(X^*),$$

with $\mathbf{K}_n(X^*) = [K(X^*, X_1) \ \ldots \ K(X^*, X_n)]^T \in \mathbb{R}^n$. Since the censoring mechanism can be handled by a proper choice of $\mathbf{D}_c$, it is not too difficult to extend the formulations of Subsection 3.5 as well.

5.5 Prediction with Transformation Models

The prediction step in survival analysis, refers to the estimation of survival and hazard functions rather than the estimation of the failure time itself. The proportional hazard model estimates these functions, by assuming that a baseline hazard function exists; the covariates changing the hazard only proportionally. The baseline hazard function is estimated using the Breslow estimator of the cumulative baseline hazard (Breslow, 1974).

In our setting, the cumulative distribution function (cdf), can be estimated, after estimation of the utility, as follows. The time axis is divided in k equidistant time intervals $[t_{l-1}, t_l], \forall\, l = 2, \ldots, k$. For each observation in the set $\{u_i, T_i, \delta_i\}_{i=1}^n$, the outcome in each time interval is defined as:

$$Y_{il} = \begin{cases} 0 \text{ if } T_i > t_l \\ 1 \text{ if } T_i \leq t_l \text{ and } \delta_i = 0. \end{cases}$$

Remark that censored observations are not considered at times later than the censoring time. Using a monotone least squares support vector regression model (Pelckmans et al., 2005a) with a Gaussian kernel, or another monotonic regression model, the utility and the time interval number l as inputs and Y_{il} as output, the cdf $\hat{F}(u_i, l)$ is estimated. The survival function is found as $\hat{S}(u_i, l) = 1 - \hat{F}(u_i, l)$. The hazard function is then found as

$$\hat{\lambda}(u_i, l) = \frac{\hat{F}(u_i, l+1) - \hat{F}(u_i, l)}{t_{l+1} - t_l}, \forall\, l = 1, \ldots, k-1.$$

Remark the analogy with the partial logistic artificial neural network approach to the survival problem proposed by Biganzoli et al. (1998). However, since the latter can not be seen as a transformation model, data replication is necessary even when one is only interested in risk groups. Thanks to the two-step approach of transformation models, data replication can be avoided.

6. Application Studies

This final Section describes some experiments to illustrate the use of the presented method. In a first Subsection, 3 artificial examples will illustrate how transformation models are used within the ranking, ordinal regression and survival setting (see also Table 1). A first real life application illustrates the use of MINLIP for ordinal data. We use 6 benchmark data sets and compare the performance of MINLIP with EXC and IMC as proposed in Chu and Keerthi (2005) and GPOR as proposed in Chu and Ghahramani (2005). The last two examples concern survival data, one with micro-array data (data also used in Bøvelstad et al. 2007) and one with clinical data (Schumacher et al., 1994).

Unless stated otherwise, 10-fold cross-validation was used for model selection. For every kernel and regularization parameter to be tuned a grid of values was searched and the combination of parameter values yielding the lowest cross-validation error or highest cross-validation performance was selected. In the first example the mean absolute error between predicted and observed output levels was used as model selection criterion since prediction is relevant in this case. For both survival examples, the cross-validation concordance index was used as model selection criterion since the main interest lies in the ranking of the patients.

6.1 Artificial Examples

This section illustrates the different steps needed to obtain the desired output for ranking, regression and survival problems, using artificial examples. Together with Table 1, this Subsection illustrates the different tasks considered in the paper.

6.1.1 RANKING

In this first example, we consider the ranks of 150 cyclists in 9 different races. Using the ranks of 100 out of these cyclists in a 10^{th} race , we want to predict the rank of the remaining 50. Additional information includes: age, weight and condition score. The outcome is defined as the ranking given by a weighted sum of the ranking in the 9 races, age, weight and condition score. Weights are drawn from a uniform distribution on the unit interval. The ranking on the previous races are numbers from 1 to 100, all other variables are drawn from a standard normal distribution.

A first step in all transformation models is to estimate the utility function u. Using Equation (8) with a linear kernel and 5-fold cross validation with the concordance index (ranking criterion) as a model selection criterion, a concordance index of 0.98 on the test set was obtained. The predicted ranking corresponds very well with the observed ranking (see Figure 6). Since one is only interested in ranking the cyclists, the value of the utility is irrelevant. Additionally, one is not interested in estimating the transformation function h.

6.1.2 ORDINAL REGRESSION

In a second artificial example, consider the scenario in which one wishes to divide students into 3 groups: bad, average and good student. For this task, the grades on 10 courses for 150 students are available. The outcome depends on the average grade. A bad, average and good student has an average grade below 55%, between 55% and 65% and above 65% respectively. The results on 100 students will be used for training, the remaining students will be used for testing.

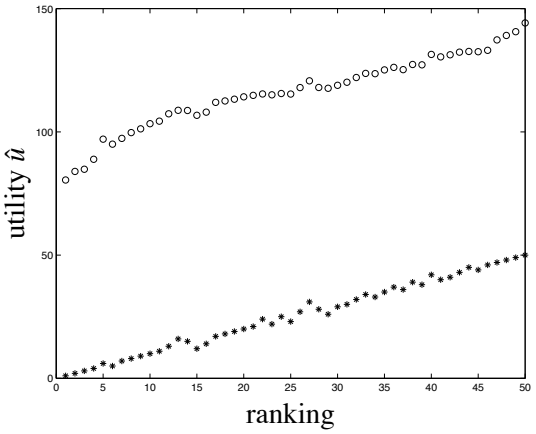


Figure 6: Artificial example illustrating the use of the MINLIP transformation model for the ranking setting. The estimated utility of the test observations are denoted by the circles. The value of the utility does not correspond to the ranking. However, the rank of the estimated utility (denoted by stars) are a good prediction of the observed rank.

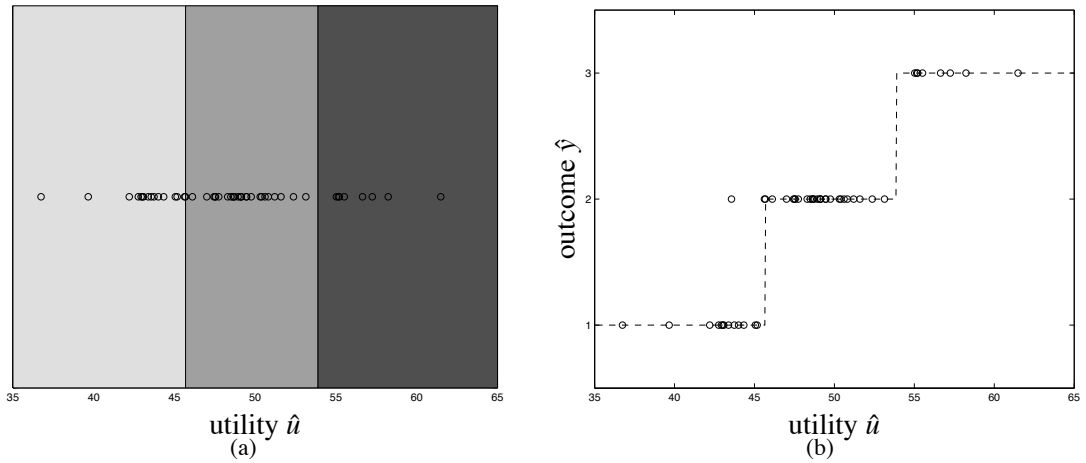


Figure 7: Artificial example illustrating the use of the MINLIP transformation model for ordinal regression. (a) The estimated utility of the test observations are denoted by the circles. The MINLIP model for ordinal regression results in an estimate of the utility function and threshold values. Students with the lowest utility (less than the first threshold) are predicted to be bad students (light grey). Students with a utility between both thresholds (medium grey) are estimated to be average students and students with a utility higher than the second threshold (dark grey) are predicted to be good students. (b) Illustration of the transformation function h (dashed line).

In a first step, all students are ranked according to the available data, namely their grades on 10 courses. Using Equation (11) with the concordance index as model selection criterion and a linear kernel, a concordance of 0.99 between the utility and the outcome on the test set is obtained. In addition to an estimate of the utility, the MINLIP model for ordinal regression gives threshold values which can be used to predict the outcome of new observations (see Figure 7). Since, the MINLIP model generates these thresholds, no additional model is needed to obtain the transformation function.

6.1.3 SURVIVAL ANALYSIS

In a last artificial example, a randomized trial is simulated. Assume 150 patients are randomly divided into 2 treatment groups. Additionally, the age (drawn from a standard normal distribution) of the patients is known. The survival time is known for 100 patients. For the first treatment arm, the survival time has a Weibull distribution with parameters 1 and 0.5. For the second treatment arm, the survival time is Weibull distributed with parameters 4 and 5. Using the information on age, treatment arm and survival on 100 patients, one would like to predict the survival for the remaining 50 patients. Assuming that the age is irrelevant for the survival, the treatment will be the only important factor in predicting the patients' survival.

As with the two previous examples, the MINLIP model is used to estimate the utility of the patients. Using a linear kernel and 5-fold cross validation in comparison with the concordance index as model selection criterion, a c-index of 0.70 is obtained on the test set. Figure 8 illustrates that the MINLIP model is able to divide the group of test patients into two groups with a significant different survival (p=0.03, logrank test). The first part of the transformation model obtained a nice result. However, in survival analysis, additional information can be provided when performing the second part of the transformation model, namely estimating the transformation function. Applying the method as explained in Section 5.5, the estimated survival curves for all patients are calculated (Figure 9). One clearly notices two distinct survival groups. The grey and black survival curves correspond to patients in the first and second treatment arm, respectively. The true survival function for the first and second treatment are illustrated in thick black and grey lines, respectively.

6.2 Ordinal Regression

At first 6 regression data sets[3] were converted to ordinal regression data sets as follows. The data sets were divided into 20 folds with 10 equal-frequency bins. The output value for each bin was set to the average output within the bin. The performance of the MINLIP model was compared with two methods described in Chu and Keerthi (2005) (see Table 2). Both of these methods optimize multiple thresholds to define parallel discriminant hyperplanes for the ordinal levels. The first method (EXC) explicitly imposes the ordering of the thresholds, whereas this is done only implicitly in the second method (IMC). Tuning of the Gaussian kernel parameter and the regularization parameter was performed with 10-fold cross-validation on an exponential grid using mean absolute error as model selection criterion. After an initial search, a finer search was performed in the neighborhood of the initial optimum. Results of the GPOR method are reported for comparison. The GPOR has a lower mean zero-one error on small data sets. For larger data sets and for mean absolute errors, it performs less. The IMC method has the disadvantage that large QP problems need to be solved for

3. Data are available at `http://www.liacc.up.pu/~ltorgo/Regression/DataSets.html`.

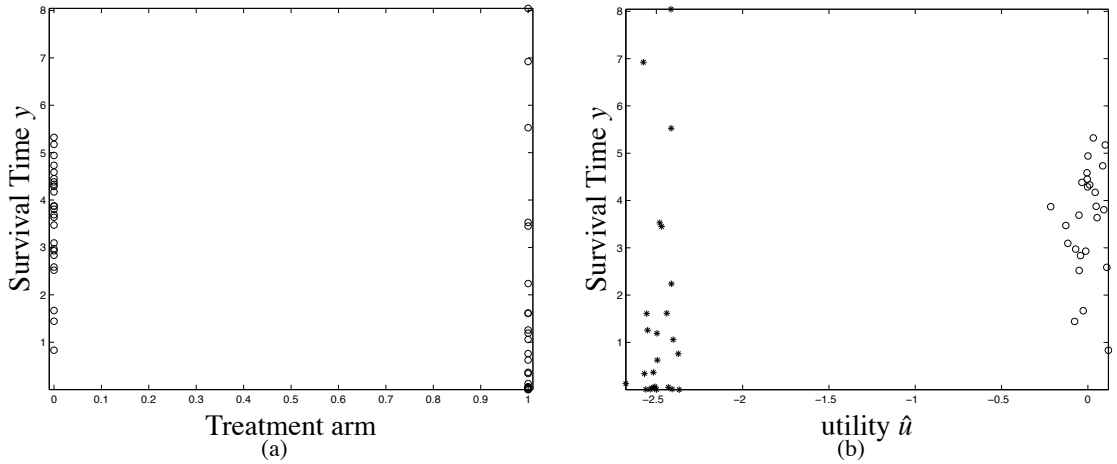


Figure 8: Artificial example illustrating the use of the MINLIP transformation model for survival analysis. (a) Survival time as a function of the treatment arm. Patients receiving the first treatment survive longer in general. The second treatment results in lower survival times. However, some patients have extremely large survival times. (b) Survival time versus estimated utility. The circles and the stars denote the first and second treatment arm, respectively. The utility is able to group the patients according to the relevant variable treatment (see the clear separation between circles and stars).

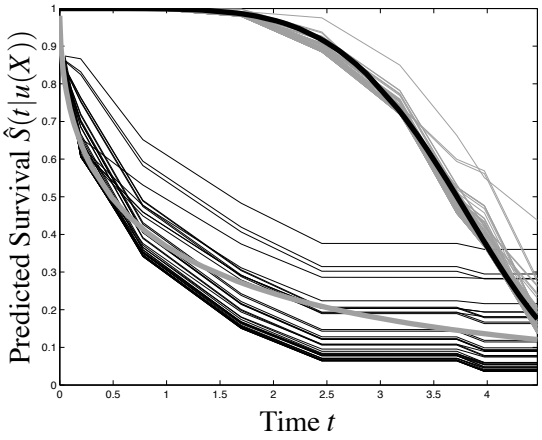


Figure 9: Artificial example illustrating the use of the MINLIP transformation model for survival analysis: illustration of the reconstruction step. For each patient, the survival curve is calculated using the method discussed in Section 5.5. Grey and black curves represent patients from the first and second treatment arm, respectively. The true survival curve for the first and second treatment, are illustrated in thick black and grey lines. One clearly notices two distinct survival groups, corresponding to the treatment groups.

data set	mean zero-one error			
	minlip	exc	imc	gpor
pyrimidines	0.74±0.07	0.79±0.08	0.75±0.09	**0.73±0.09**
triazines	**0.86±0.05**	**0.86±0.04**	0.87±0.04	**0.86±0.03**
Wisconsin	0.89±0.02	0.89±0.03	0.89±0.02	**0.85±0.03*****
machine CPU	0.65±0.04	0.65±0.06	**0.64±0.04**	0.80±0.09***
auto MPG	**0.58±0.03**	**0.58±0.03**	/	0.68±0.08***
Boston housing	**0.57±0.04**	**0.57±0.04**	/	0.61±0.03***
data set	mean absolute error			
	minlip	exc	imc	gpor
pyrimidines	**0.05±0.01**	0.06±0.01	**0.05±0.01**	0.06±0.01
triazines	0.12±0.06	**0.10±0.01***	**0.10±0.00*****	0.16±0.02***
Wisconsin	29.56±2.18	28.80±1.17	**28.23±1.28**	34.52±4.27***
machine CPU	**29.41±4.27**	30.57±6.94	30.48±4.10	157.00±124.67***
auto MPG	2.08±0.16	**2.05±0.18**	/	4.13±1.13***
Boston housing	2.49±0.27	**2.33±0.25***	/	2.97±0.29***

Table 2: Test results of MINLIP, EXC and IMC using a Gaussian kernel and GPOR. The targets of the data sets were discretized by 10 equal-frequency bins. The output value for each bin is set to the average output within the bin. The results are averaged over 20 trials. The best performing model is indicated in bold. Significant differences as calculated by Wilcoxon's signed rank sum test between the EXC, IMC, GPOR and the MINLIP (reference) model are indicated with * ($p < 0.05$), ** ($p < 0.01$) or *** ($p < 0.001$).

growing training samples, requiring more computational time. The MINLIP method makes a nice trade-off between computational time and performance.

6.3 Failure Time Data: Micro-array Studies

The MINLIP technique is derived from machine learning techniques as SVMs, techniques which are shown to be especially useful to handle high-dimensional data sets. We therefore test the performance of MINLIP on 3 micro-array data sets.

In this example we compare the performance of model (16) (MINLIP) and linear extension as in Equation (10) (MINLIP$_{L1}$) with 5 other methods which are discussed and implemented by Bøvelstad et al. (2007): principal components regression (PCR), supervised principal components regression (SPCR), partial least squares regression (PLS) and two penalized Cox regression models (Cox, 1972): ridge regression (Cox$_{L2}$) and $L1$ regularization (Cox$_{L1}$). The PCR method uses principal component analysis to select n_λ principal components which account for as much variation in the gene expression profiles as possible. The selected principal components are then used as covariates in a Cox regression model (see Hastie, Tibshirani, and Friedman, 2001, Chapter 3.4.4). The SPCR (Bair and Tibshirani, 2004; Bair, Hastie, Debashis, and Tibshirani, 2006) method first selects a subset of genes which are correlated with survival by using univariate selection and then applies PCR to this subset. The standard PLS method performs regression of the outcome using n_λ components which are a linear combination of the original covariates (Martens and Næs, 1989). The application of PLS

to the Cox model is not straightforward since the PLS algorithm assumes a linear relation between outcome and covariates. See Nygård et al. (2008) for a detailed description of the method.

Three different micro-array data sets are used in this experiment:

The Dutch Breast Cancer Data (DBCD) from van Houwelingen et al. (2006) is a subset of the data from van de Vijver et al. (2002) and contains information on 4919 gene expression levels of a consecutive series of 295 women with breast cancer from the fresh-frozen-tissue bank of the Netherlands Cancer Institute. All 295 tumors were primary invasive breast carcinoma less than 5 cm in diameter. The women were 52 years or younger. The diagnosis was made between 1984 and 1995 and there was no previous history of cancer, except non-melanoma skin cancer. In 79 (26.78%) patients distant metastases were noted within the study period. The median follow-up was 6.7 years (range 0.05-18.3).

The DLBCL data from Rosenwald et al. (2002) contains data on 240 patients with diffuse large-B-cell lymphoma. The data consist of 7399 gene expression measurements. The median follow-up time was 2.8 years and 58% of the patients died during the study period.

The Norway/Stanford breast cancer data (NSBCD) from Sørlie et al. (2003) contains gene expression measurements from 115 women with breast cancer. 549 intrinsic genes introduced in Sørlie et al. (2003) were used. Missing values were previously imputed using 10-nearest neighbor imputation (Bøvelstad et al., 2007). 38 (33%) patients experienced an event.

Figure 10 summarizes performances $\mathcal{C}_n^u$ on all methods for 100 different randomizations between training and test sets (2/3 training, 1/3 test). In the right panels of Figure 10 the time dependent receiver operator characteristics (TDROC) (Heagerty, Lumley, and Pepe, 2000) are shown. The left panel illustrates the concordance index. The performance of the MINLIP model is better or comparable to the best of the other tested models.

6.4 Failure Time Data: Cancer Study

In this last example, we investigate the ability of the MINLIP model to estimate how the different covariates influence the survival time. We use the German Breast Cancer Study Group data[4] (Schumacher et al., 1994), containing information on 686 patients and 8 variables. Available variables are: hormonal treatment, age, menopausal status, tumor size, tumor grade, the number of positive lymph nodes, the progesterone receptor (fmol) and the estrogen receptor (fmol). 299 (43.6%) patients had a breast cancer related event within the study period, leaving all other patients with a right censored failure time. The data set was randomly divided in training and test set (2/3 versus 1/3).

Since medical data are typically not highly non-linear, we use a componentwise polynomial kernel

$$K(X,Z) = \sum_{p=1}^{d} (\tau + X^{pT} Z^p)^2,\ \tau \geq 0,$$

with d the number of variables and X^p the p^{th} covariate, to model non-linearities. Model selection is done by 10-fold cross-validation with the concordance index as model selection criterion.

4. Data can be found at `http://www.blackwellpublishers.com/rss/Volumes/A162p1.htm`.

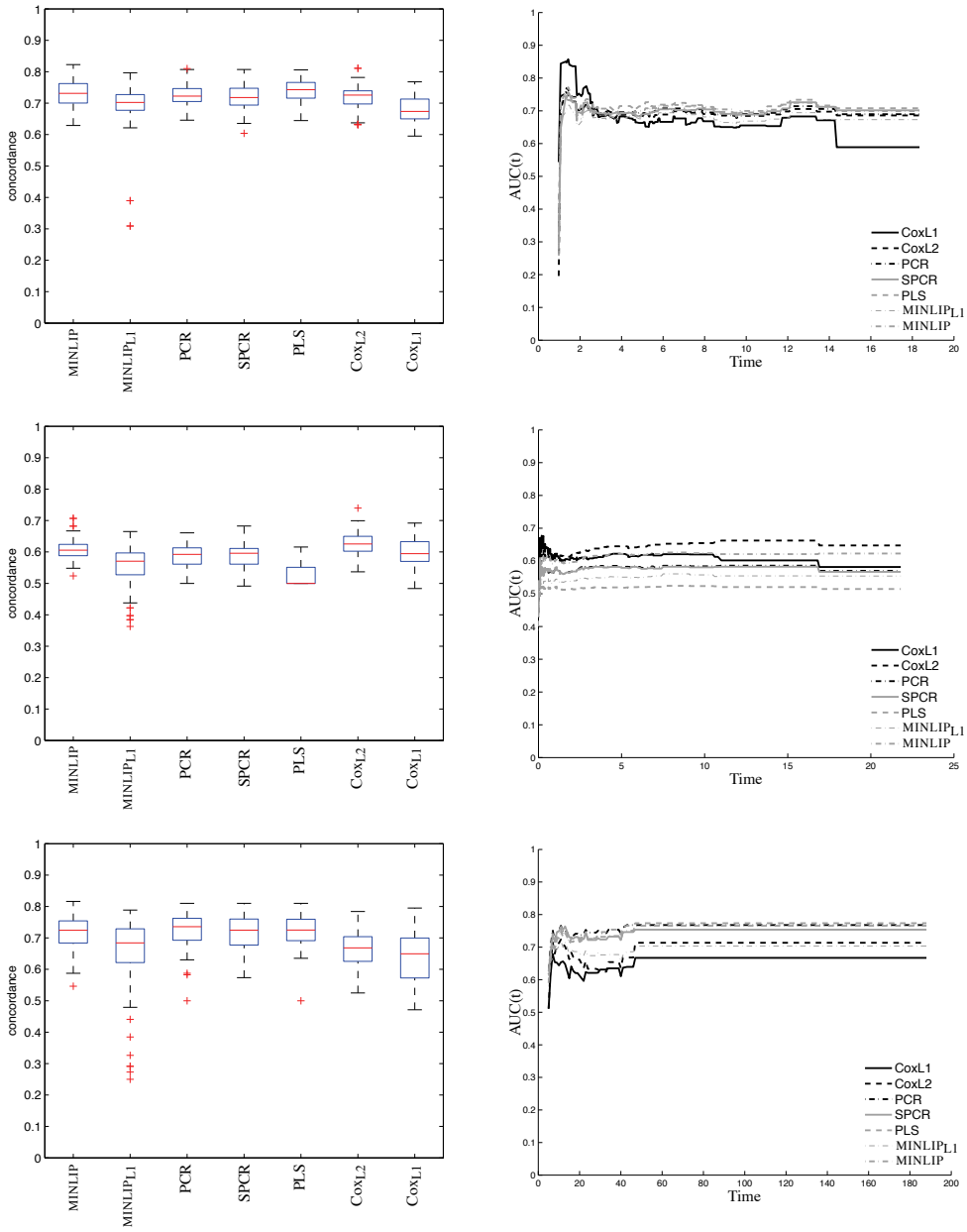


Figure 10: Concordance (left) and time dependent receiver operating characteristic curve (TDROC) (right) on the test set for three micro-array survival data sets (top: DBCD, middle: DL-BCL, bottom: NSBCD). The MINLIP model obtains a performance which is slightly higher or comparable to the other tested models.

We compare our results with Cox' proportional hazard model. However, the standard Cox model (Cox, 1972) assumes linearity in the covariates, implying that for a continuous variable as age for example, the risk ratio between two patients aged 45 and 50 is the same as the risk ratio between two patients aged 75 and 80. To allow for non-linearities in the effects of the covariates on the hazard, the functional forms of the covariates were estimated using penalized smoothing splines (Eilers and Marx, 1996; Hurvich, Simonoff, and Tsai, 1998). In this method, a comparative small set of basis functions is fit and a likelihood penalizing the integrated second derivatives (see Therneau and Grambsch, 2000, Section 5.5) is used to estimate the coefficients. Akaike's information criterion (AIC=log likelihood - degrees of freedom) is used to select the degrees of freedom for each term.

Figures 11 and 12 show the estimated covariate effects for Cox regression with penalized splines and MINLIP, respectively. Remark that in Figure 11 the estimates are inversely related with the survival time, whereas in Figure 12 the estimates are related with the survival time itself. Cox' model predicts a decreasing risk for relapse for older patients, up to the age of 40, whereafter the risk increases slowly; for tumors up to 20mm the risk for relapse increases with size, with a threshold effect for larger tumors; the number of positive lymph nodes is inversely related with survival and larger values for the progesterone and estrogen receptors are related with longer survival. All conclusions of the covariate effects agree with what is known from literature (Fisher et al., 1983; Lamy et al., 2002; Pichon et al., 1980; Verschraegen et al., 2005). The MINLIP model estimates a higher survival time for older patients, up to the age of 65, whereafter the survival time drops again. According to this model, a larger tumor, a higher number of positive lymph nodes and a lower progesterone and estrogen receptor level result in lower survival times and thus a higher risk for relapse. Cox' model with penalized smoothing splines obtains a concordance on the test set equal to 0.6715, while the MINLIP model obtains a performance of 0.6857.

Figure 13 illustrates the ability of the models to generate prognostic indices. In clinical practice one is interested in groups of patients with low/high risk for the event to occur. Therefore the median value of the model output is used as a threshold to divide the test set into two groups: one group including patients with an estimated risk lower than the average and a second group with an estimated risk higher than the average. Kaplan-Meier curves and 95%-confidence intervals are plotted in Figure 13. The logrank test χ^2 value is 20.4281 and 29.6984 for Cox and MINLIP respectively. The latter method results in a better split between low and high risk patients.

7. Conclusions

This paper studied a machine learning approach for finding transformation models. Such models are found useful in a context of ordinal regression and survival analysis, and relate directly to commonly used risk measures as the area under the curve and others. The derivations go along the same lines as used for support vector machines, except for replacing the notion of (pairwise) margin with a Lipschitz smoothness condition on the transformation function. The presented learner finds a (non-linear) non-parametric transformation model by solving a convex Quadratic Program. Extensions towards tasks where transformation models provide only a (good) approximation (agnostic case), ordinal regression and survival analysis are given. Experiments on ordinal regression and survival analysis, on both clinical and high dimensional data sets, illustrate the use of the proposed method.

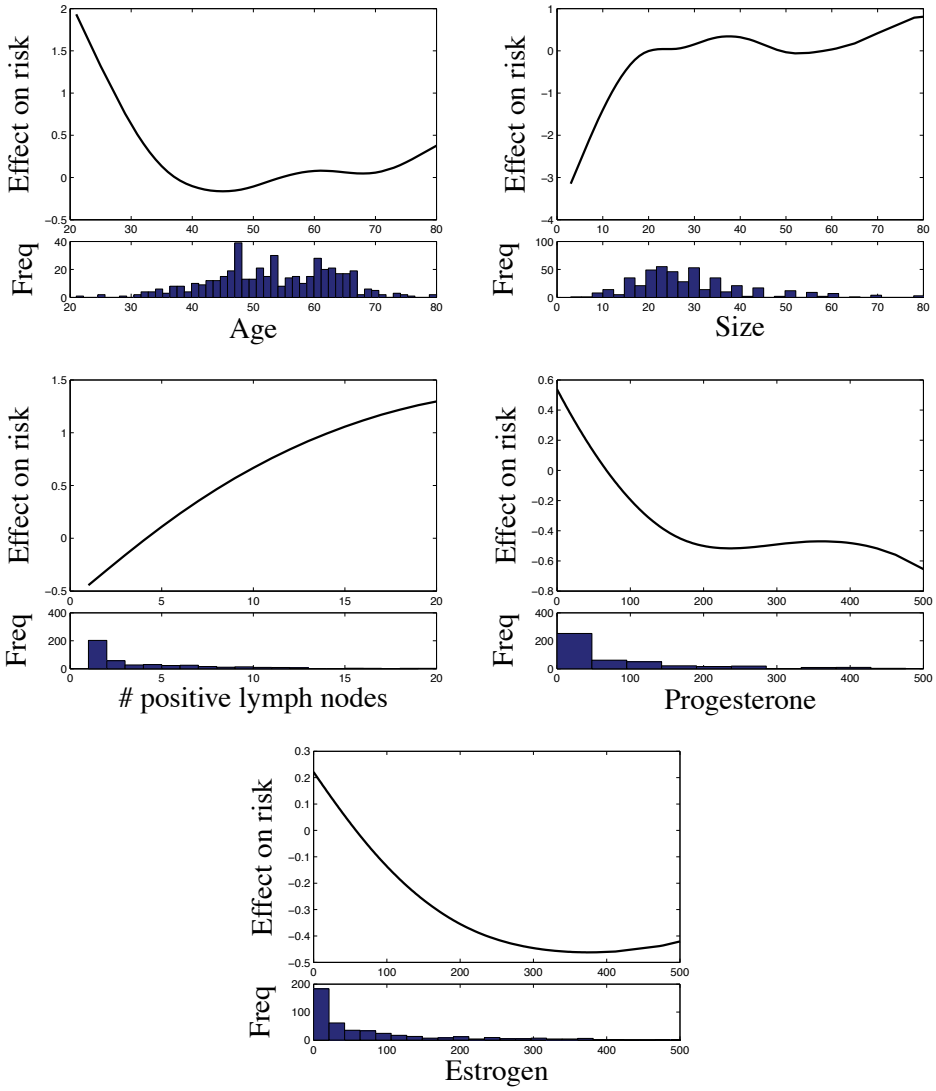


Figure 11: Estimation of the covariate effects on the risk of relapse (remark the difference with Figure 12) with smoothing splines within Cox' proportional hazard model and histograms of the variables. The estimated effects are inversely related with the survival time. The model estimates a lower chance for relapse for older patients up to the age of 40, whereafter the risk increases again, albeit slowly. The chance for relapse increases for larger tumors until a size of 20mm, whereafter the chance remains fairly constant. For common values of the number of positive lymph nodes and receptors, the risk increases for larger/lower values respectively. Conclusions drawn by the model agree with what is known from literature.

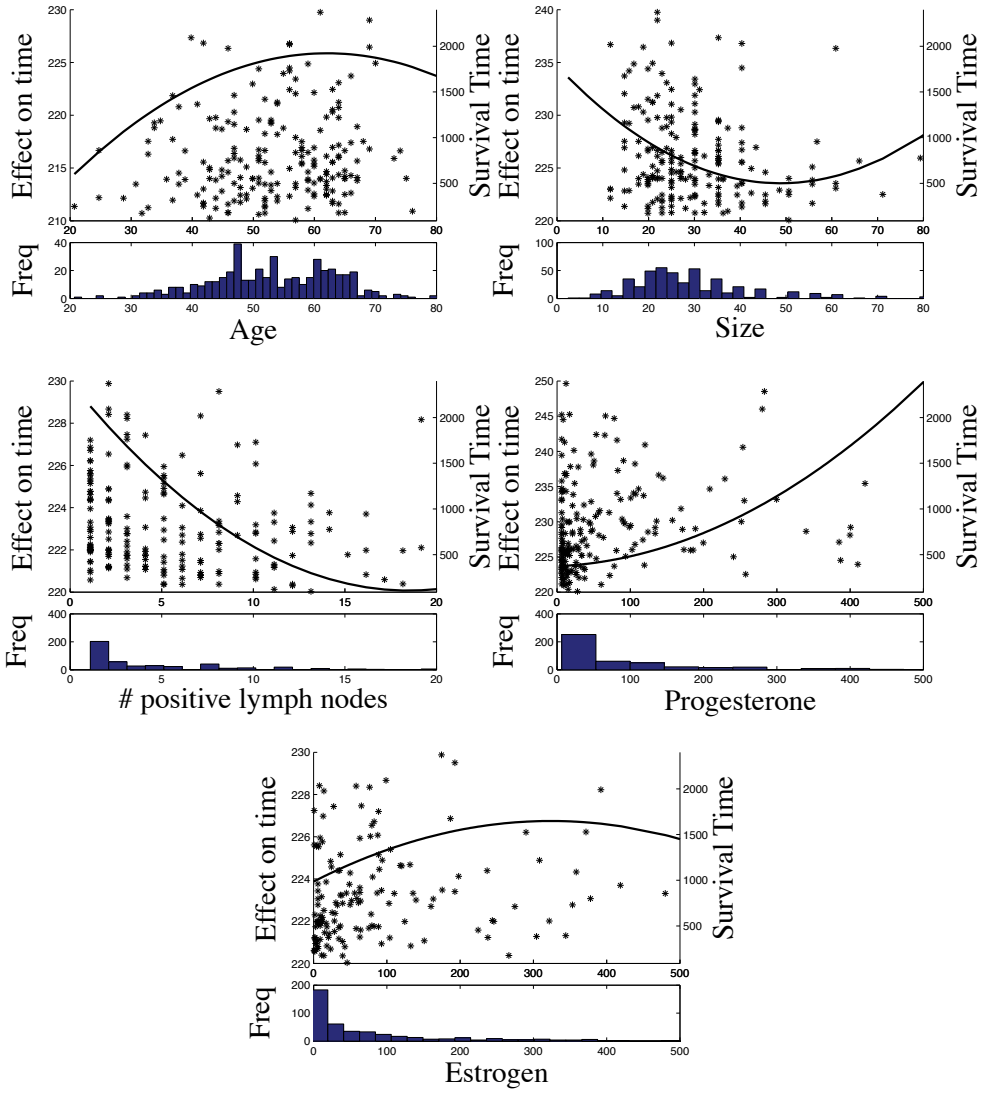
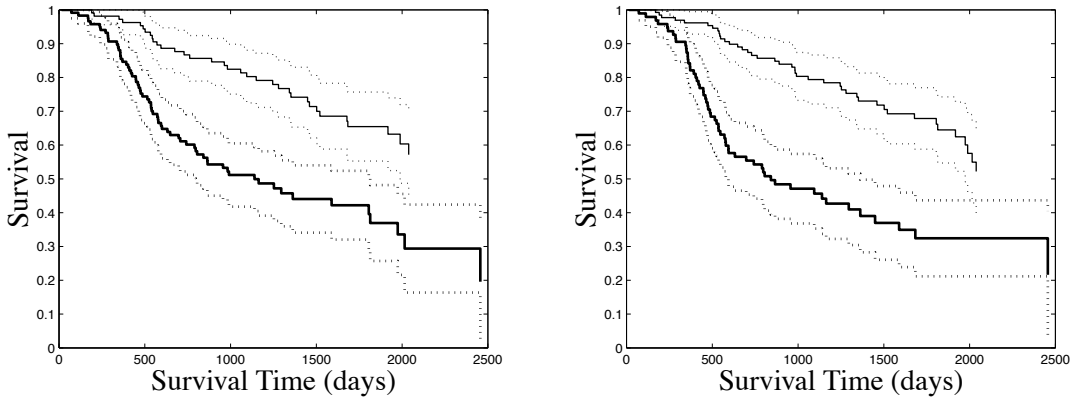


Figure 12: Estimation of covariate effects on survival time (remark the difference with Figure 11) with the MINLIP model ($C_n(u)$ was used for model selection) and histograms of the variables. The stars indicate the observed failure times for breast cancer related events. The estimated covariate effects are directly related with the survival time. The MINLIP model estimates the covariate effects as follows: the estimated survival time increases with age until the age of 65, whereafter the survival time drops slightly. The larger the tumor, the higher the number of positive lymph nodes, the lower the expression of the receptors, the lower the estimated survival time is. Conclusions drawn by the model agree with what is known from literature.



(a) Cox (b) MINLIP

Figure 13: The use of Cox' and MINLIP model as a prognostic index. The output of both models is used to divide the test set into two groups, one with high and one with low risk for relapse. The threshold between both groups is defined as the median value of the model's output. Kaplan-Meier curves and 95% confidence intervals are shown for each group. The spread in the survival curves is broader for the MINLIP model, which is confirmed by a larger value of the log rank test statistic.

Acknowledgments

We thank the editor and the reviewers for their helpful comments and suggestions. This research is supported by Research Council KUL: GOA AMBioRICS, GOA MaNet, CoE EF/05/006, IDO 05/010 IOF-KP06/11, IOF-SCORES4CHEM, several PhD, postdoc end fellow grants; Flemish Government: FWO: PhD and postdoc grants, G.0407.02, G.0360.05, G.0519.06, FWO-G.0321.06, G.0341.07, projects G.0452.04, G.0499.04, G.0211.05, G.0226.06, G.0302.07; IWT: PhD Grants, McKnow-E, Eureka-Flite; Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, 'Dynamical systems, control and optimization', 2007-2011); EU: FP6-2002-LIFESCI-HEALTH 503094, IST-2004-27214, FP6-MC-RTN-035801; Prodex-8 C90242; EU: ERNSI; the European Research Council under the Seventh Framework Program and within Advanced Grant no. 247035 "Systems and Signals Tools for Estimation and Analysis of Mathematical Models in Endocrinology and Neurology". V. Van Belle is supported by a grant from the IWT. K. Pelckmans is an associated professor/researcher ('forskarassistent') at Uppsala University, Sweden at the department of Information Technology, division of SysCon. S. Van Huffel is a full professor and J.A.K. Suykens is a professor at the Katholieke Universiteit Leuven, Belgium.

Appendix A. Consistency and Identifiability

This first Appendix deals with the issues of consistency and identifiability of the proposed method. We study the question under what conditions MINLIP is consistent, that is, if we have enough datapoints at our disposal, would the estimate $\hat{w}$ converge to the desired parameter vector?

Assume that any observation $(X,Y) \in \mathbb{R}^d \times \mathbb{R}$ would obey the relation

$$Y = h_0(w_0^T X),$$

where we refer to the (fixed but unknown) vector $w_0 \in \mathbb{R}^d$ as to the 'true' parameters, and to the (fixed but unknown) monotonically increasing function $h_0 : \mathbb{R} \to \mathbb{R}$ as the 'true' transformation function. We will focus on estimating the vector of parameters w_0, recovery of h_0 may be done in a second stage. Note that assuming that a finite value L_0 exists, together with an observation (X,Y) where this Lipschitz constant is met, is sufficient for consistency. We will fix $\|w_0\|_2 = 1$ to avoid the obvious identifiability issue, namely that for any strictly positive constant $\alpha > 0$, the system $Y = h_{0,\alpha}(\alpha w_0^T X)$ is not distinguishable from (6) when $h_{0,\alpha}(Z) \triangleq h_0(\frac{Z}{\alpha})$ for any $Z \in \mathbb{R}$.

Let the set of all (possibly an infinite number) observations $\{(X,Y)\} \subset \mathbb{R}^d \times \mathbb{R}$ obeying the system (6) be denoted as $\mathcal{D}$. We consider that this set is an ε-non-degenerate set, which is defined as follows

Definition 5 (An ε-non-degenerate Set) *Let $\varepsilon > 0$ be any (arbitrarily small) constant. We say that a set $\mathcal{D} = \{(X,Y)\} \subseteq \mathbb{R}^d \times \mathbb{R}$ is ε-non-degenarate if for any observation $(X,Y) \in \mathcal{D}$, and for any vector $v \in \mathbb{R}^d$, one has an observation $(X',Y') \in \mathcal{D}$ different from (X,Y) such that $\|X - X'\|_2 \leq \varepsilon$ so that*

$$v^T (X - X') \geq 0.$$

This requirement can be relaxed as it only has to hold for the point (X,Y) where the Lipschitz condition is met. In addition, we assume that h_0 is (L_0,a)-Lipschitz on this set $\mathcal{D} \subseteq \mathbb{R}$.

Definition 6 (h_0 is (L_0,a)-Lipschitz on $\mathcal{D}' \subseteq \mathbb{R}$) *The monotonically increasing function h_0 is said to be (L_0,a)-Lipschitz on $\mathcal{D} \subseteq \mathbb{R}$ if (1) h_0 is Lipschitz smooth with Lipschitz constant L_0 for all pairs Z,Z^*, with $Z \geq Z^*$:*

$$h_0(Z) - h_0(Z^*) \leq L_0 (Z - Z^*),$$

and (2) there exists a pair $Z,Z' \in \mathcal{D}$ with $Z > Z'$ where the Lipschitz constant is met:

$$h_0(Z) - h_0(Z') = L_0 \left(Z - Z'\right),$$

and (3) one has for any $\varepsilon > 0$ and $Z'' \in \mathcal{D}'$ where $0 \leq Z - Z'' \leq \varepsilon$ that

$$\frac{L_0}{1 + a\varepsilon}(Z - Z'') \leq h_0(Z) - h_0(Z''), \tag{17}$$

with $a \geq 0$.

Hence a denotes how 'smooth' the constant L_0 decays in a neighborhood of Z where the actual Lipschitz constraint is met (that is, a smaller a indicates higher smoothness) (see Figure 14). In particular, a value $a \to 0$ (arbitrarily small) implies that the function h_0 is linear with slope L_0. Note that this definition does not require that the function $\dfrac{\partial h_0(Z)}{\partial Z}$ exists for any $Z \in \mathbb{R}$. This definition implies the inequality

$$\frac{1}{L_0} = \frac{Z - Z'}{h_0(Z) - h_0(Z')} \leq \frac{Z - Z''}{h_0(Z) - h_0(Z'')} \leq \frac{Z - Z'}{h_0(Z) - h_0(Z')} + \frac{a\varepsilon}{L_0}. \tag{18}$$

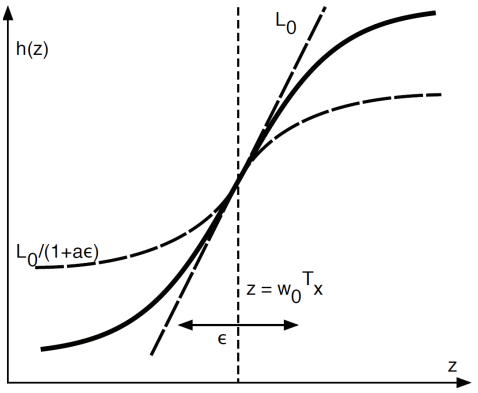


Figure 14: *Illustration of a function $h : \mathbb{R} \to \mathbb{R}$ (solid curved line) which is (L_0, a)-Lipschitz monotone according to Definition 6. The Lipschitz inequality is tight around Z indicated by the dotted line, while in the ε-neighborhood of Z the actual Lipschitz constant decays as slowly as $\frac{L_0}{1+a\varepsilon}$.*

The first inequality holds due to the Lipschitz smoothness of h_0. The second inequality follows from (17).

We now state that (w_0, h_0) can be recovered successfully ('identified') if $\mathcal{D}$ and h_0 were such that Definition 5 and 6 hold. We consider the MINLIP estimator based on an ε-non-degenerate set $\mathcal{D}$ which is defined as

$$w_\varepsilon = \arg \max_{\|w\|_2=1} \inf_{(X,Y),(X',Y')\in\mathcal{D}:Y>Y'} \frac{w^T(X-X')}{Y-Y'},$$

or equivalently (up to a scaling)

$$\begin{aligned} w_\varepsilon \propto \quad & \arg\min_w \quad \frac{1}{2} w^T w \\ & \text{s.t.} \qquad (Y-Y') \le w^T(X-X') \quad \forall\, (X,Y),(X',Y') \in \mathcal{D} : Y > Y'. \end{aligned} \tag{19}$$

Specifically $w_\varepsilon = \frac{\hat{w}}{\|\hat{w}\|_2}$ where $\hat{w}$ is the optimizer of (19). If $\mathcal{D}$ contains a finite number of elements this problem can be solved efficiently as a convex Quadratic Program (QP) using standard solvers. This estimator would return the desired result $w_\varepsilon = w_0$ if enough observations were given. This is stated more formally as follows.

Lemma 2 (Identifiability) *Let $\varepsilon > 0$ be any (arbitrarily small) constant. Given a model (h_0, w_0) governing the observations in $\mathcal{D}$. Assume that (i) the set $\mathcal{D}$ is $(a\varepsilon)$-non-degenerate as in Definition 5; (ii) the function h_0 is (L_0, a)-Lipschitz monotone on the set $\mathcal{D}' = \{Z = w_0^T X \in \mathbb{R} : (X,Y) \in \mathcal{D}\}$, as in Definition 6. Then one has for all $w \in \mathbb{R}^d$ where $\|w\|_2 = 1$ that*

$$\frac{1}{L_0} \le \inf_{(X,Y),(X',Y')\in\mathcal{D}:Y>Y'} \frac{w^T(X-X')}{Y-Y'},$$

with equality if $w = w_0$.

Proof Let $(X,Y),(X',Y') \in \mathcal{D}$ be such that $Y \neq Y'$ and the Lipschitz constant is achieved, or

$$\frac{1}{L_0} = \frac{w_0^T(X-X')}{h_0(w_0^T X) - h_0(w_0^T X')} = \frac{w_0^T(X-X')}{Y-Y'}. \tag{20}$$

Such an observation exists assuming that h_0 is (L_0,a)-Lipschitz on $\mathcal{D}'$. We prove that for a bad estimation w of w_0 ($w^T w_0 < 1-\varepsilon$), one can always find an observation $(X^*,Y^*) \in \mathcal{D}$ such that $\frac{w^T(X-X^*)}{Y-Y^*}$ is strictly lower than $\frac{1}{L_0}$. This implies that when w deviates a fraction ε from w_0, the objective in (19) can never achieve the maximum value as would be the case when $w = w_0$. This implies consistency of the MINLIP estimator.

At first, by the (L_0,a)-Lipschitz condition on h_0, one has for all $(X'',Y'') \in \mathcal{D}$ where $|w_0^T(X-X'')| \leq \varepsilon$ and $Y \neq Y''$ that (as in inequality (18)),

$$\frac{w_0^T(X-X')}{Y-Y'} \geq \frac{w_0^T(X-X'')}{Y-Y''} - \frac{a\varepsilon}{L_0}. \tag{21}$$

According to the Cauchy-Schwarz' inequality, the condition $|w_0^T(X-X'')| \leq \varepsilon$ is fulfilled for $||X-X''||_2 \leq \varepsilon$. Secondly, for any $w \in \mathbb{R}^d$ with $\|w\|_2 = 1$ and $w_0^T w < 1 - a\varepsilon$, one has by the orthogonal decomposition of a vector that

$$\begin{aligned} w_0 - w &= \frac{w_0 w_0^T}{||w_0||_2}(w_0 - w) + v \\ &= w_0(w_0^T w_0 - w_0^T w) + v \\ &= w_0 a\varepsilon^+ + v, \end{aligned}$$

with v the orthogonal complement of the projection of $w_0 - w$ on w_0 and $\varepsilon^+ > \varepsilon$. It follows for any $(X''',Y''') \in \mathcal{D}$ where $Y \neq Y'''$ that

$$\frac{(w_0-w)^T(X-X''')}{Y-Y'''} = \frac{a\varepsilon^+ w_0^T(X-X''')}{Y-Y'''} + \frac{v^T(X-X''')}{Y-Y'''}.$$

Hence by assumption of the set $\mathcal{D}$ being $(a\varepsilon)$-non-degenerate, there exists for any $w \in \mathbb{R}^d$ (and thus for any $v \in \mathbb{R}^d$) an observation $(X^*,Y^*) \in \mathcal{D}$ with $\|X-X^*\|_2 \leq a\varepsilon, Y \neq Y^*$ such that

$$\frac{(w_0-w)^T(X-X^*)}{Y-Y^*} = a\varepsilon^+ \frac{w_0^T(X-X^*)}{Y-Y^*} > a\varepsilon \frac{w_0^T(X-X^*)}{Y-Y^*} \geq \frac{a\varepsilon}{L_0}. \tag{22}$$

From (21) and (22) it then follows that

$$\frac{1}{L_0} = \frac{w_0^T(X-X')}{Y-Y'} \geq \frac{w_0^T(X-X^*)}{Y-Y^*} - \frac{a\varepsilon}{L_0} > \frac{w^T(X-X^*)}{Y-Y^*}.$$

Hence, for all $w \in \mathbb{R}^d$ for which $\|w\|_2 = 1$ and $w_0^T w < 1 - a\varepsilon$, there are two observations $(X,Y),(X^*,Y^*) \in \mathcal{D}$ such that

$$\frac{1}{L_0} > \frac{w^T(X-X^*)}{Y-Y^*},$$

proving the result. Equality as in (20) is reached for $w = w_0$. ■

Appendix B. MINLIP for Ranking Problems

The formal derivation of the MINLIP method is given in this Appendix. We start with the problem formulation as denoted in Equation (8):

$$\begin{aligned} \min_{w,\varepsilon} \quad & \tfrac{1}{2} w^T w + \gamma \|\varepsilon\|_1 \\ \text{s.t.} \quad & \mathbf{D}(\mathbf{\Phi} w + \varepsilon) \geq \mathbf{DY}, \end{aligned}$$

with $\mathbf{\Phi} = [\varphi(X_1), \cdots, \varphi(X_n)]^T$. Take $\varepsilon = e^+ + e^-$ and suppose $e^+ \geqslant 0$ and $e^- \geqslant 0$. The problem can than be formulated as

$$\begin{aligned} \min_{w,e^+,e^-} \quad & \tfrac{1}{2} w^T w + \gamma 1_n^T (e^+ + e^-) \\ \text{s.t.} \quad & \begin{cases} \mathbf{D}(\mathbf{\Phi} w + (e^+ - e^-)) \geq \mathbf{DY}, \\ e^+ \geqslant 0, \\ e^- \geqslant 0. \end{cases} \end{aligned}$$

The Lagrangian becomes

$$\mathcal{L}(w, e^+, e^-; \alpha, \beta_+, \beta_-) = \frac{1}{2} w^T w + \gamma 1_n^T (e^+ + e^-) - \beta_+^T e^+ - \beta_-^T e^- - \alpha^T \mathbf{D}(\mathbf{\Phi} w + e^+ - e^- - \mathbf{Y}),$$

with Lagrange multipliers α, β^+, $\beta^- \geqslant 0$. The conditions for optimality (Karush-Kuhn-Tucker (KKT) conditions) become

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \to w = (\Phi)^T \mathbf{D}^T \alpha \\ \frac{\partial \mathcal{L}}{\partial e^+} = 0 \to \gamma = \mathbf{D}^T \alpha + \beta^+ \\ \frac{\partial \mathcal{L}}{\partial e^-} = 0 \to \gamma = -\mathbf{D}^T \alpha + \beta^- \\ \text{diag}(\alpha) \mathbf{D}(\mathbf{\Phi} w + e^+ - e^- - \mathbf{Y}) = 0 \\ \text{diag}(\beta^+) e^+ = 0 \\ \text{diag}(\beta^-) e^- = 0 \\ \alpha \geqslant 0 \\ \beta^+ \geqslant 0 \\ \beta^- \geqslant 0, \end{cases} \tag{23}$$

where $\text{diag}(a)$ indicates a diagonal matrix with the elements of the vector a on the main diagonal. Now from Slater's condition one could exchange $\min_{w,e^+,e^-} \max_\alpha$ with $\max_\alpha \min_{w,e^+,e^-}$. Solving for w, e^+ and e^- gives the dual problem

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2} \alpha^T \mathbf{DKD}^T \alpha - \alpha^T \mathbf{DY} \\ \text{s.t.} \quad & \begin{cases} -\gamma 1_n \leq \mathbf{D}^T \alpha \leq \gamma 1_n \\ \alpha \geq 0_{n-1}, \end{cases} \end{aligned}$$

and from the first condition of (23) and the model specification $u(X) = w^T \varphi(X)$ one could write the solution for a new point X^* as

$$\hat{u}(X^*) = \mathbf{K}_n^* \mathbf{D}^T \hat{\alpha},$$

with $\mathbf{K}_n^* \in \mathbb{R}^n$ and $\mathbf{K}_n^* = [K(X^*, X_1) \ \ldots \ K(X^*, X_n)]^T$.

Appendix C. MINLIP for Ordinal Regression

In this appendix the derivation of the MINLIP method for ordinal regression is exposed. In the ordinal regression case unknown thresholds v are introduced corresponding to an outcome intermediate between two successive outcome levels. The model is built by indicating that the difference between the utility of a certain observation X_i and the largest threshold lower than the outcome of that observation Y_i should be larger than the difference between Y_i and the outcome corresponding to the before mentioned threshold. Analogously, the difference between the smallest threshold higher than Y_i should be larger than the difference between the outcome corresponding to that threshold and Y_i. As an extra constraint we impose that successive threshold are increasing values of the utility function. More formally the problem is formulated as in Equation (11), now using the kernel based version:

$$\begin{array}{ll} \min\limits_{w,e,e^*,v} & \|w\|_2 + \gamma 1_n^T (e+e^*) \\ \text{s.t.} & \begin{cases} \boldsymbol{\Phi} w - \mathbf{Q} v + e \geq \mathbf{Y} - \mathbf{QB} \\ -\boldsymbol{\Phi} w + \mathbf{Q}^* v + e^* \geq -\mathbf{Y} + \mathbf{Q}^* \mathbf{B} \\ e \geq 0 \\ e^* \geq 0 \\ \mathbf{M} v \leq 0\,. \end{cases} \end{array}$$

As in Appendix B we build the Lagrangian

$$\begin{aligned} \mathcal{L}(w,e^+,e^-;\alpha,\beta,\eta,\eta_*,\nu) = \; & \tfrac{1}{2} w^T w + \gamma 1_n^T (e+e^*) - \alpha^T (\boldsymbol{\Phi} w - \mathbf{Q} v + e - \mathbf{Y} + \mathbf{QB}) \\ & - \beta^T (-\boldsymbol{\Phi} w + \mathbf{Q}^* v + e^* + \mathbf{Y} - \mathbf{Q}^* \mathbf{B}) - \eta^T e - \eta^{*T} e^* \\ & + \nu^T \mathbf{M} v\,, \end{aligned}$$

and derive the set of optimality conditions

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \rightarrow w = \boldsymbol{\Phi}^T (\alpha - \beta) \\ \frac{\partial \mathcal{L}}{\partial e} = 0 \rightarrow \gamma 1_n = \alpha + \eta \\ \frac{\partial \mathcal{L}}{\partial e^*} = 0 \rightarrow \gamma = \beta + \eta^* \\ \frac{\partial \mathcal{L}}{\partial v} = 0 \rightarrow \alpha^T \mathbf{Q} - \beta^T \mathbf{Q}^* + \nu^T \mathbf{M} v = 0 \\ \operatorname{diag}(\alpha) D(\boldsymbol{\Phi} w - \mathbf{Q} v + e - \mathbf{Y} + \mathbf{QB}) = 0 \\ \operatorname{diag}(\beta)(-\boldsymbol{\Phi} w + \mathbf{Q}^* v + e^* + \mathbf{Y} - \mathbf{Q}^* \mathbf{B}) = 0 \\ \operatorname{diag}(\eta) e = 0 \\ \operatorname{diag}(\eta^*) e^* = 0 \\ \operatorname{diag}(\nu) \mathbf{M} v = 0 \\ \alpha \geqslant 0 \\ \beta \geqslant 0 \\ \eta \geqslant 0 \\ \eta^* \geqslant 0 \\ \nu \geqslant 0\,. \end{cases}$$

The dual problem formulation is than found as

$$\min_{\alpha,\beta} \quad \tfrac{1}{2}\alpha^T \mathbf{K}\alpha + \tfrac{1}{2}\beta^T \mathbf{K}\beta - \alpha^T \mathbf{K}\beta - \alpha^T(\mathbf{Y} - \mathbf{B}^T\mathbf{Q}) + \beta^T(\mathbf{Y} - \mathbf{B}^T\mathbf{Q}^*)$$

$$\text{s.t.} \quad \begin{cases} 0_n \leq \alpha \leq \gamma 1_n \\ 0_n \leq \beta \leq \gamma 1_n \\ 0_{k-2} \leq \nu \\ \mathbf{Q}^T\alpha - \mathbf{Q}^{*T}\beta + \mathbf{M}^T\nu = 0_{k-1}\,. \end{cases}$$

References

N. Ailon and M. Mohri. An efficient reduction of ranking to classification. In *COLT*, pages 87–98. Omnipress, 2008.

P.K. Andersen, O. Borgan, R.D. Gill, and N. Leiding. *Statistical Models based on Counting Processes*. Springer-Verlag, New York, 1993.

E. Bair and R. Tibshirani. Semi-supervised methods to predict patient survival from gene expression data. *PLoS Biology*, 2(4):511–522, April 2004.

R. Bair, T. Hastie, P. Debashis, and R. Tibshirani. Prediction by supervised principal components. *Journal of the American Statistical Association*, 101:119–137, 2006.

E. Biganzoli, P. Boracchi, L. Mariani, and E. Marubini. Feedforward neural networks for the analysis of censored survival data: a partial logistic regression approach. *Statistics in Medicine*, 17 (10):1169–1186, 1998.

H.M.M. Bøvelstad, S. Nygård, H.L.L. Størvold, M. Aldrin, O. Borgan, A. Frigessi, and O.C.C. Lingjærde. Predicting survival from microarray data - a comparative study. *Bioinformatics*, 23 (16):2080–2087, 2007.

N. Breslow. Covariance analysis of censored survival data. *Biometrics*, 30(1):89–99, 1974.

S.C. Cheng, L.J. Wei, and Z. Ying. Predicting survival probabilities with semiparametric transformation models. *Journal of the American Statistical Association*, 92(437):227–235, 1997.

W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:1019–1041, 2005.

W. Chu and S.S. Keerthi. New approaches to support vector ordinal regression. In *In ICML 2005: Proceedings of the 22nd international conference on Machine Learning*, pages 145–152, 2005.

S. Clémençon and N. Vayatis. Ranking the best instances. *Journal of Machine Learning Research*, 8:2671–2699, 2007.

S. Clémençon, G. Lugosi, and N. Vayatis. Ranking and scoring using empirical risk minimization. In *Proceedings of the 18th Annual Conference in Learning Theory (COLT)*, Bertinoro (Italy), June 2005.

D.R. Cox. Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society, Series B*, 34(2):187–220, 1972.

D.M. Dabrowska and K.A. Doksum. Partial likelihood in transformation models with censored data. *Scandinavian Journal of Statistics*, 15(1):1–23, 1988.

P.H. Eilers and B.D. Marx. Flexible smoothing with B-splines and penalties. *Statistical Science*, 11:89–121, 1996.

R.C. Elandt-Johnson and N.L. Johnson. *Survival Models and Data Anlysis*. John Wiley & Sons, Inc., 1980.

D.M. Finkelstein. A proportional hazards model for interval-censored failure time data. *Biometrics*, 42:845–854, 1986.

B. Fisher, M. Bauer, L. Wickerham, C.K. Redmong, and E.R. Fisher. Relation of number of positive axillary nodes to the prognosis of patients with primary breast cancer. An NSABP update. *Cancer*, 52(9):1551–1557, 1983.

Y. Freund, R. Iyer, R.E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4(6):933–969, 2004.

J. Fürnkranz and E. Hüllermeier. Pairwise preference learning and ranking. In *Proceedings of the European Conference on Machine Learning 2003, Cavtat-Dubrovnik*, pages 145–156. Springer-Verlag, 2003.

F. Harrell. *Regression Modeling Strategies. With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer, 2001.

F. Harrell, K. Klee, R. Califf, D. Pryor, and R. Rosati. Regression modeling strategies for improved prognostic prediction. *Statistics in Medicine*, 3(2):143–152, 1984.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2001.

P. Heagerty, T. Lumley, and M. Pepe. Time-dependent ROC curves for censored survival data and a diagnostic marker. *Biometrics*, 56(2):337–344, 2000.

R. Herbrich, T. Graepel, P. Bollmann-Sdorra, and K. Obermayer. Learning preference relations for information retrieval. In *Proceedings of the Fifteenth Conference of the American Association for Artificial Intelligence*, pages 1–4, 1998.

R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In Smola, Bartlett, Schoelkopf, and Schuurmans, editors, *Advances in Large Margin Classifiers*, 2000.

C.M. Hurvich, J.S. Simonoff, and C.L. Tsai. Smoothing parameter selection in nonparametric regression using an improved akaike information criterion. *Journal of the Royal Statistical Society: Series B*, 60:371–293, 1998.

J.D. Kalbfleisch and R.L. Prentice. *The Statistical Analysis of Failure Time Data*. Wiley series in probability and statistics. 2002.

M.W. Kattan, H. Ishida, P.T. Scardino, and J.R. Beck. Applying a neural network to prostate cancer survival data. In N. Lavrac, R. Keravnou, and B. Zupan, editors, *Intelligent Data Analysis in Medicine and Pharmacology*, pages 295–306. Kluwer, Boston, 1997.

R. Koenker and O. Geling. Reappraising medfly longevity: A quantile regression survival analysis. *Journal of the American Statistical Association*, 96:458–468, 2001.

P.J. Lamy, P. Pujol, S. Thezenas, A. Kramar, P. Rouanet, F. Guilleux, and J. Grenier. Progesterone receptor quantification as a strong prognostic determinant in postmenopausal breast cancer women under tamoxifen therapy. *Breast cancer reasearch and treatment*, 76(1):65–71, 2002.

D.J.C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computing*, 4(3):448–472, 1992.

H. Martens and T. Næs. *Multivariate Calibration*. New York: John Wiley & Sons Inc., 1989.

R.G. Miller. *Survival Anlysis*. John Wiley & Sons, 1981.

S. Nygård, O. Borgan, O. Lingjærde, and H. Størvold. Partial least squares Cox regression for genome-wide data. *Lifetime Data Analysis*, 14(2):179–195, 2008.

K. Pelckmans, M. Espinoza, J. De Brabanter, J.A.K. Suykens, and B. De Moor. Primal-dual monotone kernel regression. *Neural Processing Letters*, 22(2):171–182, 2005a.

K. Pelckmans, I. Goethals, J. De Brabanter, J.A.K. Suykens, and B. De Moor. *Componentwise Least Squares Support Vector Machines*, chapter in Support Vector Machines: Theory and Applications, pages 77–98. (L. Wang, ed.), Springer, 2005b.

R. Peto. Discussion of paper by D.R. Cox. *Journal of the Royal Statistical Society, Series B*, 34: 205–207, 1972.

M.F. Pichon, C. Pallud, M. Brunet, and E. Milgrom. Relationship of presence of progesterone receptors to prognosis in early breast cancer. *Cancer Research*, 40:3357–3360, 1980.

A. Rosenwald, G. Wright, W.C. Chan, J.M. Connors, E. Campo, R.I. Fisher, R.D. Gascoyne, H.K. Muller-Hermelink, R.B. Smeland, J.M. Giltnane, E.M. Hurt, H. Zhao, L. Averett, and L. Yang. The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. *The New England Journal of Medicine*, 346(25):1937–1947, 2002.

G.A. Satten. Rank-based inference in the proportional hazards model for interval censored data. *Biometrika*, 83:355–370, 1996.

M. Schumacher, G. Basert, H. Bojar, K. Huebner, M. Olschewski, W. Sauerbrei, C. Schmoor, C. Beyerle, R.L.A. Neumann, and H.F. Rauschecker. Randomized 2 x 2 trial evaluating hormonal treatment and the duration of chemotherapy in node-positive breast cancer patients. *Journal of Clinical Oncology*, 12, 1994.

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

T. Sørlie, R. Tibshirani, J. Parker, T. Hastie, J.S. Marron, A. Nobel, S. Deng, H. Johnsen, R. Pesich, S. Geisler, J. Demeter, C.M. Perou, P.E. Lønning, P.O. Brown, A. Børresen-Dale, and D. Botstein. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proceedings of the National Academy of Sciences of the United States of America*, 100(14):8418–8423, 2003.

J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

T.M. Therneau and P.M. Grambsch. *Modeling Survival Data: Extending the Cox Model*. Springer, 2 edition, 2000.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

V. Van Belle, K. Pelckmans, J.A.K. Suykens, and S. Van Huffel. Support vector machines for survival analysis. In *Proceedings of the Third International Conference on Computational Intelligence in Medicine and Healthcare (CIMED2007)*, pages 1–8, Plymouth (UK), July 2007.

V. Van Belle, K. Pelckmans, J.A.K. Suykens, and S. Van Huffel. Survival SVM: a practical scalable algorithm. In *Proceedings of the 16th European Symposium on Artificial Neural Networks (ESANN2008)*, pages 89–94, Bruges (Belgium), April 2008.

V. Van Belle, K. Pelckmans, J.A.K. Suykens, and S. Van Huffel. MINLIP: Efficient learning of transformation models. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN2009)*, pages 60–69, Limassol (Cyprus), September 2009.

M.J. van de Vijver, L.J. van't Veer, and H. Dai. A gene-expression signature as a predictor of survival in breast cancer. *The New England Journal of Medicine*, 347(25):1999–2009, 2002.

H.C. van Houwelingen, T. Bruinsma, A.A.M. Hart, L.J. van't Veer, and L.F.A. Wessels. Cross-validated cox regression on microarray gene expression data. *Statistics in Medicine*, 25(18): 3201–3216, 2006.

V. Vapnik. *Statistical Learning Theory*. Wiley and Sons, 1998.

C. Verschraegen, C. Vihn-Hung, G. Cserni, R. Gordon, M.E. Royce, G. Vlastos, P. Tai, and G. Storme. Modeling the effect of tumor size in early breast cancer. *Annals of Surgery*, 241 (2):309–318, 2005.

Journal of Machine Learning Research 12 (2011) 863-905 Submitted 10/09; Revised 10/10; Published 3/11

Sparse Linear Identifiable Multivariate Modeling

Ricardo Henao* RHENAO@BINF.KU.DK
Ole Winther* OWI@IMM.DTU.DK
DTU Informatics
Richard Petersens Plads, Building 321
Technical University of Denmark
DK-2800 Lyngby, Denmark

Editor: Aapo Hyvärinen

Abstract

In this paper we consider sparse and identifiable linear latent variable (factor) and linear Bayesian network models for parsimonious analysis of multivariate data. We propose a computationally efficient method for joint parameter and model inference, and model comparison. It consists of a fully Bayesian hierarchy for sparse models using slab and spike priors (two-component δ-function and continuous mixtures), non-Gaussian latent factors and a stochastic search over the ordering of the variables. The framework, which we call SLIM (Sparse Linear Identifiable Multivariate modeling), is validated and bench-marked on artificial and real biological data sets. SLIM is closest in spirit to LiNGAM (Shimizu et al., 2006), but differs substantially in inference, Bayesian network structure learning and model comparison. Experimentally, SLIM performs equally well or better than LiNGAM with comparable computational complexity. We attribute this mainly to the stochastic search strategy used, and to parsimony (sparsity and identifiability), which is an explicit part of the model. We propose two extensions to the basic i.i.d. linear framework: non-linear dependence on observed variables, called SNIM (Sparse Non-linear Identifiable Multivariate modeling) and allowing for correlations between latent variables, called CSLIM (Correlated SLIM), for the temporal and/or spatial data. The source code and scripts are available from `http://cogsys.imm.dtu.dk/slim/`.

Keywords: parsimony, sparsity, identifiability, factor models, linear Bayesian networks

1. Introduction

Modeling and interpretation of multivariate data are central themes in machine learning. Linear latent variable models (or factor analysis) and linear directed acyclic graphs (DAGs) are prominent examples of models for continuous multivariate data. In factor analysis, data is modeled as a linear combination of independently distributed factors thus allowing for capture of a rich underlying covariation structure. In the DAG model, each variable is expressed as regression on a subset of the remaining variables with the constraint that total connectivity is acyclic in order to have a properly defined joint distribution. Parsimonious (interpretable) modeling, using sparse factor loading matrix or restricting the number of parents of a node in a DAG, are good prior assumptions in many applications. Recently, there has been a great deal of interest in detailed modeling of sparsity in factor models, for example in the context of gene expression data analysis (West, 2003; Lucas et al.,

*. Ricardo Henao and Ole Winther hold joint affiliations with DTU Informatics, Technical University of Denmark and the Bioinformatics Centre, University of Copenhagen.

©2011 Ricardo Henao and Ole Winther.

2006; Knowles and Ghahramani, 2007; Thibaux and Jordan, 2007; Carvalho et al., 2008; Rai and Daume III, 2009). Sparsity arises for example in gene regulation because the latent factors represent driving signals for gene regulatory sub-networks and/or transcription factors, each of which only includes/affects a limited number of genes. A parsimonious DAG is particularly attractable from an interpretation point of view but the restriction to only having observed variables in the model may be a limitation because one rarely measures all relevant variables. Furthermore, linear relationships might be unrealistic for example in gene regulation, where it is generally accepted that one cannot replace the driving signal (related to concentration of a transcription factor protein in the cell nucleus) with the measured concentration of corresponding mRNA. Bayesian networks represent a very general class of models, encompassing both observed and latent variables. In many situations it will thus be relevant to learn parsimonious Bayesian networks with both latent variables and a non-linear DAG parts. Although attractive, by being closer to what one may expect in practice, such modeling is complicated by difficult inference (Chickering 1996 showed that DAG structure learning is NP-hard) and by potential non-identifiability. Identifiability means that each setting of the parameters defines a unique distribution of the data. Clearly, if the model is not identifiable in the DAG and latent parameters, this severely limits the interpretability of the learned model.

Shimizu et al. (2006) provided the important insight that every DAG has a factor model representation, that is, the connectivity matrix of a DAG gives rise to a triangular mixing matrix in the factor model. This provided the motivation for the Linear Non-Gaussian Acyclic Model (LiNGAM) algorithm which solves the identifiable factor model using Independent Component Analysis (ICA, Hyvärinen et al., 2001) followed by iterative permutation of the solutions towards triangular, aiming to find a suitable ordering for the variables. As final step, the resulting DAG is pruned based on different statistics, for example, Wald, Bonferroni, χ^2 second order model fit tests. Model selection is then performed using some pre-chosen significance level, thus LiNGAM select from models with different sparsity levels and a fixed deterministically found ordering. There is a possible number of extensions to their basic model, for instance Hoyer et al. (2008) extend it to allow for latent variables, for which they use a probabilistic version of ICA to obtain the variable ordering, pruning to make the model sparse and bootstrapping for model selection. Although the model seems to work well in practice, as commented by the authors, it is restricted to very small problems (3 or 4 observed and 1 latent variables). Non-linear DAGs are also a possibility, however finding variable orderings in this case is known to be far more difficult than in the linear case. These methods inspired by Friedman and Nachman (2000), mainly consist of two steps: performing non-linear regression for a set of possible orderings, and then testing for independence to prune the model, see for instance Hoyer et al. (2009) and Zhang and Hyvärinen (2010). For tasks where exhaustive order enumeration is not feasible, greedy approaches like DAG-search (see "ideal parent" algorithm, Elidan et al., 2007) or PC (Prototypical Constraint, see kernel PC, Tillman et al., 2009) can be used as computationally affordable alternatives.

Factor models have been successfully employed as exploratory tools in many multivariate analysis applications. However, interpretability using sparsity is usually not part of the model, but achieved through post-processing. Examples of this include, bootstrapping, rotating the solutions to maximize sparsity (varimax, procrustes), pruning or thresholding. Another possibility is to impose sparsity in the model through L_1 regularization to obtain a maximum a-posteriori estimate (Jolliffe et al., 2003; Zou et al., 2006). In fully Bayesian sparse factor modeling, two approaches have been proposed: parametric models with bimodal sparsity promoting priors (West, 2003; Lucas et al., 2006; Carvalho et al., 2008; Henao and Winther, 2009), and non-parametric models where

the number of factors is potentially infinite (Knowles and Ghahramani, 2007; Thibaux and Jordan, 2007; Rai and Daume III, 2009). It turns out that most of the parametric sparse factor models can be seen as finite versions of their non-parametric counterparts, for instance West (2003) and Knowles and Ghahramani (2007). The model proposed by West (2003) is, as far as the authors know, the first attempt to encode sparsity in a factor model explicitly in the form of a prior. The remaining models improve the initial setting by dealing with the optimal number of factors in Knowles and Ghahramani (2007), improved hierarchical specification of the sparsity prior in Lucas et al. (2006), Carvalho et al. (2008) and Thibaux and Jordan (2007), hierarchical structure for the loading matrices in Rai and Daume III (2009) and identifiability without restricting the model in Henao and Winther (2009).

Many algorithms have been proposed to deal with the NP-hard DAG structure learning task. LiNGAM, discussed above, is the first fully identifiable approach for continuous data. All other approaches for continuous data use linearity and (at least implicitly) Gaussianity assumptions so that the model structure learned is only defined up to equivalence classes. Thus in most cases the directionality information about the edges in the graph must be discarded. Linear Gaussian-based models have the added advantage that they are computationally affordable for the many variables case. The structure learning approaches can be roughly divided into stochastic search and score (Cooper and Herskovits, 1992; Heckerman et al., 2000; Friedman and Koller, 2003), constraint-based (with conditional independence tests) (Spirtes et al., 2001) and two stage; like LiNGAM, (Tsamardinos et al., 2006; Friedman et al., 1999; Teyssier and Koller, 2005; Schmidt et al., 2007; Shimizu et al., 2006). In the following, we discuss in more detail previous work in the last category, as it is closest to the work in this paper and can be considered representative of the state-of-the-art. The Max-Min Hill-Climbing algorithm (MMHC, Tsamardinos et al., 2006) first learns the skeleton using conditional independence tests similar to PC algorithms (Spirtes et al., 2001) and then the order of the variables is found using a Bayesian-scoring hill-climbing search. The Sparse Candidate (SC) algorithm (Friedman et al., 1999) is in the same spirit but restricts the skeleton to within a predetermined link candidate set of bounded size for each variable. The Order Search algorithm (Teyssier and Koller, 2005) uses hill-climbing first to find the ordering, and then looks for the skeleton with SC. L_1 regularized Markov Blanket (Schmidt et al., 2007) replaces the skeleton learning from MMHC with a dependency network (Heckerman et al., 2000) written as a set of local conditional distributions represented as regularized linear regressors. Since the source of identifiability in Gaussian DAG models is the direction of the edges in the graph, a still meaningful approach consists of entirely focusing on inferring the skeleton of the graph by keeping the edges undirected as in Dempster (1972), Dawid and Lauritzen (1993), Giudici and Green (1999) and Rajaratman et al. (2008).

In this paper we propose a framework called SLIM (Sparse Linear Identifiable Multivariate modeling, see Figure 1) in which we learn models from a rather general class of Bayesian networks and perform quantitative model comparison between them.[1] Model comparison may be used for model selection or serve as a hypothesis-generating tool. We use the likelihood on a test set as a computationally simple quantitative proxy for model comparison and as an alternative to the marginal likelihood. The other two key ingredients in the framework are the use of sparse and identifiable model components (Carvalho et al., 2008; Kagan et al., 1973, respectively) and the stochastic search for the correct order of the variables needed by the DAG representation. Like LiNGAM, SLIM ex-

1. A preliminary version of our approach appears in NIPS 2009: Henao and Winther, Bayesian sparse factor models and DAGs inference and comparison.

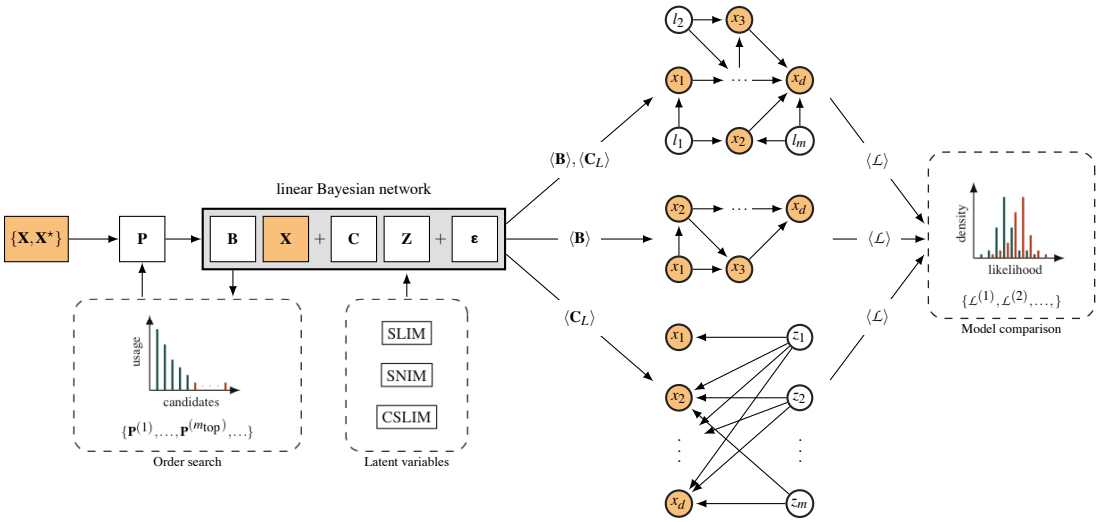


Figure 1: SLIM in a nutshell. Starting from a training-test set partition of data $\{\mathbf{X}, \mathbf{X}^\star\}$, our framework produces factor models $\mathbf{C}$ and DAG candidates $\mathbf{B}$ with and without latent variables $\mathbf{Z}$ that can be compared in terms of how well they fit the data using test likelihoods $\mathcal{L}$. The variable ordering $\mathbf{P}$ needed by the DAG is obtained as a byproduct of a factor model inference. Besides, changing the prior over latent variables $\mathbf{Z}$ produces two variants of SLIM called CSLIM and SNIM.

ploits the close relationship between factor models and DAGs. However, since we are interested in the factor model by itself, we will not constrain the factor loading matrix to have triangular form, but allow for sparse solutions so pruning is not needed. Rather we may ask whether there exists a permutation of the factor-loading matrix agreeing to the DAG assumption (in a probabilistic sense). The slab and spike prior biases towards sparsity so it makes sense to search for a permutation in parallel with factor model inference. We propose to use stochastic updates for the permutation using a Metropolis-Hastings acceptance ratio based on likelihoods with the factor-loading matrix being masked. In practice this approach gives good solutions up to at least fifty dimensions. Given a set of possible variable orderings inferred by this method, we can then learn DAGs using slab and spike priors for their connectivity matrices. The so-called slab and spike prior is a two-component mixture of a continuous distribution and degenerate δ-function point mass at zero. This type of model implicitly defines a prior over structures and is thus a computationally attractive alternative to combinatorial structure search since parameter and structure inference are performed simultaneously. A key to effective learning in these intractable models is Markov Chain Monte Carlo (MCMC) sampling schemes that mix well. For non-Gaussian heavy-tailed distributions like the Laplace and t-distributions, Gibbs sampling can be efficiently defined using appropriate infinite scale mixture representations of these distributions (Andrews and Mallows, 1974). We also show that our model is very flexible in the sense that it can be easily extended by only changing the prior distribution of a set of latent variables, for instance to allow for time series data (CSLIM, Correlated SLIM) and non-linearities in the DAG structure (SNIM, Sparse non-Linear Identifiable Multivariate modeling) through Gaussian process priors.

The rest of the paper is organized as follows: Section 2 describes the model and its identifiability properties. Section 3 provides all prior specification including sparsity, latent variables and driving signals, order search and extensions for correlated data (CSLIM) and non-linearities (SNIM). Section 4 elaborates on model comparison. Section 5 and Appendix A provide an overview of the model and practical details on the MCMC-based inference, proposed workflow and computational cost requirements. Section 6 contains the experiments. We show simulations based on artificial data to illustrate all the features of the model proposed. Real biological data experiments illustrate the advantages of considering different variants of Bayesian networks. For all data sets we compare with some of the most relevant existing methods. Section 7 concludes with a discussion, open questions and future directions.

2. Linear Bayesian Networks

A Bayesian network is essentially a joint probability distribution defined via a directed acyclic graph, where each node in the graph represents a random variable x. Due to the acyclic property of the graph, its node set $x_1,\ldots,x_d$ can be partitioned into d subsets $\{V_1,V_2,\ldots,V_d\} \equiv \mathcal{V}$, such that if $x_j \to x_i$ then $x_j \in V_i$, that is, V_i contains all *parents* of x_i. We can then write the joint distribution as a product of conditionals of the form

$$P(x_1,\ldots,x_d) = \prod_{i=1}^{d} P(x_i|V_i) \ ,$$

thus x_i is conditionally independent of $\{x_j|x_i \notin V_j\}$ given V_i for $i \neq j$. This means that $p(x_1,\ldots,x_d)$ can be used to describe the joint probability of any set of variables once $\mathcal{V}$ is given. The problem is that $\mathcal{V}$ is usually unknown and thus needs to be (at least partially) inferred from observed data.

We consider a model for a fairly general class of linear Bayesian networks by putting together a linear DAG, $\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{z}$, and a factor model, $\mathbf{x} = \mathbf{C}\mathbf{z} + \boldsymbol{\varepsilon}$. Our goal is to explain each one of d observed variables $\mathbf{x}$ as a linear combination of the remaining ones, a set of $d+m$ independent latent variables $\mathbf{z}$ and additive noise $\boldsymbol{\varepsilon}$. We have then

$$\mathbf{x} = (\mathbf{R} \odot \mathbf{B})\mathbf{x} + (\mathbf{Q} \odot \mathbf{C})\mathbf{z} + \boldsymbol{\varepsilon} \ , \tag{1}$$

where $\odot$ is the element-wise product and we can further identify the following elements:

- $\mathbf{z}$ is partitioned into two subsets, $\mathbf{z}_D$ is a set of d driving signals for each observed variable in $\mathbf{x}$ and $\mathbf{z}_L$ is a set of m shared general purpose latent variables. $\mathbf{z}_D$ is used here to describe the intrinsic behavior of the observed variables that cannot regarded as "external" noise.

- $\mathbf{R}$ is a $d \times d$ binary connectivity matrix that encodes whether there is an edge between observed variables, by means of $r_{ij} = 1$ if $x_i \to x_j$. Since every non-zero element in $\mathbf{R}$ is an edge of a DAG, $r_{ii} = 0$ and $r_{ij} = 0$ if $r_{ji} \neq 0$ to avoid self-interactions and bi-directional edges, respectively. This also implies that there is at least one permutation matrix $\mathbf{P}$ such that $\mathbf{P}^\top\mathbf{R}\mathbf{P}$ is strictly lower triangular where we have used that $\mathbf{P}$ is orthonormal then $\mathbf{P}^{-1} = \mathbf{P}^\top$.

- $\mathbf{Q} = [\mathbf{Q}_D \ \mathbf{Q}_L]$ is a $d \times (d+m)$ binary connectivity matrix, this time for the conditional independence relations between observed and latent variables. We assume that each observed variable has a dedicated latent variable, thus the first d columns of $\mathbf{Q}_D$ are the identity. The remaining m columns can be arbitrarily specified, by means of $q_{ij} \neq 0$ if there is an edge between x_i and z_j for $d < j \leq m$.

- $\mathbf{B}$ and $\mathbf{C} = [\mathbf{C}_L \ \mathbf{C}_D]$ are respectively, $d \times d$ and $d \times (d+m)$ weight matrices containing the edge strengths for the Bayesian network. Their elements are constrained to be non-zero only if their corresponding connectivities are also non-zero.

The model (1) has two important special cases, (i) if all elements in $\mathbf{R}$ and $\mathbf{Q}_D$ are zero it becomes a standard factor model (FM) and (ii) if $m = 0$ or all elements in $\mathbf{Q}_L$ are zero it is a pure DAG. The model is not a completely general linear Bayesian network because connections to latent variables are absent (see for example Silva, 2010). However, this restriction is mainly introduced to avoid compromising the identifiability of the model. In the following we will only write $\mathbf{Q}$ and $\mathbf{R}$ explicitly when we specify the sparsity modeling.

2.1 Identifiability

We will split the identifiability of the model in Equation (1) in three parts addressing first the factor model, second the pure DAG and finally the full model. By identifiability we mean that each different setting of the parameters $\mathbf{B}$ and $\mathbf{C}$ gives a unique distribution of the data. In some cases the model is only unique up to some symmetry of the model. We discuss these symmetries and their effect on model interpretation in the following.

Identifiability in factor models $\mathbf{x} = \mathbf{C}_L\mathbf{z}_L + \boldsymbol{\varepsilon}$ can be obtained in a number of ways (see Chapter 10, Kagan et al., 1973). Probably the easiest way is to assume sparsity in $\mathbf{C}_L$ and restrict its number of free parameters, for example by restricting the dimensionality of $\mathbf{z}$, namely m, according to the Ledermann bound $m \leq (2d+1-(8d+1)^{1/2})/2$ (Bekker and ten Berge, 1997). The Ledermann bound guarantees the identification of $\boldsymbol{\varepsilon}$ and follows just from counting the number of free parameters in the covariance matrices of $\mathbf{x}$, $\boldsymbol{\varepsilon}$ and in $\mathbf{C}_L$, assuming Gaussianity of $\mathbf{z}$ and $\boldsymbol{\varepsilon}$. Alternatively, identifiability is achieved using non-Gaussian distributions for $\mathbf{z}$. Kagan et al. (Theorem 10.4.1, 1973) states that when at least $m-1$ latent variables are non-Gaussian, $\mathbf{C}_L$ is identifiable up to scale and permutation of its columns, that is, we can identify $\widehat{\mathbf{C}}_L = \mathbf{C}_L\mathbf{S}_\mathrm{f}\mathbf{P}_\mathrm{f}$, where $\mathbf{S}_\mathrm{f}$ and $\mathbf{P}_\mathrm{f}$ are arbitrary scaling and permutation matrices, respectively. Comon (1994) provided an alternative well-known proof for the particular case of $m-1=d$. The $\mathbf{S}_\mathrm{f}$ and $\mathbf{P}_\mathrm{f}$ symmetries are inherent in the factor model definition in all cases and will usually not affect interpretability. However, some researchers prefer to make the model completely identifiable, for example, by making $\mathbf{C}_L$ triangular with non-negative diagonal elements (Lopes and West, 2004). In addition, if all components of $\boldsymbol{\varepsilon}$ are Gaussian and the rank of $\mathbf{C}_L$ is m, then the distributions of $\mathbf{z}$ and $\boldsymbol{\varepsilon}$ are uniquely defined to within common shift in mean (Theorem 10.4.3, Kagan et al., 1973). In this paper, we use the non-Gaussian $\mathbf{z}$ option for two reasons, (i) restricting the number of latent variables severely limits the usability of the model and (ii) non-Gaussianity is a more realistic assumption in many application areas such as for example biology.

For pure DAG models $\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{C}_D\mathbf{z}_D$, identifiability can be obtained using the factor model result from Kagan et al. (1973) by rewriting the DAG into an equivalent factor model $\mathbf{x} = \mathbf{D}\mathbf{z}$ with $\mathbf{D} = (\mathbf{I}-\mathbf{B})^{-1}\mathbf{C}_D$, see Figure 2. From the factor model result it only follows that $\mathbf{D}$ is identifiable up to a scaling and permutation. However, as mentioned above, due to the acyclicity there is at least one permutation matrix $\mathbf{P}$ such that $\mathbf{P}^\top\mathbf{B}\mathbf{P}$ is strictly lower triangular. Now, if $\mathbf{x}$ admits DAG representation, the same $\mathbf{P}$ makes the permuted $\widehat{\mathbf{D}} = (\mathbf{I}-\mathbf{P}^\top\mathbf{B}\mathbf{P})^{-1}\mathbf{C}_D$, triangular with $\mathbf{C}_D$ on its diagonal. The constraint on the number of non-zero elements in $\mathbf{D}$ due to triangularity removes the permutation freedom $\mathbf{P}_\mathrm{f}$ such that we can subsequently identify $\mathbf{P}$, $\mathbf{B}$ and $\mathbf{C}_D$. It also implies that any valid permutation $\mathbf{P}$ will produce exactly the same distribution for $\mathbf{x}$.

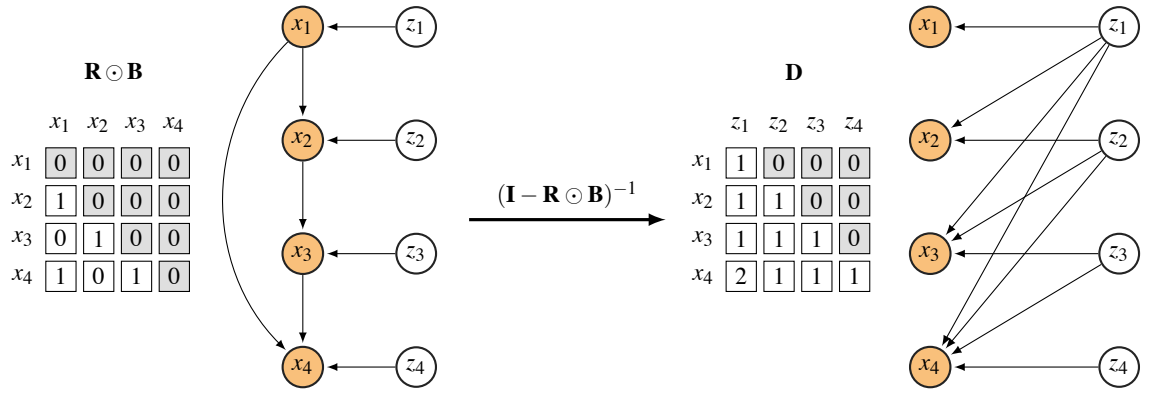


Figure 2: FM-DAG equivalence illustration. In the left side, a DAG model with four variables with corresponding connectivity matrix $\mathbf{R}$, $b_{ij} = 1$ when $r_{ij} = 1$ and $\mathbf{C}_D = \mathbf{I}$. In the right hand side, the equivalent factor model with mixing matrix $\mathbf{D}$. Note that the factor model is sparse even if its corresponding DAG is dense. The gray boxes in $\mathbf{D}$ and $\mathbf{R} \odot \mathbf{B}$ represent elements that must be zero by construction.

In the general case in Equation (1), $\mathbf{D} = (\mathbf{I} - \mathbf{B})^{-1}\mathbf{C}$ is of size $d \times (d+m)$. What we will show is that even if $\mathbf{D}$ is still identifiable, we can no longer obtain $\mathbf{B}$ and $\mathbf{C}$ uniquely unless we "tag" the model by requiring the distributions of driving signals $\mathbf{z}_D$ and latent signals $\mathbf{z}_L$ to differ. In order to illustrate why we get non-identifiability, we can write $\mathbf{x} = \mathbf{D}\mathbf{z}$ inverting $\mathbf{D}$ explicitly. For simplicity we consider $m = 1$ and $\mathbf{P} = \mathbf{I}$ but generalizing to $m > 1$ is straight forward

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_d \end{bmatrix} = \begin{bmatrix} c_{11} & 0 & 0 & \cdots & c_{1L} \\ b_{21}c_{11} & c_{22} & 0 & \cdots & b_{21}c_{1L} + c_{2L} \\ b_{31}c_{11} + b_{32}b_{21}c_{11} & b_{32}c_{22} & c_{33} & \cdots & b_{31}c_{1L} + b_{32}b_{21}c_{1L} + a_{32}c_{2L} + c_{3L} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{11} + \sum_{k=1}^{i-1} b_{ik}d_{k1} & \cdots & \cdots & \cdots & c_{iL} + \sum_{k=1}^{i-1} b_{ik}d_{kL} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{d+1} \end{bmatrix}.$$

We see from this equation that if all latent variables have the same distribution and c_{1L} is non-zero then we may exchange the first and last column in $\mathbf{D}$ to get two equivalent distributions with different elements for $\mathbf{B}$ and $\mathbf{C}$. The model is thus non-identifiable. If the first i elements in latent column of $\mathbf{C}$ are zero then the $(i+1)$-th and last column can be exchanged. Hoyer et al. (2008) made the same basic observation through a number of examples. Interestingly, we also see from the triangularity requirement of the "driving signal" part of $\mathbf{D}$ that $\mathbf{P}$ is actually identifiable despite the fact that $\mathbf{B}$ and $\mathbf{C}$ are not. To illustrate that the non-identifiability may lead to quite severe confusion about inferences, consider a model with only two observed variables $\mathbf{x} = [x_1, x_2]^\top$ and $c_{11} = c_{22} = 1$. Two different hypothesis $\{b_{21}, c_{1L}, c_{2L}\} = \{0, 1, 1\}$ and $\{b_{21}, c_{1L}, c_{2L}\} = \{1, 1, -1\}$ with graphs shown in Figure 3 have equivalent factor models written as

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_L \end{bmatrix} \text{ and } \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} z'_1 \\ z'_2 \\ z'_L \end{bmatrix}.$$

The two models above have the same mixing matrix $\mathbf{D}$, up to permutation of columns $\mathbf{P}_\mathrm{f}$. In general we expect the number of solutions with equivalent distribution may be as large as 2^m, corresponding

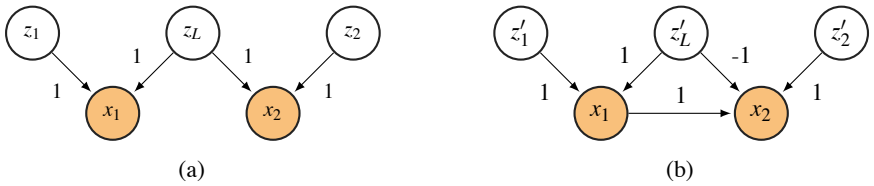


Figure 3: Two DAGs with latent variables. They are equivalent if $\mathbf{z}$ has the same distribution as $\mathbf{z}'$.

to the number of times a column of $\mathbf{D}$ from its latent part (last m columns) con be exchanged with a column from its observed part (first d columns). This readily assumes that the sparsity pattern in $\mathbf{D}$ is identified, which follows from the results of Kagan et al. (1973).

One way to get identifiability is to change the distributions $\mathbf{z}_D$ and $\mathbf{z}_L$ such that they differ and cannot be exchanged. Here it is not enough to change the scale of the variables, that is, variance for continuous variables, because this effect can be countered by rescaling $\mathbf{C}$ with $\mathbf{S}_\mathrm{f}$. So we need distributions that differ beyond rescaling. In our examples we use Laplace and the more heavy-tailed Cauchy for $\mathbf{z}_D$ and $\mathbf{z}_L$, respectively. This specification is not unproblematic in practical situations however it can be sometimes restrictive and prone to model mismatch issues. We nevertheless show one practical example which leads to sensible inferences.

In time series applications for example, it is natural to go beyond an i.i.d. model for $\mathbf{z}$. One may for example use a Gaussian process prior for each factor to get smoothness over time, that is, $z_{j1},\ldots,z_{jN}|\nu_j \sim \mathcal{N}(0,\mathbf{K}_{\nu_j})$, where $\mathbf{K}_{\nu_j}$ is the covariance matrix with elements $k_{j,nn'} = k_{\upsilon_j,n}(n,n')$ and $k_{\upsilon_j,n}(\cdot)$ is the covariance function. For the i.i.d. Gaussian model the source distribution is only identifiable up to an arbitrary rotation matrix $\mathbf{U}$, that is, the rotated factors $\mathbf{Uz}$ are still i.i.d. . We can show that contrary to the i.i.d. Gaussian model, the Gaussian process factor model is identifiable if the covariance functions differ. We need to show that $\widehat{\mathbf{Z}} = \mathbf{UZ}$ has a different covariance structure than $\mathbf{Z} = [\mathbf{z}_1 \ \ldots \ \mathbf{z}_N]$. We get $\overline{\mathbf{z}_n\mathbf{z}_{n'}^\top} = \mathrm{diag}(k_{1,nn'},\ldots,k_{d+m,nn'})$ and $\overline{\widehat{\mathbf{z}}_n\widehat{\mathbf{z}}_{n'}^\top} = \mathbf{U}\overline{\mathbf{z}_n\mathbf{z}_{n'}^\top}\mathbf{U}^\top = \mathbf{U}\mathrm{diag}(k_{1,nn'},\ldots,k_{d+m,nn'})\mathbf{U}^\top$ for the original and rotated variables, respectively. The covariances are indeed different and the model is thus identifiable if no covariance functions $k_{\upsilon_j,n}(n,n')$, $j = 1,\ldots,d+m$ are the same.

3. Prior Specification

In this section we provide a detailed description of the priors used for each one of the elements of our sparse linear identifiable model already defined in Equation (1). We start with $\boldsymbol{\varepsilon}$, the noise term that allow us to quantify the mismatch between a set of N observations $\mathbf{X} = [\mathbf{x}_1 \ \ldots \ \mathbf{x}_N]$ and the model itself. For this purpose, we use uncorrelated Gaussian noise components $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{\varepsilon}|\mathbf{0},\boldsymbol{\Psi})$ with conjugate inverse gamma priors for their variances as follows

$$\mathbf{X}|\mathbf{m},\boldsymbol{\Psi} \sim \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}_n|\mathbf{m},\boldsymbol{\Psi}) \ ,$$

$$\boldsymbol{\Psi}^{-1}|s_s,s_r \sim \prod_{i=1}^{d} \mathrm{Gamma}(\psi_i^{-1}|s_s,s_r) \ ,$$

where we have already marginalized out $\boldsymbol{\varepsilon}$, $\boldsymbol{\Psi}$ is a diagonal covariance matrix denoting uncorrelated noise across dimensions and $\mathbf{m}$ is the mean vector such that $\mathbf{m}_\mathrm{FM} = \mathbf{Cz}_n$ and $\mathbf{m}_\mathrm{DAG} = \mathbf{Bx}_n + \mathbf{Cz}_n$. In the noise covariance hyperprior, s_s and s_r are the shape and rate, respectively. The selection of

hyperparameters for $\mathbf{\Psi}$ should not be very critical as long as both "signal and noise" hypotheses are supported, that is, diffuse enough to allow for small values of ψ_i as well as for $\psi_i = 1$ (assuming that the data is standardized in advance). We set $s_s = 20$ and $s_r = 1$ in the experiments for instance. Another issue to consider when selecting s_s and s_r is the Bayesian analogue of the Heywood problem in which likelihood functions are bounded below away from zero as ψ_i tends to zero, hence inducing multi-modality in the posterior of ψ_i with one of the modes at zero. The latter can be avoided by specifying s_s and s_r such that the prior decays to zero at the origin, as we did above. It is well known, for example, that Heywood problems cannot be avoided using improper reference priors, $p(\psi_i) \propto 1/\psi_i$ (Martin and McDonald, 1975).

The remaining components of the model are described as it follows in five parts named sparsity, latent variables and driving signals, order search, allowing for correlated data and allowing for non-linearities. The first part addresses the interpretability of the model by means of parsimonious priors for $\mathbf{C}$ and $\mathbf{D}$. The second part describes the type of non-Gaussian distributions used on $\mathbf{z}$ in order to keep the model identifiable. The third part considers how a search over permutations of the observed variables can be used in order to handle the constraints imposed on matrix $\mathbf{R}$. The last two parts describe how introducing Gaussian process process priors in the model can be used to model non-independent observations and non-linear dependencies in the DAGs.

3.1 Sparsity

The use of sparse models will in many cases give interpretable results and is often motivated by the principle of parsimony. Also, in many application domains it is also natural from a prediction point of view to enforce sparsity because the number of explanatory variables may exceed the number of examples by orders of magnitude. In regularized maximum likelihood type formulations of learning (maximum a-posteriori) it has become popular to use one-norm (L_1) regularization for example to achieve sparsity (Tibshirani, 1996). In the fully Bayesian inference setting (with averaging over variables), the corresponding Laplace prior will not lead to sparsity because it is very unlikely for a posterior summary like the mean, median or mode to be estimated as exactly zero even asymptotically. The same effect can be expected from any continuous distribution used for sparsity like Student's t, α-stable and bimodal priors (continuous slab and spike priors, Ishwaran and Rao, 2005). Exact zeros can only be achieved by placing a point mass at zero, that is, explicitly specifying that the variable at hand is zero or not with some probability. This has motivated the introduction of many variants over the years of so-called slab and spike priors consisting of two component mixtures of a continuous part and a δ-function at zero (Lempers, 1971; Mitchell and Beauchamp, 1988; George and McCulloch, 1993; Geweke, 1996; West, 2003). In this paradigm, the columns of matrices $\mathbf{C}$ or $\mathbf{B}$ encode respectively, the connectivity of a factor or the set of parents associated to an observed variable. It is natural then to share information across elements in column j by assuming a common sparsity level $1-\nu_j$, suggesting the following hierarchy

$$\begin{aligned} c_{ij}|q_{ij},\cdot &\sim (1-q_{ij})\delta(c_{ij}) + q_{ij}\text{Cont}(c_{ij}|\cdot) \ , \\ q_{ij}|\nu_j &\sim \text{Bernoulli}(q_{ij}|\nu_j) \ , \\ \nu_j|\beta_m,\beta_p &\sim \text{Beta}(\nu_j|\beta_p\beta_m, \beta_p(1-\beta_m)) \ , \end{aligned} \tag{2}$$

where $\mathbf{Q}$, the binary matrix in Equation (1) appears naturally, $\delta(\cdot)$ is a Dirac δ-function, $\text{Cont}(\cdot)$ is the continuous slab component, $\text{Bernoulli}(\cdot)$ and $\text{Beta}(\cdot)$ are Bernoulli and beta distributions, respectively. Reparameterizing the beta distribution as $\text{Beta}(\nu_j|\alpha\beta/m, \beta)$ and taking the number of

columns m of $\mathbf{Q} \odot \mathbf{C}$ to infinity, leads to the non-parametric version of the slab and spike model with a so-called Indian buffet process prior over the (infinite) masking matrix $\mathbf{Q} = \{q_{ij}\}$ (Ghahramani et al., 2006). Note also that $q_{ij}|\nu_j$ is mainly used for clarity to make the binary indicators explicit, nevertheless in practice we can work directly with $c_{ij}|\nu_j, \cdot \sim (1-\nu_j)\delta(c_{ij}) + \nu_j \text{Cont}(c_{ij}|\cdot)$ because q_{ij} can be marginalized out.

As illustrated and pointed out by Lucas et al. (2006) and Carvalho et al. (2008) the model with a shared beta-distributed sparsity level per factor introduces the undesirable side-effect that there is strong co-variation between the elements in each column of the masking matrix. For example, in high dimensions we might expect that only a finite number of elements are non-zero, implying a prior favoring a very high sparsity rate $1-\nu_j$. Because of the co-variation, even the parameters that are clearly non-zero will have a posterior probability of being non-zero, $p(q_{ij} = 1|\mathbf{x}, \cdot)$, quite spread over the unit interval. Conversely, if our priors do not favor sparsity strongly, then the opposite situation will arise and the solution will become completely dense. In general, it is difficult to set the hyperparameters to achieve a sensible sparsity level. Ideally, we would like to have a model with a high sparsity level with high certainty about the non-zero parameters. We can achieve this by introducing a sparsity parameter η_{ij} for each element of $\mathbf{C}$ which has a mixture distribution with exactly this property

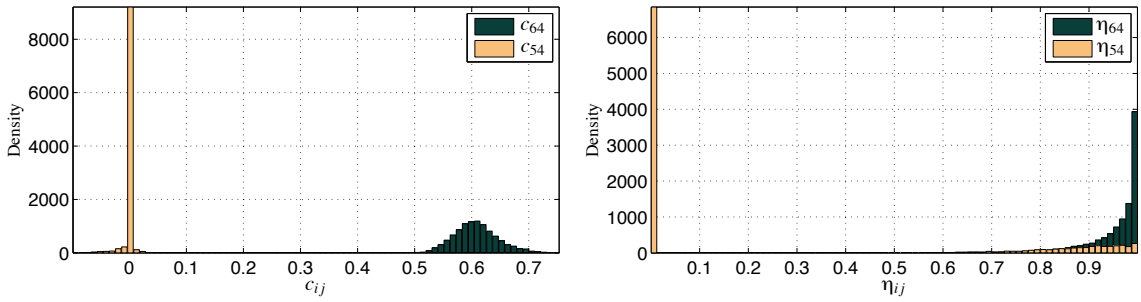
$$\begin{aligned} q_{ij}|\eta_{ij} &\sim \text{Bernoulli}(q_{ij}|\eta_{ij}) \,, \\ \eta_{ij}|\nu_j, \alpha_p, \alpha_m &\sim (1-\nu_j)\delta(\eta_{ij}) + \nu_j \text{Beta}(\eta_{ij}|\alpha_p\alpha_m, \alpha_p(1-\alpha_m)) \,. \end{aligned} \tag{3}$$

The distribution over η_{ij} expresses that we expect parsimony: either η_{ij} is zero exactly (implying that q_{ij} and c_{ij} are zero) or non-zero drawn from a beta distribution favoring high values, that is, q_{ij} and c_{ij} are non-zero with high probability. We use $\alpha_p = 10$ and $\alpha_m = 0.95$ which has mean $\alpha_m = 0.95$ and variance $\alpha_m(1-\alpha_m)/(1+\alpha_p) \approx 0.086$. The expected sparsity rate of the modified model is $(1-\alpha_m)(1-\nu_j)$. This model has the additional advantage that the posterior distribution of η_{ij} directly measures the distribution of $p(q_{ij} = 1|\mathbf{x}, \cdot)$. This is therefore the statistic for ranking/selection purposes. Besides, we may want to reject interactions with high uncertainty levels when the probability of $p(q_{ij} = 1|\mathbf{x}, \cdot)$ is less or very close to the expected value, $\alpha_m(1-\nu_j)$.

To complete the specification of the prior, we let the continuous slab part in Equation (2) be Gaussian distributed with inverse gamma prior on its variance. In addition, we scale the variances with ψ_i as

$$\begin{aligned} \text{Cont}(c_{ij}|\psi_i, \tau_{ij}) &= \mathcal{N}(c_{ij}|0, \psi_i\tau_{ij}) \,, \\ \tau_{ij}^{-1}|t_s, t_r &\sim \text{Gamma}(\tau_{ij}^{-1}|t_s, t_r) \,. \end{aligned} \tag{4}$$

This scaling makes the model easier to specify and tend to have better mixing properties (see Park and Casella, 2008). The slab and spike for $\mathbf{B}$ (DAG) is obtained from Equations (2), (3) and (4) by simply replacing c_{ij} with b_{ij} and q_{ij} with r_{ij}. As already mentioned, we use $\alpha_p = 10$ and $\alpha_m = 0.95$ for the hierarchy in Equation (3). For the column-shared parameter ν_j defined in Equation (2) we set the precision to $\beta_p = 100$ and consider the mean values for factor models and DAGs separately. For the factor model we set a diffuse prior by making $\beta_m = 0.9$ to reflect that some of the factors can be in general nearly dense or empty. For the DAG we consider two settings, if we expect to obtain dense graphs we set $\beta_m = 0.99$, otherwise we set $\beta_m = 0.1$. Both settings can produce sparse graphs, however smaller values of β_m increase the overall sparsity rate and the gap between $p(r_{ij} = 0)$ and $p(r_{ij} = 1)$. A large separation between these two probabilities makes interpretation easier and also



(a) (b)

Figure 4: Slab and spike prior example. (a) Posterior unnormalized densities for the magnitude of two particular elements of $\mathbf{C}$. (b) Posterior density for $\eta_{ij} = p(c_{ij} \neq 0|\mathbf{x}, \cdot)$. Here, $c_{64} \neq 0$ and $c_{54} = 0$ correspond to elements of the mixing matrix from the experiment shown in Figure 8.

helps to spot non-zeros (edges) with high uncertainty. The hyperparameters for the variance of the non-zero elements of $\mathbf{B}$ and $\mathbf{C}$ are set to get a diffuse prior distribution bounded away from zero ($t_s = 2$ and $t_r = 1$), to allow for a better separation between slab and spike components. For the particular case of $\mathbf{C}_L$, in principle the prior should not have support on zero at all, that is, the driving signal should not vanish, however for simplicity we allow this anyway as it has not given any problems in practice. Figure 4 shows a particular example of the posterior, $p(c_{ij}, \eta_{ij}|\mathbf{x}, \cdot)$ for two elements of $\mathbf{C}$ under the prior just described. In the example, $c_{64} \neq 0$ with high probability according to η_{ij}, whereas c_{54} is almost certainly zero since most of its probability mass is located exactly at zero, with some residual mass on the vicinity of zero, in Figure 4(a). In the one level hierarchy Equation (2) sparsity parameters are shared, $\eta_{64} = \eta_{54} = \nu_4$. The result would then be less parsimonious with the posterior density of ν_4 being spread in the unit interval with a single mode located close to β_m.

3.2 Latent Variables and Driving Signals

We consider two different non-Gaussian—heavy-tailed priors for $\mathbf{z}$, in order to obtain identifiable factor models and DAGs. A wide class of continuous, unimodal and symmetric distributions in one dimension can be represented as infinite scale mixtures of Gaussians, which are very convenient for Gibbs-sampling-based inference. We focus on Student's t and Laplace distributions which have the following mixture representation (Andrews and Mallows, 1974)

$$\text{Laplace}(z|\mu,\lambda) = \int_0^\infty \mathcal{N}(z|\mu,\upsilon)\text{Exponential}(\upsilon|\lambda^2)d\upsilon \,, \tag{5}$$

$$t(z|\mu,\theta,\sigma^2) = \int_0^\infty \mathcal{N}(z|\mu,\upsilon\sigma^2)\text{Gamma}\left(\upsilon^{-1}\middle|\frac{\theta}{2},\frac{\theta}{2}\right)d\upsilon \,, \tag{6}$$

where $\lambda > 0$ is the rate, $\sigma^2 > 0$ the scale, $\theta > 0$ is the degrees of freedom, and the distributions have exponential and gamma mixing densities accordingly. For varying degrees of freedom θ, the t distribution can interpolate between very heavy-tailed (power law and Cauchy when $\theta = 1$) and very light tailed, that is, it becomes Gaussian when the degrees of freedom approaches infinity. The Laplace (or bi-exponential) distribution has tails which are intermediate between a t (with finite

degrees of freedom) and a Gaussian. In this sense, the *t* distribution is more flexible but requires more careful selection of its hyperparameters because the model may become non-identifiable in the large θ limit (Gaussian).

An advantage of the Laplace distribution is that we can fix its parameter $\lambda = 1$ and let the model learn the appropriate scaling from $\mathbf{C}$ in Equation (1). If we use the pure DAG model, we will need to have a hyperprior for λ^2 in order to learn the variances of the latent variables/driving signals, as in Henao and Winther (2009). A hierarchical prior for the degrees of freedom in the *t* distribution is not easy to specify because there is no conjugate prior available with a standard closed form. Although a conjugate prior exists, is not straightforward to sample from it, since numerical integration must be used to compute its normalization constant. Another possibility is to treat θ as a discrete variable so computing the normalizing constant becomes straight forward.

Laplace and Student's *t* are not the only distributions admitting scale mixture representation. This mean that any other compatible type can be used as well, if the application requires it, and without considerable additional effort. Some examples include the logistic distribution (Andrews and Mallows, 1974), the stable family (West, 1987) and skewed versions of heavy-tailed distributions (Branco and Dey, 2001). Another natural extension to the mixtures scheme could be, for example, to set the mean of each component to arbitrary values and let the number of components be an infinite sum, thus ending up providing each factor with a Dirichlet process prior. This might be useful for cases when the latent factors are expected to be scattered in clusters due to the presence of subgroups in the data, as was shown by Carvalho et al. (2008).

3.3 Order Search

We need to infer the order of the variables in the DAG to meet the constraints imposed on $\mathbf{R}$ in Section 2. The most obvious way is to try to solve this task by inferring all parameters $\{\mathbf{P}, \mathbf{B}, \mathbf{C}, \mathbf{z}, \boldsymbol{\varepsilon}\}$ by a Markov chain Monte Carlo (MCMC) method such as Gibbs sampling. However, algorithms for searching over variable order prefer to work with models for which parameters other than $\mathbf{P}$ can be marginalized analytically (see Friedman and Koller, 2003; Teyssier and Koller, 2005). For our model, where we cannot marginalize analytically over $\mathbf{B}$ (due to $\mathbf{R}$ being binary), estimating $\mathbf{P}$ and $\mathbf{B}$ by Gibbs sampling would mean that we had to propose a new $\mathbf{P}$ for fixed $\mathbf{B}$. For example, exchanging the order of two variables would mean that they also exchange parameters in the DAG. Such a proposal would have very low acceptance, mainly as a consequence of the size of the search space and thus very poor mixing. In fact, for a given d number of variables there are $d!$ possible orderings $\mathbf{P}$, while there are $d!2^{(d(d+2m-1))/2}$ possible structures for $\{\mathbf{P}, \mathbf{B}, \mathbf{C}\}$. We therefore opt for an alternative strategy by exploiting the equivalence between factor models and DAGs shown in Section 2.1. In particular for $m = 0$, since $\mathbf{B}$ can be permuted to strictly lower triangular, then $\mathbf{D} = (\mathbf{I} - \mathbf{B})^{-1}\mathbf{C}_D$ can be permuted to triangular. This means that we can perform inference for the factor model to obtain $\mathbf{D}$ while searching in parallel for a set of permutations $\mathbf{P}$ that are in good agreement (in a probabilistic sense) with the triangular requirement of $\mathbf{D}$. Such a set of orderings is found during the inference procedure of the factor model. To set up the stochastic search, we need to modify the factor model slightly by introducing separate data (row) and factor (column) permutations, $\mathbf{P}$ and $\mathbf{P}_\mathrm{f}$ to obtain $\mathbf{x} = \mathbf{P}^\top \mathbf{D}\mathbf{P}_\mathrm{f}\mathbf{z} + \boldsymbol{\varepsilon}$. The reason for using two different permutation matrices, rather than only one like in the definition of the DAG model, is that we need to account for the permutation freedom of the factor model (see Section 2.1). Using the same permutation for row and column would thus require an additional step to identify the columns in the factor model. We make inference for

the unrestricted factor model, but propose $\mathbf{P}^\star$ and $\mathbf{P}_\mathrm{f}^\star$ independently according to $q(\mathbf{P}^\star|\mathbf{P})q(\mathbf{P}_\mathrm{f}^\star|\mathbf{P}_\mathrm{f})$. Both distributions draw a new permutation matrix by exchanging two randomly chosen elements, for example, the order may change as $[x_1,x_2,x_3,x_4]^\top \rightarrow [x_1,x_4,x_3,x_2]^\top$. In other words, the proposals $q(\mathbf{P}^\star|\mathbf{P})$ and $q(\mathbf{P}_\mathrm{f}^\star|\mathbf{P}_\mathrm{f})$ are uniform distributions over the space of transpositions for $\mathbf{P}$ and $\mathbf{P}_\mathrm{f}$. Assuming we have no a-priori preferred ordering, we may use a Metropolis-Hastings (M-H) acceptance probability $\min(1,\xi_{\rightarrow\star})$ with $\xi_{\rightarrow\star}$ as a simple ratio of likelihoods with the permuted $\mathbf{D}$ masked to match the triangularity assumption. Formally, we use the binary mask $\mathbf{M}$ (containing zeros above the diagonal of its d first columns) and write

$$\xi_{\rightarrow\star} = \frac{\mathcal{N}(\mathbf{X}|(\mathbf{P}^\star)^\top(\mathbf{M}\odot\mathbf{P}^\star\mathbf{D}(\mathbf{P}_\mathrm{f}^\star)^\top)\mathbf{P}_\mathrm{f}^\star\mathbf{Z},\mathbf{\Psi})}{\mathcal{N}(\mathbf{X}|\mathbf{P}^\top(\mathbf{M}\odot\mathbf{P}\mathbf{D}\mathbf{P}_\mathrm{f}^\top)\mathbf{P}_\mathrm{f}\mathbf{Z},\mathbf{\Psi})}\,, \tag{7}$$

where $\mathbf{M}\odot\mathbf{D}$ is the masked $\mathbf{D}$ and $\mathbf{Z} = [\mathbf{z}_1\ \ldots\mathbf{z}_N]$. The procedure can be seen as a simple approach for generating hypotheses about good orderings, producing close to triangular versions of $\mathbf{D}$, in a model where the slab and spike prior provide the required bias towards sparsity. Once the inference is done, we end up having an estimate for the desired distribution over permutations $\mathbf{P} = \sum_i^{d!}\pi_i\delta_{\mathbf{P}_i}$, where $\boldsymbol{\pi} = [\pi_1\ \pi_2\ \ldots]$ is a sparse vector containing the probability for $\mathbf{P} = \mathbf{P}_i$, which in our case is proportional to the number of times permutation $\mathbf{P}_i$ was accepted by the M-H update during inference. Note that $\mathbf{P}_\mathrm{f}$ is just a nuisance variable that does not need to be stored or summarized.

3.4 Allowing for Correlated Data (CSLIM)

For the case where independence of observed variables cannot be assumed, for instance due to (time) correlation or smoothness, the priors discussed before for the latent variables and driving signals do not really apply anymore, however the only change we need to make is to allow elements in rows of $\mathbf{Z}$ to correlate. We can assume then independent Gaussian process (GP) priors for each latent variable instead of scale mixtures of Gaussians, to obtain what we have called correlated sparse linear identifiable modeling (CSLIM). For a set of N realizations of variable j we set

$$z_{j1},\ldots,z_{jN}|\upsilon_j \sim \mathrm{GP}(z_{j1},\ldots,z_{jN}|k_{\upsilon_j,n}(\cdot))\,, \tag{8}$$

where the covariance function has the form $k_{\upsilon_j,n}(n,n') = \exp(-\upsilon_j(n-n')^2)$, $\{n,n'\}$ is a pair of observation indices or time points and υ_j is the length scale controlling the overall level of correlation allowed for each variable (row) in $\mathbf{Z}$. Conceptually, Equation (8) implies that each latent variable j is sampled from a function and the GP acts as a prior over continuous functions. Since such a length scale is very difficult to set just by looking at the data, we further place priors on υ_j as

$$\upsilon_j|u_s,\kappa \sim \mathrm{Gamma}(\upsilon_j|u_s,\kappa)\,,\quad \kappa|k_s,k_r \sim \mathrm{Gamma}(\kappa|k_s,k_r)\,. \tag{9}$$

Given that the conditional distribution of $\boldsymbol{\upsilon} = [\upsilon_1,\ldots,\upsilon_m]$ is not of any standard form, Metropolis-Hastings updates are used. In the experiments we use that $u_s = k_s = 2$ and $k_r = 0.02$. The details concerning inference for this model are given in Appendix A.

It is also possible to easily expand the possible applications of GP priors in this context by, for instance, using more structured covariance functions through scale mixture of Gaussian representations to obtain a prior distribution for continuous functions with heavy-tailed behavior—a t-processes (Yu et al., 2007), or learning the covariance function as well using inverse Wishart hyperpriors.

3.5 Allowing for Non-linearities (SNIM)

Provided that we know the true ordering of the variables, that is, $\mathbf{P}$ is known then $\mathbf{B}$ is surely strictly lower triangular. It is very easy to allow for non-linear interactions in the DAG model from Equation (1) by rewriting it as

$$\mathbf{Px} = (\mathbf{R} \odot \mathbf{B})\mathbf{Py} + (\mathbf{Q} \odot \mathbf{C})\mathbf{z} + \boldsymbol{\varepsilon} \ , \tag{10}$$

where $\mathbf{y} = [y_1, \ldots, y_d]^\top$ and $y_{i1}, \ldots, y_{iN} | \upsilon_i \sim \mathrm{GP}(y_{i1}, \ldots, y_{iN} | k_{\upsilon_i, x}(\cdot))$ has a Gaussian process prior with for instance, but not limited to, a stationary covariance function like $k_{\upsilon_i, x}(\mathbf{x}, \mathbf{x}') = \exp(-\upsilon_i(\mathbf{x} - \mathbf{x}')^2)$, similar to Equation (8) and with the same hyperprior structure as in Equation (9). This is a straight forward extension that we call sparse non-linear multivariate modeling (SNIM) that is in spirit similar to Friedman and Nachman (2000), Hoyer et al. (2009), Zhang and Hyvärinen (2009), Zhang and Hyvärinen (2010) and Tillman et al. (2009), however instead of treating the inherent multiple regression problem in Equation (10) and the conditional independence of the observed variables independently, we proceed within our proposed framework by letting the multiple regressor be sparse, thus the conditional independences are encoded through $\mathbf{R}$. The main limitation of the model in Equation (10) is that if the true ordering of the variables is unknown, the exhaustive enumeration of $\mathbf{P}$ is needed. This means that this could be done for very small networks, for example, up to 5 or 6 variables. In principle, an ordering search procedure for the non-linear model only requires the latent variables $\mathbf{z}$ to have Gaussian process priors as well. The main difficulty is that in order to build covariance functions for $\mathbf{z}$ we need a set of observations that are not available because $\mathbf{z}$ is latent.

4. Model Comparison

Quantitative model comparison between factor models and DAGs is a key ingredient in SLIM. The joint probability of data $\mathbf{X}$ and parameters for the factor model part in Equation (1) is

$$p(\mathbf{X}, \mathbf{C}, \mathbf{Z}, \boldsymbol{\varepsilon}, \cdot) = p(\mathbf{X}|\mathbf{C}, \mathbf{Z}, \boldsymbol{\varepsilon}) p(\mathbf{C}|\cdot) p(\mathbf{Z}|\cdot) p(\boldsymbol{\varepsilon}) p(\cdot) \ ,$$

where $(\cdot)$ indicates additional parameters in the hierarchical model. Formally the Bayesian model selection yardstick, the marginal likelihood for model $\mathcal{M}$

$$p(\mathbf{X}|\mathcal{M}) = \int p(\mathbf{X}|\boldsymbol{\Theta}, \mathbf{Z}) p(\boldsymbol{\Theta}|\mathcal{M}) p(\mathbf{Z}|\mathcal{M}) d\boldsymbol{\Theta} d\mathbf{Z} \ ,$$

can be obtained by marginalizing the joint over the parameters $\boldsymbol{\Theta}$ and latent variables $\mathbf{Z}$. Computationally this is a difficult task because the marginal likelihood cannot be written as an average over the posterior distribution in a simple way. It is still possible using MCMC methods, for example by partitioning of the parameter space and multiple chains or thermodynamic integration (see Chib, 1995; Neal, 2001; Murray, 2007; Friel and Pettitt, 2008), but in general it must be considered as computationally expensive and non-trivial. On the other hand, evaluating the likelihood on a test set $\mathbf{X}^\star$, using predictive densities $p(\mathbf{X}^\star|\mathbf{X}, \mathcal{M})$ is simpler from a computational point of view because it can be written in terms of an average over the posterior of the *intensive variables*, $p(\mathbf{C}, \boldsymbol{\varepsilon}, \cdot|\mathbf{X})$ and the prior distribution of the *extensive variables* associated with the test points,[2] $p(\mathbf{Z}^\star|\cdot)$ as

$$\mathcal{L}_{\mathrm{FM}} \stackrel{\mathrm{def}}{=} p(\mathbf{X}^\star|\mathbf{X}, \mathcal{M}_{\mathrm{FM}}) = \int p(\mathbf{X}^\star|\mathbf{Z}^\star, \boldsymbol{\Theta}_{\mathrm{FM}}, \cdot) p(\mathbf{Z}^\star|\cdot) p(\boldsymbol{\Theta}_{\mathrm{FM}}, \cdot|\mathbf{X}) d\mathbf{Z}^\star d\boldsymbol{\Theta}_{\mathrm{FM}} d(\cdot) \ , \tag{11}$$

2. Intensive means not scaling with the sample size. Extensive means scaling with sample size in this case the size of the test sample.

where $\mathbf{\Theta}_{\text{FM}} = \{\mathbf{C}, \boldsymbol{\varepsilon}\}$. This average can be approximated by a combination of standard sampling and exact marginalization using the scale mixture representation of the heavy-tailed distributions presented in Section 3.2. For the full DAG model in Equation (1), we will not average over permutations $\mathbf{P}$ but rather calculate the test likelihood for a number of candidates $\mathbf{P}^{(1)}, \ldots, \mathbf{P}^{(c)}, \ldots$ as

$$\begin{aligned}\mathcal{L}_{\text{DAG}} &\stackrel{\text{def}}{=} p(\mathbf{X}^\star|\mathbf{P}^{(c)}, \mathbf{X}, \mathcal{M}_{\text{DAG}}) \ , \\ &= \int p(\mathbf{X}^\star|\mathbf{P}^{(c)}, \mathbf{X}, \mathbf{Z}^\star, \mathbf{\Theta}_{\text{DAG}}, \cdot) p(\mathbf{Z}^\star|\cdot) p(\mathbf{\Theta}_{\text{DAG}}, \cdot|\mathbf{X}) d\mathbf{Z}^\star d\mathbf{\Theta}_{\text{DAG}} d(\cdot) \ , \end{aligned} \tag{12}$$

where $\mathbf{\Theta}_{\text{DAG}} = \{\mathbf{B}, \mathbf{C}, \boldsymbol{\varepsilon}\}$. We use sampling to compute the test likelihoods in Equations (11) and (12). With Gibbs, we draw samples from the posterior distributions $p(\mathbf{\Theta}_{\text{FM}}, \cdot|\mathbf{X})$ and $p(\mathbf{\Theta}_{\text{DAG}}, \cdot|\mathbf{X})$, where $(\cdot)$ is shorthand for example for the degrees of freedom θ, if Student t distributions are used. The average over the extensive variables associated with the test points $p(\mathbf{Z}^\star|\cdot)$ is slightly more complicated because naively drawing samples from $p(\mathbf{Z}^\star|\cdot)$ results in an estimator with high variance—for $\psi_i \ll \upsilon_{jn}$. Instead we exploit the infinite mixture representation to marginalize exactly $\mathbf{Z}^\star$ and then draw samples in turn for the scale parameters. Omitting the permutation matrices for clarity, in general we get

$$\begin{aligned} p(\mathbf{X}^\star|\mathbf{\Theta}, \cdot) &= \int p(\mathbf{X}^\star|\mathbf{Z}^\star, \mathbf{\Theta}, \cdot) p(\mathbf{Z}^\star|\cdot) d\mathbf{Z}^\star \ , \\ &= \prod_n \int \mathcal{N}(\mathbf{x}_n^\star|\mathbf{m}_n, \mathbf{\Sigma}_n) \prod_j p(\upsilon_{jn}|\cdot) d\upsilon_{jn} \approx \frac{1}{N_{\text{rep}}} \prod_n \sum_r^{N_{\text{rep}}} \mathcal{N}(\mathbf{x}_n^\star|\mathbf{m}_n, \mathbf{\Sigma}_n) \ , \end{aligned}$$

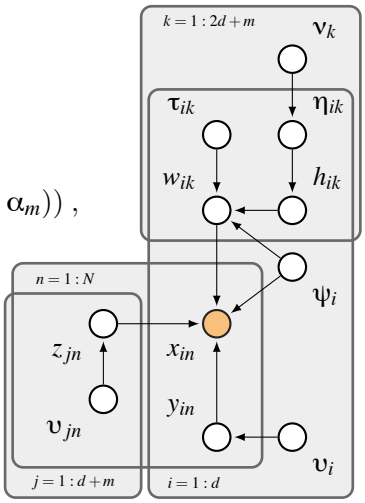
where N_{rep} is the number of samples generated to approximate the intractable integral ($N_{\text{rep}} = 500$ in the experiments). For the factor model $\mathbf{m}_n = \mathbf{0}$ and $\mathbf{\Sigma}_n = \mathbf{C}_D \mathbf{U}_n \mathbf{C}_D^\top + \mathbf{\Psi}$. For the DAG, $\mathbf{m}_n = \mathbf{B}\mathbf{x}_n^\star$ and $\mathbf{\Sigma}_n = \mathbf{C}\mathbf{U}_n\mathbf{C}^\top + \mathbf{\Psi}$. The covariance matrix $\mathbf{U}_n = \text{diag}(\upsilon_{1n}, \ldots, \upsilon_{(d+m)n})$ with elements υ_{jn}, is sampled directly from the prior, accordingly. Once we have computed $p(\mathbf{X}^\star|\mathbf{\Theta}_{\text{FM}}, \cdot)$ for the factor model and $p(\mathbf{X}^\star|\mathbf{\Theta}_{\text{DAG}}, \cdot)$ for the DAG, we can use them to average over $p(\mathbf{\Theta}_{\text{FM}}, \cdot|\mathbf{X},)$ and $p(\mathbf{\Theta}_{\text{DAG}}, \cdot|\mathbf{X})$ to obtain the predictive densities $p(\mathbf{X}^\star|\mathbf{X}, \mathcal{M}_{\text{FM}})$ and $p(\mathbf{X}^\star|\mathbf{X}, \mathcal{M}_{\text{DAG}})$, respectively.

For the particular case in which $\mathbf{X}$ and consequently $\mathbf{Z}$ are correlated variables—CSLIM, we use a slightly different procedure for model comparison. Instead of using a test set, we randomly remove some proportion of the elements of $\mathbf{X}$ and perform inference with missing values, then we summarize the likelihood on the missing values. In particular, for the factor model we use $\mathbf{M}_{\text{miss}} \odot \mathbf{X} = \mathbf{M}_{\text{miss}} \odot (\mathbf{Q}_L \odot \mathbf{C}_L\mathbf{Z} + \boldsymbol{\varepsilon})$ where $\mathbf{M}_{\text{miss}}$ is a binary masking matrix with zeros corresponding to test points, that is, the missing values. See details in Appendix A. Note that this scheme is not exclusive to CSLIM thus can be also used with SLIM or when the observed data contain actual missing values.

5. Model Overview and Practical Details

The three models described in the previous section namely SLIM, CSLIM and SNIM can be summarized as a graphical model and as a probabilistic hierarchy as follows

$$
\begin{aligned}
\mathbf{x}_n|\mathbf{W},\mathbf{y}_n,\mathbf{z}_n,\mathbf{\Psi} &\sim \mathcal{N}(\mathbf{x}_n|\mathbf{W}[\mathbf{y}_n\ \mathbf{z}_n]^\top,\mathbf{\Psi})\ ,\quad \mathbf{W}=[\mathbf{B}\ \mathbf{C}]\ ,\\
\psi_i^{-1}|s_s,s_r &\sim \text{Gamma}(\psi_i^{-1}|s_s,s_r)\ ,\\
w_{ik}|h_{ik},\psi_i,\tau_{ik} &\sim (1-h_{ik})\delta_0(w_{ik})+h_{ik}\mathcal{N}(w_{ik}|0,\psi_i\tau_{ik})\ ,\\
h_{ik}|\eta_{ik} &\sim \text{Bernoulli}(h_{ik}|\eta_{ik})\ ,\quad \mathbf{H}=[\mathbf{R}\ \mathbf{Q}]\ ,\\
\eta_{ik}|\nu_k,\alpha_p,\alpha_m &\sim (1-\nu_k)\delta(\eta_{ik})+\nu_k\text{Beta}(\eta_{ik}|\alpha_p\alpha_m,\alpha_p(1-\alpha_m))\ ,\\
\nu_k|\beta_m,\beta_p &\sim \text{Beta}(\nu_k|\beta_p\beta_m,\beta_p(1-\beta_m))\ ,\\
\tau_{ik}^{-1}|t_s,t_r &\sim \text{Gamma}(\tau_{ik}^{-1}|t_s,t_r)\ ,\\
z_{j1},\ldots,z_{jN}|\upsilon &\sim \begin{cases}\prod_n \mathcal{N}(z_{jn}|0,\upsilon_{jn})\ , & \text{(SLIM)}\\ \text{GP}(z_{j1},\ldots,z_{jN}|k_{\upsilon_j,n}(\cdot))\ , & \text{(CSLIM)}\end{cases}\\
y_{i1},\ldots,y_{iN}|\upsilon &\sim \begin{cases}x_{i1},\ldots,x_{iN}\ , & \text{(SLIM)}\\ \text{GP}(y_{i1},\ldots,y_{iN}|k_{\upsilon_i,x}(\cdot))\ , & \text{(SNIM)}\end{cases}
\end{aligned}
$$



where we have omitted $\mathbf{P}$ and the hyperparameters in the graphical model. Latent variable and driving signal parameters υ can have one of several priors: Exponential$(\upsilon|\lambda^2)$ (Laplace), Gamma$(\upsilon^{-1}|\theta/2,\theta/2)$ (Student's t) or Gamma$(\upsilon|u_s,\kappa)$ (GP), see Equations (5), (6) and (9), respectively. The latent variables/driving signals z_{jn} and the mixing/connectivity matrices with elements c_{ij} or b_{ij} are modeled independently. Each element in $\mathbf{B}$ and $\mathbf{C}$ has its own slab variance τ_{ij} and probability of being non-zero η_{ij}. Moreover, there is a shared sparsity rate per column ν_k. Variables υ_{jn} are variances if z_{jn} use a scale mixture of Gaussian's representation, or length scales in the GP prior case. Since we assume no sparsity for the driving signals, $\eta_{ik}=1$ for $d+i=k$ and $\eta_{ik}=0$ for $d+i\neq k$. In addition, we can recover the pure DAG by making $m=0$ and the standard factor model by making instead $\eta_{ik}=0$ for $k\leq 2d$. All the details for the Gibbs sampling based inference are summarized in appendix A.

5.1 Proposed Workflow

We propose the workflow shown in Figure 1 to integrate all elements of SLIM, namely factor model and DAG inference, stochastic order search and model selection using predictive densities.

1. Partition the data into $\{\mathbf{X},\mathbf{X}^\star\}$.

2. Perform inference on the factor model and stochastic order search. One Gibbs sampling update consists of computing the conditional posteriors in Equations (13), (14), (15), (16), (17), (18) and (19) in sequence, followed by several repetitions (we use 10) of the M-H update in Equation (7) for the permutation matrices $\mathbf{P}$ and $\mathbf{P}_\text{f}$.

3. Summarize the factor model, mainly $\mathbf{C}$, $\{\eta_{ij}\}$ and $\mathcal{L}_\text{FM}$ using quantiles $(0.025, 0.5$ and $0.975)$.

4. Summarize the orderings, $\mathbf{P}$. Select the top m_top candidates according to their frequency during inference in step 2.

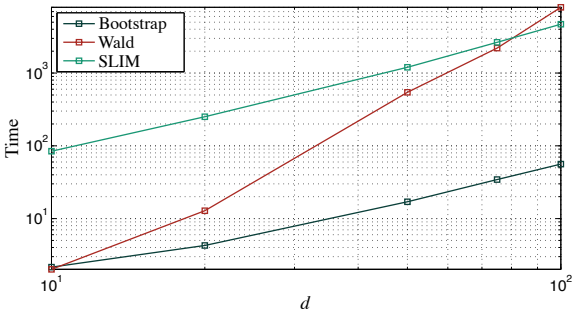


Figure 5: Runtime comparison.

5. Perform inference on the DAGs for each one of the ordering candidates, $\mathbf{P}^{(1)}, \ldots, \mathbf{P}^{(m_{\text{top}})}$ using Gibbs sampling by computing Equations (13), (14), (15), (16), (17), (18) and (19) in sequence, up to minor changes described in Appendix A.

6. Summarize the DAGs, $\mathbf{B}$, $\mathbf{C}_L$, $\{\eta_{ik}\}$ and $\mathcal{L}_{\text{DAG}}^{(1)}, \ldots, \mathcal{L}_{\text{DAG}}^{(m_{\text{top}})}$ using quantiles (0.025, 0.5 and 0.975). Note that $\{\eta_{ik}\}$ contains non-zero probabilities for $\mathbf{R}$ and $\mathbf{Q}$ corresponding to $\mathbf{B}$ and $\mathbf{C}_L$, respectively.

We use medians to summarize all quantities in our model because $\mathbf{D}, \mathbf{B}$ and $\{\eta_{ik}\}$ are bimodal while the remaining variables are in general skewed posterior distributions. Inference with GP priors for time series data (CSLIM) or non-linear DAGs (SNIM) is fairly similar to the i.i.d. case, see Appendix A for details. Source code for SLIM and all its variants proposed so far has been made available at `http://cogsys.imm.dtu.dk/slim/` as Matlab scripts.

5.2 Computational Cost

The cost of running the linear DAG with latent variables or the factor model is roughly the same, that is, $O(N_s d^2 N)$ where N_s is the total number of samples including the burn-in period. The memory requirements on the other hand are approximately $O(N_p d^2)$ if all the samples after the burn-in period N_p are stored. This means that the inference procedures scale reasonably well if N_s is kept in the lower ten thousands. The non-linear version of the DAG is considerably more expensive due to the GP priors, hence the computational cost rises up to $O(N_s(d-1)N^3)$.

The computational cost of LiNGAM, being the closest to our linear models, is mainly dependent on the statistic used to prune/select the model. Using bootstrapping results in $O(N_b^3)$, where N_b is the number of bootstrap samples. The Wald statistic leads to $O(d^6)$, while Wald with χ^2 second order model fit test amounts to $O(d^7)$. As for the memory requirements, bootstrapping is very economic whereas Wald-based statistics require $O(d^6)$.

The method for non-linear DAGs described in Hoyer et al. (2009) is defined for a pair of variables, and it uses GP-based regression and kernelized independence tests. The computational cost is $O(N_g N^3)$ where N_g is the number of gradient iterations used to maximize the marginal likelihood of the GP. This is the same order of complexity as our non-linear DAG sampler.

Figure 5 shows average running times in a standard desktop machine (two cores, 2.6GHz and 4Gb RAM) over 10 different models with $N = 1000$ and $d = \{10, 20, 50, 100\}$. As expected, LiNGAM with bootstrap is very fast compared to the others while our model approaches LiNGAM with Wald statistic as the number of observations increases. We did not include LiNGAM with

second order model fit because for $d = 50$ it is already prohibitive. For this small test we used a C implementation of our model with $N_s = 19000$. We are aware that the performance of a C and a Matlab implementation can be different, however we still do the comparison because the most expensive operations in the Matlab code for LiNGAM are computed through BLAS routines not involving large loops, thus a C implementation of LiNGAM should not be noticeably faster than its Matlab counterpart.

6. Simulation Results

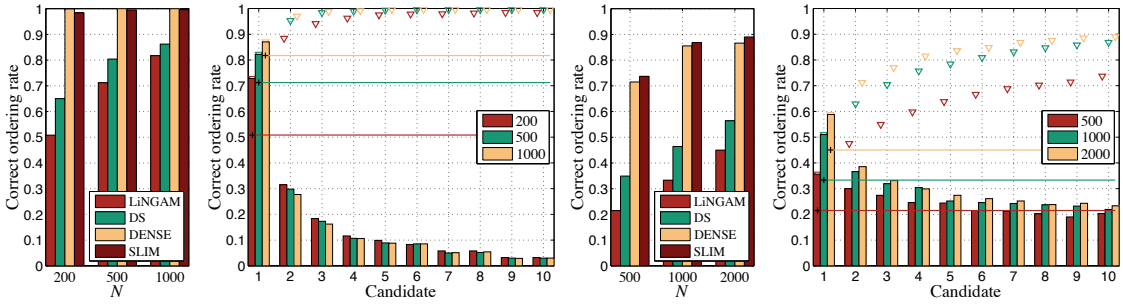
We consider six sets of experiments to illustrate the features of SLIM. In our comparison with other methods we focus on the DAG structure learning part because it is somewhat easier to benchmark a DAG than a factor model. However, we should stress that DAG learning is just one component of SLIM. Both types of model and their comparison are important, as will be illustrated through the experiments. For the reanalysis of flow cytometry data using our models, quantitative model comparison favors the DAG with latent variables rather than the standard factor model or the pure DAG which was the paradigm used in the structure learning approach of Sachs et al. (2005).

The first two experiments consist of extensive tests using artificial data in a setup originally from LiNGAM and network structures taken from the Bayesian net repository. We test the features of SLIM and compare with LiNGAM and some other methods in settings where they have proved to work well. The third set of experiments addresses model comparison, the fourth and fifth present results for our DAG with latent variables and the non-linear DAG (SNIM) on both artificial and real data. The sixth uses real data previously published by Sachs et al. (2005) and the last one provides simple results for a factor model using Gaussian process priors for temporal smoothness (CSLIM), tested on a time series gene expression data set (Kao et al., 2004). In all cases we ran 10000 samples after a burn-in period of 5000 for the factor model, and a single chain with 3000 samples and 1000 as burn-in iterations for the DAG, that is, $N_s = 19000$ used in the computational cost comparison. As a summary statistic we use median values everywhere, and Laplace distributions for the latent factors if not stated otherwise.

6.1 Artificial Data

We evaluate the performance of our model against LiNGAM,[3] using the artificial model generator presented and fully explained in Shimizu et al. (2006). Concisely, the generator produces both dense and sparse networks with different degrees of sparsity, $\mathbf{Z}$ is generated from a heavy-tailed non-Gaussian distribution through a generalized Gaussian distribution with zero mean, unit variance and random shape, $\mathbf{X}$ is generated recursively using Equation (1) with $m = 0$ and then randomly permuted to hide the correct order, $\mathbf{P}$. Approximately, half of the networks are fully connected while the remaining portion comprises sparsity levels between 10% and 80%. Having dense networks (0% sparsity) in the benchmark is crucial because in such cases the correct order of the variables is unique, thus more difficult to find. This setup is particularly challenging because the model needs to identify both dense and sparse models. For the experiment we have generated 1000 different data set/models using $d = \{5, 10\}, N = \{200, 500, 1000, 2000\}$ and the DAG was selected using the median of the training likelihood, $p(\mathbf{X}|\mathbf{P}_{\mathrm{r}}^{(k)}, \mathbf{R}^{(k)}, \mathbf{B}^{(k)}, \mathbf{C}_D^{(k)}, \mathbf{Z}, \mathbf{\Psi}, \cdot)$, for $k = 1, \ldots, m_{\mathrm{top}}$.

3. Matlab package (v.1.42) available at `http://www.cs.helsinki.fi/group/neuroinf/lingam/`.

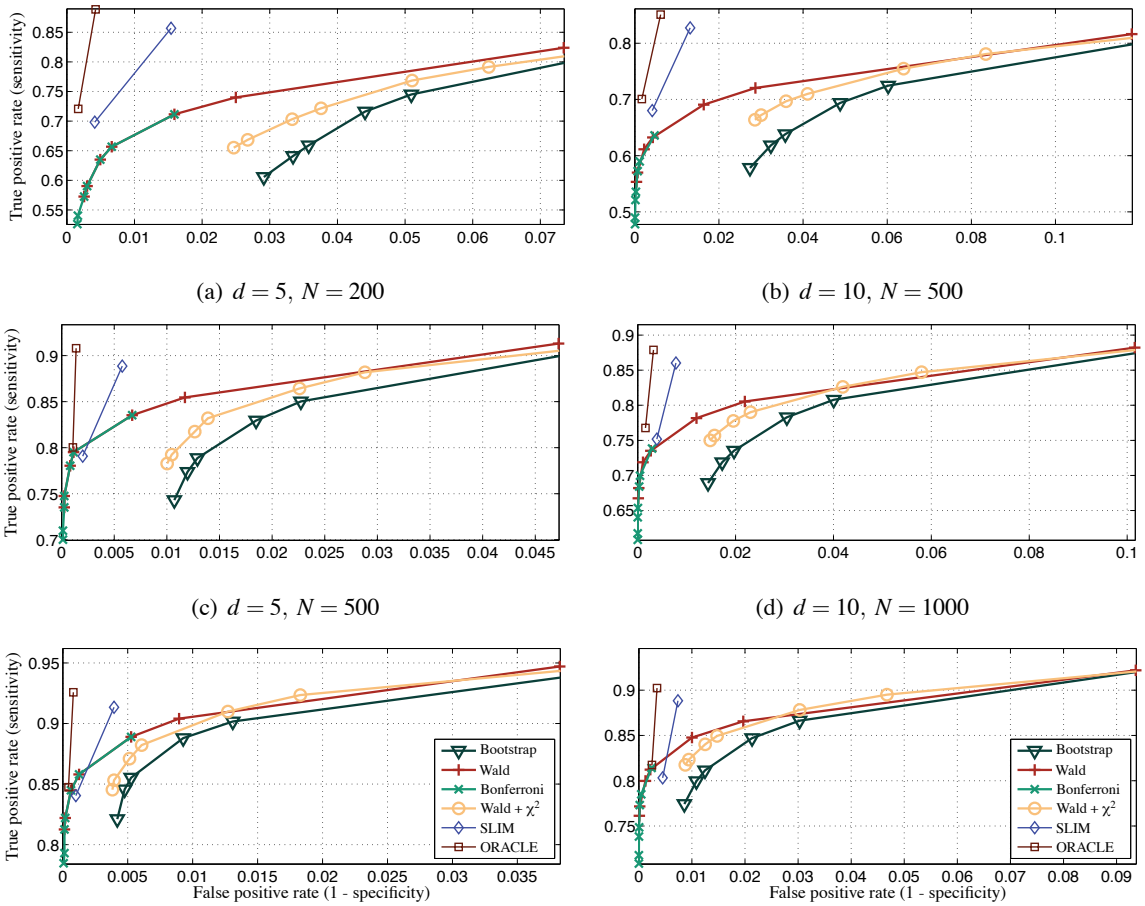


(a) (b) (c) (d)

Figure 6: Ordering accuracies for LiNGAM suite using $d = 5$ in (a,b) and $d = 10$ in (c,d). (a,c) Total correct ordering rates where DENSE is our factor model without sparsity prior and DS corresponds to DENSE but using the deterministic ordering search used in LiNGAM. (b,c) Correct ordering rate vs. candidates from SLIM. The crosses and horizontal lines correspond to LiNGAM while the triangles are accumulated correct orderings across candidates used by SLIM.

6.1.1 ORDER SEARCH

With this experiment we want to quantify the impact of using sparsity, stochastic ordering search and more than one ordering candidate, that is, $m_{\text{top}} = 10$ in total. Figure 6 evaluates the proportion of correct orderings for different settings. We have the following abbreviations for this experiment, DENSE is our factor model without sparsity prior, that is, assuming that $p(r_{ij} = 1) = 1$ a priori. DS (deterministic search) assumes no sparsity as in DENSE but replaces our stochastic search for permutations with the deterministic approach used by LiNGAM, that is, we replace the M-H update from Equation (7) by the procedure described next: after inference we compute $\mathbf{D}^{-1}$ followed by a column permutation search using the Hungarian algorithm and a row permutation search by iterative pruning until getting a version of $\mathbf{D}$ as triangular as possible (Shimizu et al., 2006). Several comments can be made from the results, (i) For $d = 5$ there is no significant gain for increasing N, mainly because the size of the permutation space is small, that is, 5!. (ii) The difference in performance between SLIM and DENSE is not significative because we look for triangular matrices in a probabilistic sense, hence there is no real need for exact zeros but just very small values, this does not mean that the sparsity in the factor model is unnecessary, on the contrary we still need it if we want to have readily interpretable mixing matrices. (iii) Using more than one ordering candidate considerably improves the total correct ordering rate, for example, by almost 30% for $d = 5$, $N = 200$ and 35% for $d = 10$, $N = 500$. (iv) The number of accumulated correct orderings found saturates as the number of candidates used increases, suggesting that further increasing m_{top} will not considerably change the overall results. (v) The number of correct orderings tends to accumulate on the first candidate when N increases since the uncertainty of the estimation of the parameters in the factor model decreases accordingly. (vi) When the network is not dense, it could happen that more than one candidate has a correct ordering, hence the total rates (triangles) are not just the sum of the bar heights in Figures 6(b) and 6(d). (vii) It seems that except for $d = 10$, $N = 5000$ it is enough to consider just the first candidate in SLIM to obtain as many correct orderings as LiNGAM does. (viii) From Figures 6(a) and 6(c), the three variants of SLIM considered perform better than LiNGAM,



(a) $d = 5$, $N = 200$

(b) $d = 10$, $N = 500$

(c) $d = 5$, $N = 500$

(d) $d = 10$, $N = 1000$

(e) $d = 5$, $N = 1000$

(f) $d = 10$, $N = 2000$

Figure 7: Performance measures for LiNGAM suite. Results include the settings: $d = \{5, 10\}, N = \{200, 500, 1000, 2000\}$, four model selectors for LiNGAM (bootstrap, Wald, Bonferroni and Wald + χ^2 statistics) and seven p-value cutoffs for the statistics used in LiNGAM (0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5). ORACLE corresponds to oracle results for SLIM, both computed for two settings: diffuse $\beta_m = 0.99$ and sparse $\beta_m = 0.1$ priors. Markers close to the top-left corner denote better results in average.

even when using the same single candidate ordering search proposed by Shimizu et al. (2006). (ix) In some cases the difference between SLIM and LiNGAM is very large, for example, for $d = 10$ using two candidates and $N = 1000$ is enough to obtain as many correct orderings as LiNGAM with $N = 5000$.

6.1.2 DAG LEARNING

Now we evaluate the ability of our model to capture the DAG structure in the data, provided the permutation matrices obtained in the previous stage as a result of our stochastic order search. Results are summarized in Figure 7 using receiving operating characteristic (ROC) curves. The true and

false positive rates are averaged over the number of trials (1000) for each setting to make the scaling in the plots more meaningful given the various levels of sparsity considered. The rates are computed in the usual way, however it must be noted that the true number of absent links in a network can be as large as $d(d-1)$, that is, twice the number of links in a DAG, because in the case of an estimated DAG based in a wrong ordering the number of false positives can sum up to $d(d-1)/2$ even if the true network is not empty. For LiNGAM we use four different statistics to prune the DAG after the ordering has been found, namely bootstrapping, Wald, Bonferroni and Wald with second order χ^2 model fit test. In every case we run LiNGAM for 7 different p-value cutoffs, namely, $0.0005, 0.001, 0.005, 0.01, 0.05, 0.1$ and 0.5 to build the ROC curve. For SLIM we consider the two settings for β_m discussed in Section 3.1, that is, a diffuse prior supporting the existence of dense graphs, $\beta_m = 0.99$ and $\beta_m = 0.1$. In order to test how good SLIM is at selecting one DAG out of the m_{top} candidates, we also report the oracle results under the name of ORACLE, where in every case we select the candidate with less error instead of $\text{argmax}_k \; p(\mathbf{X}|\mathbf{P}_{\text{r}}^{(k)}, \mathbf{R}^{(k)}, \mathbf{B}^{(k)}, \mathbf{C}_D^{(k)}, \mathbf{Z}, \mathbf{\Psi}, \cdot)$. Using $\beta_m = 0.99$ is not very useful in practice because in a real situation we expect that the underlying DAG is sparse, however the LiNGAM suite has as many dense graphs as sparse ones making $\beta_m = 0.1$ a poor choice. From Figure 7, it is clear that for $\beta_m = 0.99$, SLIM is clearly superior, providing the best true positive rate (TPR) - false positive rate (FPR) tradeoff. For $\beta_m = 0.1$ there is no real difference between SLIM and some settings of LiNGAM (Wald and Bonferroni). Concerning SLIM's model selection procedure, it can be seen that the difference between SLIM and ORACLE nicely decreases as the number of observations increases. We also tested the DAG learning procedure in SLIM when the true ordering is known (results not shown) and we found only a very small difference compared to ORACLE. It is important to mention that further increasing or reducing β_m does not significantly change the results shown; this is because β_m does not fully control the sparsity of the model, thus even for $\beta_m = 1$ the model will be still sparse due to element-wise link confidence, α_m. As for LiNGAM, it seems that Wald performs better than Wald + χ^2, however just by looking at Figure 7, it is to be expected that for larger N the latter perform better because the Wald statistic alone will tend to select more dense models.

6.1.3 ILLUSTRATIVE EXAMPLE

Finally we want to show some of the most important elements of SLIM taking one successfully estimated example from the LiNGAM suite. Figure 8 shows results for a particular DAG with 10 variables obtained using 500 observations, see Figures 8(d) and 8(e) for the ground truth and the estimated DAG, respectively. True and estimated mixing matrices $\mathbf{D}$ for the equivalent factor model are also shown in Figures 8(a) and 8(b), respectively. In total our algorithm produced 92 orderings out of 3.6×10^6 possible, from which all $m_{\text{top}} = 10$ candidates were correct. Figure 8(c) shows the first 50 candidates and their frequency during sampling, the shaded area encloses the $m_{\text{top}} = 10$ candidates. From Figure 8(f) we see that the elements of $\mathbf{B}$ are correctly estimated and their credible intervals are small, mainly due to the lack of model mismatch. Figure 8(g) shows a good separation between zero and non-zero elements of $\mathbf{B}$ as summarized by $p(r_{ij} = 1|\mathbf{X}, \cdot)$. It is worthwhile mentioning that using $\beta_m = 0.99$ instead of $\beta_m = 0.1$ in this example, still produces the right DAG, although the separation between zero and non-zero elements in Figure 8(g) will be smaller and with higher uncertainty, that is, larger credible intervals.

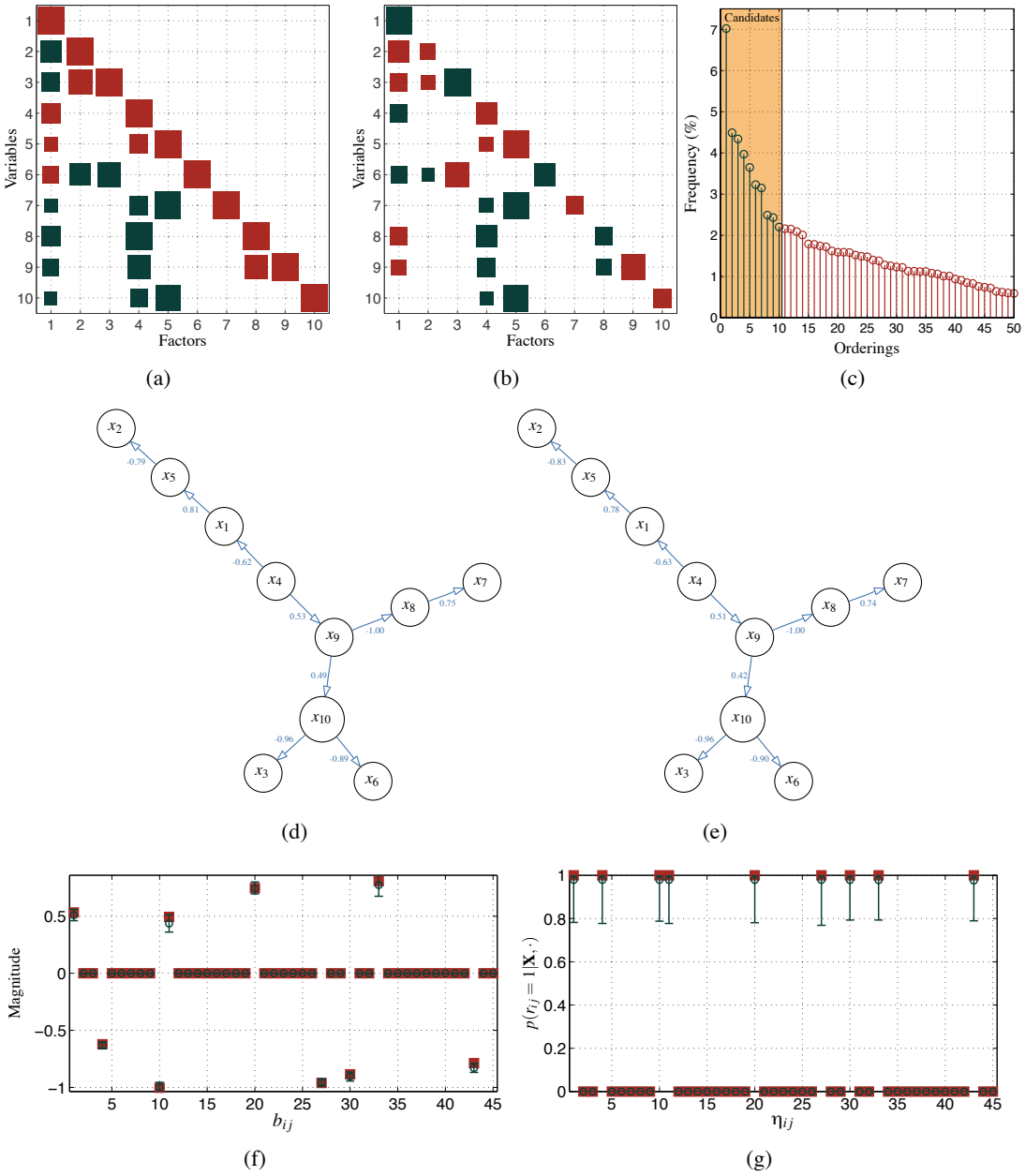
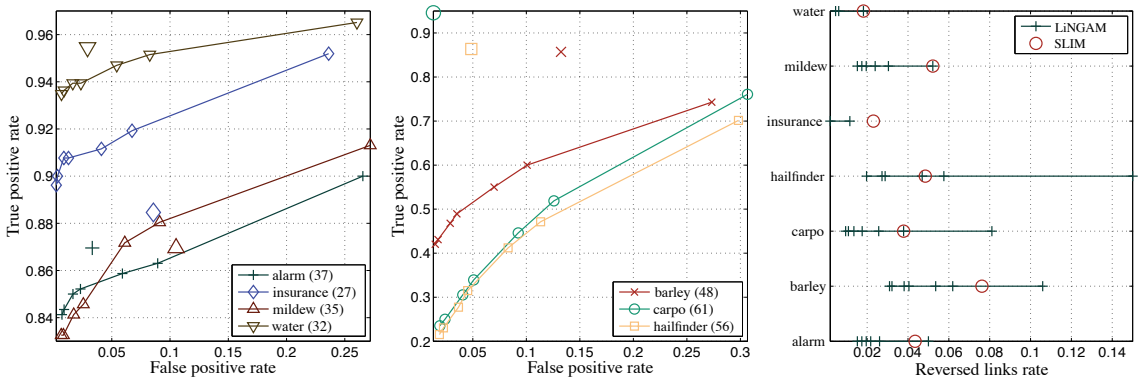


Figure 8: Ground truth and estimated structures. (a) Ground truth mixing matrix. (b) Estimated mixing matrix using our sparse factor model. Note the sign ambiguity in some of the columns. (c) First 50 (out of 92) ordering candidates produced by our method during inference and their frequency, the first m_{top} candidates were used for to learn DAGs. (d) Ground truth DAG. (e) Top candidate estimated using SLIM. (f) Estimated median weights for the DAG including 95% credible intervals and ground truth (squares). (g) Summary of link probabilities measured as $\eta_{ij} = p(r_{ij} = 1|\mathbf{X}, \cdot)$.



(a) (b) (c)

Figure 9: Performance measures for the Bayesian networks repository experiments. Each connected marker correspond to a different p-value in LiNGAM, starting left to right from 0.005. Disconnected markers denote SLIM results. Numbers in parentheses indicate number of variables.

6.2 Bayesian Networks Repository

Next we want to compare our method against LiNGAM on some realistic structures. We consider 7 well known benchmark structures from the Bayesian network repository,[4] namely alarm, barley, carpo, hailfinder, insurance, mildew and water ($d = 37, 48, 61, 56, 27, 35, 32$ respectively). Since we do not have continuous data for any of the structures, we generated 10 data sets of size $N = 500$ for each of them using heavy-tailed distributions with different parameters and Equation (1) with $m = 0$, in a similar way as we did for the previous set of experiments, with $\mathbf{R}$ set to the ground truth and $\mathbf{B}$ from $\text{sign}(\mathcal{N}(0,1)) + \mathcal{N}(0,0.2)$. For LiNGAM, we only use Wald statistics because as seen in the previous experiment, it performs significantly better that bootstrapping. Again, we estimate models for different p-value cutoffs ($0.0005, 0.001, 0.005, 0.01, 0.05, 0.1$ and 0.5). For SLIM, we set $\beta_m = 0.1$ since all the networks in the repository are sparse. Figures 9(a), 9(b) and 9(c) show averaged performance measures respectively as ROC curves and the proportion of links reversed in the estimated model due to ordering errors.

In this case, the results are mixed when looking at the performances obtained. Figure 9(b) shows that SLIM is better than LiNGAM in the larger data sets with a significant difference. Figure 9(a) shows for the remaining four data sets, that LiNGAM is better in two cases corresponding to the insurance and mildew networks. In general, both methods perform reasonably well given the size of the problems and the amount of data used to fit the models. However, SLIM tends to be more stable, when looking at the range of the true positive rates. It is important to note that the best and worst case for SLIM correspond to the largest and smallest network, respectively. We do not have a sensible explanation about why SLIM is performing that poorly on the insurance network. Figure 9(c) implicitly reveals that both methods are unable to find the right ordering of the variables.

We also tried the following methods with encoded Gaussian assumptions: standard DAG search, order search, sparse candidate pruning then DAG search (Friedman et al., 1999), L1MB then DAG

4. Network structures available at `http://compbio.cs.huji.ac.il/Repository/`.

search (Schmidt et al., 2007), and sparse candidate pruning then order search (Teyssier and Koller, 2005). We observed (results not shown) that these methods produce similar results to those obtained by either LiNGAM or SLIM when only looking at the resulting undirected graph, that is, removing the directionality of the links. Evaluation of directionality in Gaussian models is out of the question because such methods can only find DAGs up to Markov equivalence classes, thus evaluation must be made using partially directed acyclic graphs (PDAGs). It is still possible to modify some of the methods mentioned above to handle non-Gaussian data by for instance using some other appropriate conditional independence tests, however this is out of the scope of this paper.

6.3 Model Comparison

In this experiment we want to evaluate the model selection procedure described in Section 4. For this purpose we have generated 1000 different data sets/models with $d = 5$ and $N = \{500, 1000\}$ following the same procedure described in the first experiment, but this time we selected the true model to be either a factor model or a DAG with equal probability. In order to generate a factor model, we basically just need to ensure that $\mathbf{D}$ cannot be permuted to a triangular form, so the data generated from it does not admit a DAG representation. We kept 20% of the data to compute the predictive densities to then select between all estimated DAG candidates and the factor model. We found that for $N = 500$ our approach was able to select true DAGs 96.78% of the times and true factor models 87.05%, corresponding to an overall accuracy of 91.9%. Increasing the number of observations, that is, for $N = 1000$, the true DAG, true factor model rates and overall error increased to 98.99%, 95.0% and 96.99%, respectively. Figure 10 shows separately the empirical log-likelihood ratio distributions obtained from the 1000 data sets for DAGs and factor models. The shaded areas correspond to the true DAG/factor model regions, with zero as their boundary. Note that when the wrong model is selected the likelihood ratio is nicely close to the boundary and the overlap of the two distributions decreases with the number of observations used, since the quality of the predictive density increases accordingly. The true DAG rates tend to be larger than for factor models because it is more likely that the latter is confused with a DAG due to estimation errors or closeness to a DAG representation, than a DAG being confused with a factor model which is naturally more general. This is precisely why the likelihood ratios tend to be larger on the factor model side of he plots. All in all, these results demonstrate that our approach is very effective at selecting the true underlying structure when the data is generated by one of the two hypotheses.

6.4 DAGs with Latent Variables

We will start by illustrating the identifiability issues of the model in Equation (1) discussed in Section 2.1 with a very simple example. We generated $N = 500$ observations from the graph in Figure 3(b) and kept 20% of the data to compute test likelihoods. Now, we perform inference on two slightly different models, namely, (u) where $\mathbf{z}' = [z_1' \; z_2' \; z_L']$ is provided with Laplace distributions with unit variance, that is, $\lambda = 2$, and (i) where z_1, z_2 have Laplace distributions with unit variance and z_L is Cauchy distributed. We want to show that even if both models match the true generating process, (u) is non-identifiable whereas (i) can be successfully estimated. In order to keep the experiment controlled as much as possible, we set $\beta_m = 0.99$ to reflect that the ground truth is dense and we did not infer $\mathbf{C}_D$ and set it to the true values, that is, the identity. Then, we ran 10 independent chains for each one of the models and summarized $\mathbf{B}$, $\mathbf{C}_L$, $\mathbf{D}$ and the test likelihoods in Figure 11.

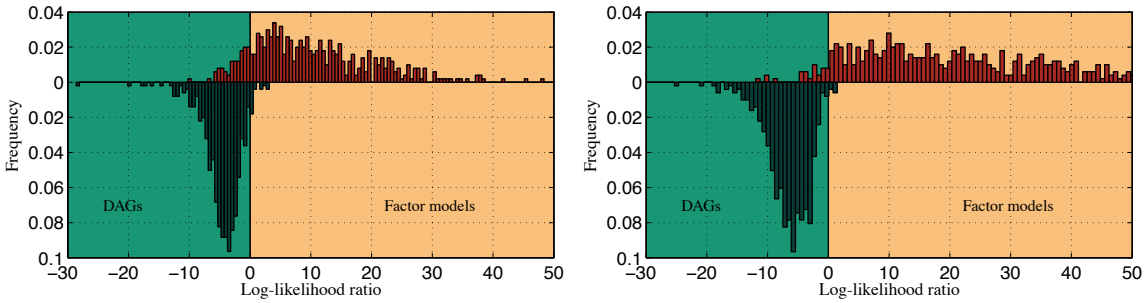


Figure 10: Log-likelihood ratio empirical distributions for, (a) $N = 500$ and (b) $N = 1000$. Top bars correspond to true factor models, bottom bars to true DAGs and the ratio is computed as described in Section 4. Top bars lying below zero are true factor models predicted to be better explained by DAGs, thus model comparison errors.

Figure 11(a) shows that model (u) finds the DAG in Figure 3(b) (the ground truth) in 3 cases, and in the remaining 7 cases it finds the DAG in Figure 3(a). Note also that the test likelihoods in Figure 11(c) are almost identical, as must be expected due to the lack of identifiability of the model, so they cannot be used to select among the two alternatives. Model (i) finds the right structure all the times as shown in Figure 11(d). The mixing matrix of the equivalent factor model, $\mathbf{D}$ is shown in Figures 11(b) and 11(e) for (u) and (i), respectively. In Figure 11(b), the first and third column of $\mathbf{D}$ exchange positions because all the components of $\mathbf{z}$ have the same distribution, which is not the case of Figure 11(e). The small quantities in $\mathbf{D}$ are due to estimation errors when computing $b_{21}c_{1L} + c_{2L}$, and this cancels out in the true model. The sign changes in Figures 11(a) and 11(d) are caused by the sign ambiguity of $\mathbf{z}_L$ in the product $\mathbf{C}_L\mathbf{z}_L$. We also tested the alternative model in Figure 3(b) obtaining equivalent results, that is, 4 successes for model (u) and 10 for model (i). This small example shows how non-identifiability may lead to two very different DAG solutions with distinct interpretations of the data.

Hoyer et al. (2008) recently presented an approach to DAGs with latent variables based on LiNGAM (Shimizu et al., 2006). Their procedure uses probabilistic ICA and bootstrapping to infer the equivalent factor model distribution $p(\mathbf{D}|\mathbf{X})$, then greedily selects m columns of $\mathbf{D}$ to be latent variables until the remaining ones can be permuted to triangular and the resulting DAG is compatible with the faithfulness assumption (see, Pearl, 2000). If we assume that their procedure is able to find the exact $\mathbf{D}$ for the graphs in Figures 3(a) and 3(b), due to the faithfulness assumption, the DAG in Figure 3(a) will be always selected regardless of the ground truth.[5] In practice, the solution obtained for $\mathbf{D}$ is dense and needs to be pruned, hence we rely on $p(\mathbf{X},\mathbf{D})$ being larger for the ground truth in Figure 3(b) than for the graph in Figure 3(a), however since both models differ only by a permutation of the columns of $\mathbf{D}$, they have exactly the same joint density $p(\mathbf{X},\mathbf{D})$—they are non-identifiable, thus the algorithm will select one of the options by chance. Since the source of non-identifiability of their algorithm is permutations of columns of $\mathbf{D}$, it does not matter if probabilistic ICA match or not the distribution of the underlying process as in our model. Anyway, we decided to try models (u)

5. See Robins et al. (2003) for a very interesting explanation of faithfulness using the same example presented here.

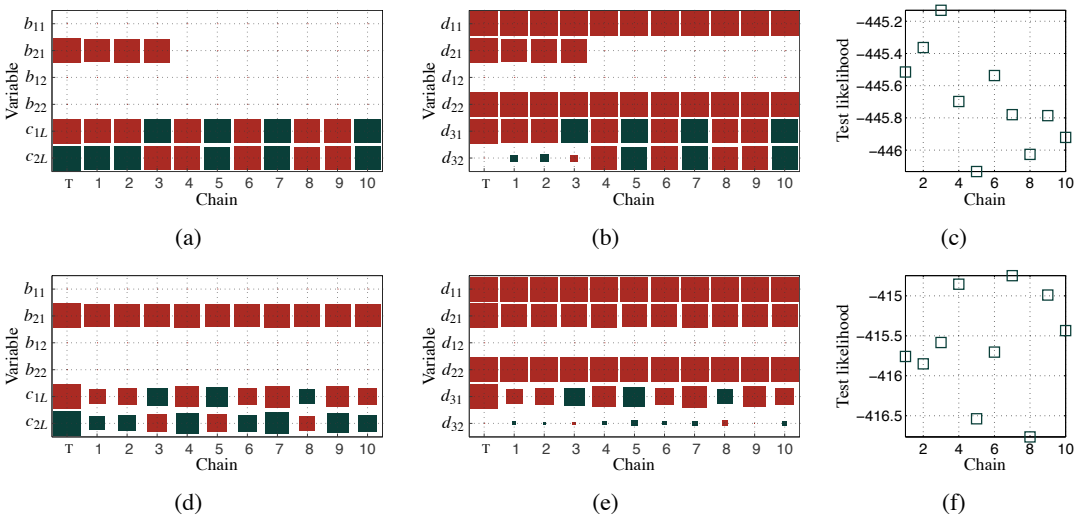
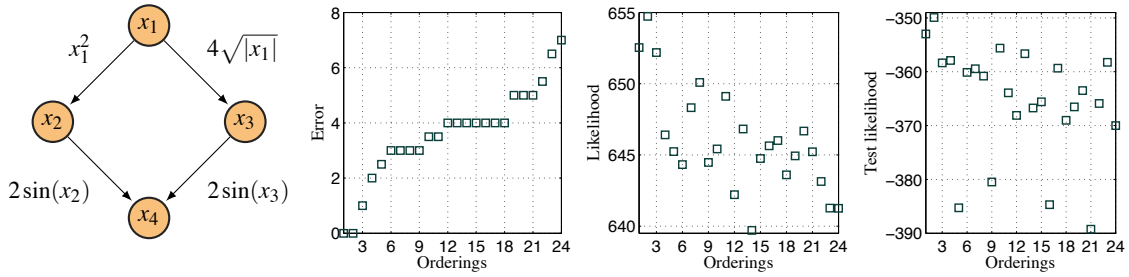


Figure 11: Identifiability experiment for the DAG with latent variables. Connectivities $\mathbf{B}$ and $\mathbf{C}_L$ are shown for (u) in (a) and (i) in (d). Equivalent mixing matrix $\mathbf{D}$ for (u) in (b) and for (i) in (d). Test likelihoods for (u) and (i) are shown in (c) and (f) respectively. The first column in (a,b,d,e) denoted as T is the ground truth. Dark and light boxes are negative and positive numbers, accordingly.

and (i) described above using the algorithm just described.[6] Regardless of the ground truth, Figures 3(a) or 3(b), the algorithm always selected the DAG in Figure 3(b), which in this particular case is due to $p(\mathbf{X},\mathbf{D})$ being slightly larger for the denser model.

Now we test the model in a more general setting. We generate 100 models and data sets of size $N = 500$ using a similar procedure to the one in the artificial data experiment. The models have $d = 5$ and $m = 1$, no dense structures are generated and the distributions for $\mathbf{z}$ are heavy-tailed, drawn from a generalized Gaussian distribution with random shape. For SLIM, we use the following settings, $\beta_m = 0.1$, $\mathbf{z}_D$ is Laplace with unit variances and $\mathbf{z}_L$ is Cauchy. Furthermore, we have doubled the number of iterations of the DAG sampler, that is, 6000 samples and a burn-in period of 2000, so as to compensate for the additional parameters that need to be inferred due to inclusion of latent variables. Our ordering search procedure was able to find the right ordering 78 out of 100 times. The true positive rates, true negative rates and median AUC are 88.28%, 96.40% and 0.929, respectively, corresponding to approximately 1.5 structure errors per network. Using Hoyer et al. (2008) we obtained 1 true ordering out of 100, 91.63% true positive rate, 65.18% true negative rate and 0.800 median AUC, showing again the preference of the algorithm for denser models. We regard these results as very satisfactory for both methods considering the difficulty of the task and the lack of identifiability of the model by Hoyer et al. (2008).

6. Matlab package (v.1.1) freely available at `http://www.cs.helsinki.fi/group/neuroinf/lingam/`.



(a) (b) (c) (d)

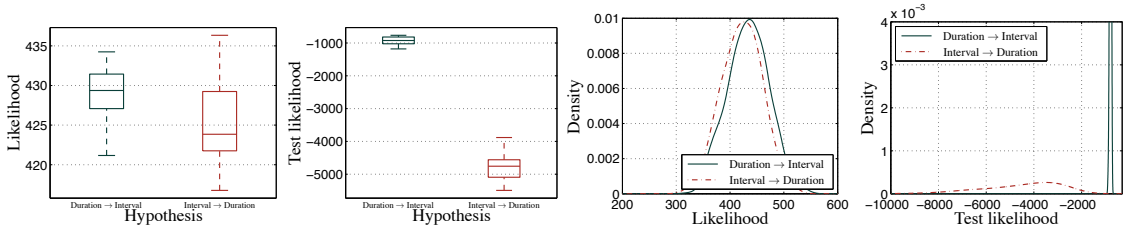
Figure 12: Non-linear DAG artificial example. (a) Network with non-linear interactions between observed nodes used as ground truth. (b,c,d) Median error, likelihood and test likelihood for all possible orderings and 10 independent repetitions. The plots are sorted according to number of errors and only the first two are valid according to the ground truth in (a), that is, $(1,2,3,4)$ and $(1,3,2,4)$. Note that when the error is zero in (b) the likelihoods are larger with respect to the remaining orderings in (c) and (d).

6.5 Non-linear DAGs

For Sparse Non-linear Identifiable Modeling (SNIM) described in Section 3.5, first we want to show that our method can find and select from DAGs with non-linear interactions. We used the artificial network from Hoyer et al. (2009) shown here in Figure 12(a) and generated 10 different data sets corresponding to $N = 100$ observations, each time using driving signals sampled from different heavy-tailed distributions. Since we do not yet have an ordering search procedure for non-linear DAGs, we perform DAG inference for all possible orderings and data sets. The results obtained are evaluated in two ways, first we check if we can find the true connectivity matrix when the ordering is correct. Second, we need to validate that the likelihood is able to select the model with less error and correct ordering among all possible candidates so we can use it in practice. Figures 12(b), 12(c) and 12(d) show the median errors, training and test likelihoods (using 20% of the data) for each one of the orderings, respectively. In this particular case we only have two correct orderings, namely, $(1,2,3,4)$ and $(1,3,2,4)$, corresponding to the first and second candidates in the plots. Figure 12(b) shows that the error is zero only for the two correct orderings, then our model is able to infer the structure once the right ordering is given as desired. As a result of the identifiability, data and test likelihoods shown in Figures 12(c) and 12(d) correlate nicely with the structural error in Figure 12(b). This means that we can use use the likelihoods as a proxy for the structural error just as in the linear case.

We also tested the network in Figure 12(a) using three non-linear structure learning procedures namely greedy standard hill-climbing DAG search, the "ideal parent" algorithm (Elidan et al., 2007) and kernel PC (Tillman et al., 2009). The first two methods use a scaled sigmoid function to capture the non-linearities in the data. In particular, they assume that a variable x can be explained as scaled sigmoid transformation of a linear combination of its parents. The best median result we could obtain after tuning the parameters of the algorithms was 2 errors and 2 reversed links.[7] Both

7. Maximum number of iterations, random restarts to avoid local minima, regularization of the non-linear regression and the number of ranking candidates in ideal parent algorithm.



(a) (b) (c) (d)

Figure 13: Testing {duration, interval} in Old Faithful data set. (a,b) Data and test likelihood boxplots for 10 independent repetitions. (c,d) Training and test likelihood densities for one of the repetitions. The test likelihood separates consistently the two tested hypotheses.

methods perform similarly in this particular example, the only significant difference being their computational cost, which is considerably smaller for the "ideal parent" algorithm, as it was also pointed out by Elidan et al. (2007). The reason why we consider these algorithms do not perform well here is that the sigmoid function can be very limited at capturing certain non-linearities due to its parametric form whereas the nonparametric GP gives flexible non-linear functions. The third method uses non-linear independence tests together with non-linear regression (relevance vector machines) and the PC algorithm to produce mixed DAGs. The best median result we could get in this case was 2 errors, 0 reversed links and 1 bidirectional links. These three non-linear DAG search algorithms have the great advantage of not requiring exhaustive enumeration of the orderings as our method and others available in the literature. Zhang and Hyvärinen (2009) provides theoretical evidence of the possibility for flexible non-linear modeling without exhaustive order search but not a way to do it in practice. Yet another possibility not tried here will be to take the best parts of both strategies by taking the outcome of the non-linear DAG search algorithm and refine it using a nonparametric method like SNIM. However, it is not entirely clear how the non-linearities can affect the ordering of the variables. In the remaining part of this section we only focus on tasks for pairs of variables where the ordering search is not an issue.

The data set known as Old Faithful (Asuncion and Newman, 2007) contains 272 observations of two variables measuring waiting time between eruptions and duration of eruptions for the Old Faithful geyser in Yellowstone National Park, USA. We want to test the two possible orderings, duration $\rightarrow$ interval and interval $\rightarrow$ duration. Figures 13(a) and 13(b) show training and test likelihood boxplots for 10 independent randomizations of the data set with 20% of the observations used to compute test likelihoods. Our model was able to find the right ordering, that is, duration $\rightarrow$ interval in all cases when the test likelihood was used but only 7 times with the training likelihood due to the proximity of the densities, see Figure 13(c). On the other hand, the predictive density is very discriminative, as shown for instance in Figure 13(d). This is not a very surprising result since making the duration a function of the interval results in a very non-linear function, whereas the alternative function is almost linear (data not shown).

Abalone is one of the data sets from the UCI ML repository (Azzalini and Bowman, 1990). It is targeted to predict the age of abalones from a set of physical measurements. The data set contains 9 variables and 4177 observations. First we want to test the pair {age, length}. For this purpose, we use 10 subsets of $N = 200$ observations to build the models and compute likelihoods just as

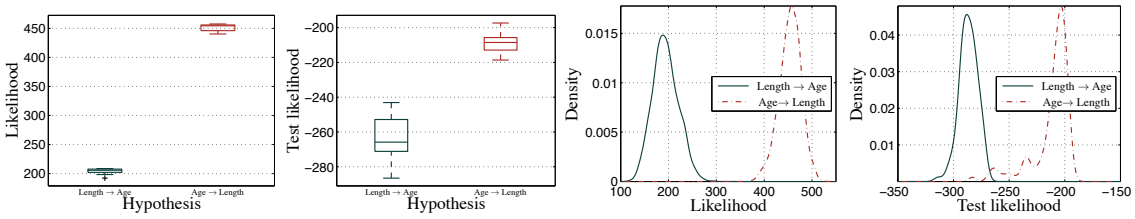


Figure 14: Testing {length, age} in Abalone data set. (a,b) Data and test likelihood boxplots for 10 independent repetitions. (c,d) Training and test likelihood densities for one of the repetitions. The likelihoods largely separate the two tested hypotheses.

before. Figures 14(a) and 14(b) show training and test likelihoods respectively as boxplots. Both training and test likelihoods pointed to the right ordering in all 10 repetitions. In this experiment, the separation of the densities for the two hypotheses considered is very large, making age $\rightarrow$ length significantly better supported by the data. Figures 14(c) and 14(d) show predictive densities for one of the trials indicating again that age $\rightarrow$ length is consistently preferred. We also decided to try another three sets of hypotheses: {age, diameter}, {age, weight} and {age, length, weight} for which we found the right orderings $\{10, 10\}$, $\{10, 10\}$ and $\{10, 6\}$ out of 10 by looking at the training and the test likelihoods, respectively. In the model with three variables, increasing the number of observations used to fit the model from $N = 200$ to $N = 400$, increased the number of cases in which the test likelihood selected the true hypothesis from 6 to 8 times, which is more than enough to make a decision about the leading hypothesis.

To conclude this set of experiments we test SNIM against another three recently proposed methods,[8] namely Non-linear Additive Noise (NAN) model (Hoyer et al., 2009), Post-Non-Linear (PNL) model (Zhang and Hyvärinen, 2009) and Informational Geometric Causal Inference (IGCI) (Daniusis et al., 2010), using an extended version of "cause-effect pairs" task for the NIPS 2008 causality competition[9] (Mooij and Janzing, 2010). The task consists on distinguishing the cause from the effect of 51 different pairs of observed variables. NAN and PNL rely on an independence test (HSIC, Hilbert-Schmidt Independence Criterion, Gretton et al., 2008) to decide which of the two variable is the cause. NAN was able to take 10 decisions all being accurate. PNL was accurate 40 times out of 42 decisions made. IGCI and SNIM obtained an accuracy of 40 and 39 pairs, respectively.[10] The results indicate (i) that NAN and PNL are very accurate when the independence test used is able to reach a decision and (ii) in terms of accuracy, the results obtained by PNL, IGCI and SNIM are comparable. For SNIM we decide based upon the test likelihood and for IGCI we used a uniform reference measure (rescaling the data between 0 and 1). From the four tested methods we can identify two main trends. One is to explicitly model the data and decide the cause-effect direction using independence tests or test likelihoods like in NAN, PNL and SNIM. The second is to directly define a measure for directionality as in IGCI. The first option has the advantage of being able to convey

8. Matlab packages available at `http://webdav.tuebingen.mpg.de/causality/`.
9. Data available at `http://webdav.tuebingen.mpg.de/cause-effect/`.
10. Results for NAN, PNL and IGCI were taken from Daniusis et al. (2010) because we were unable to entirely reproduce their results with the software provided by the authors.

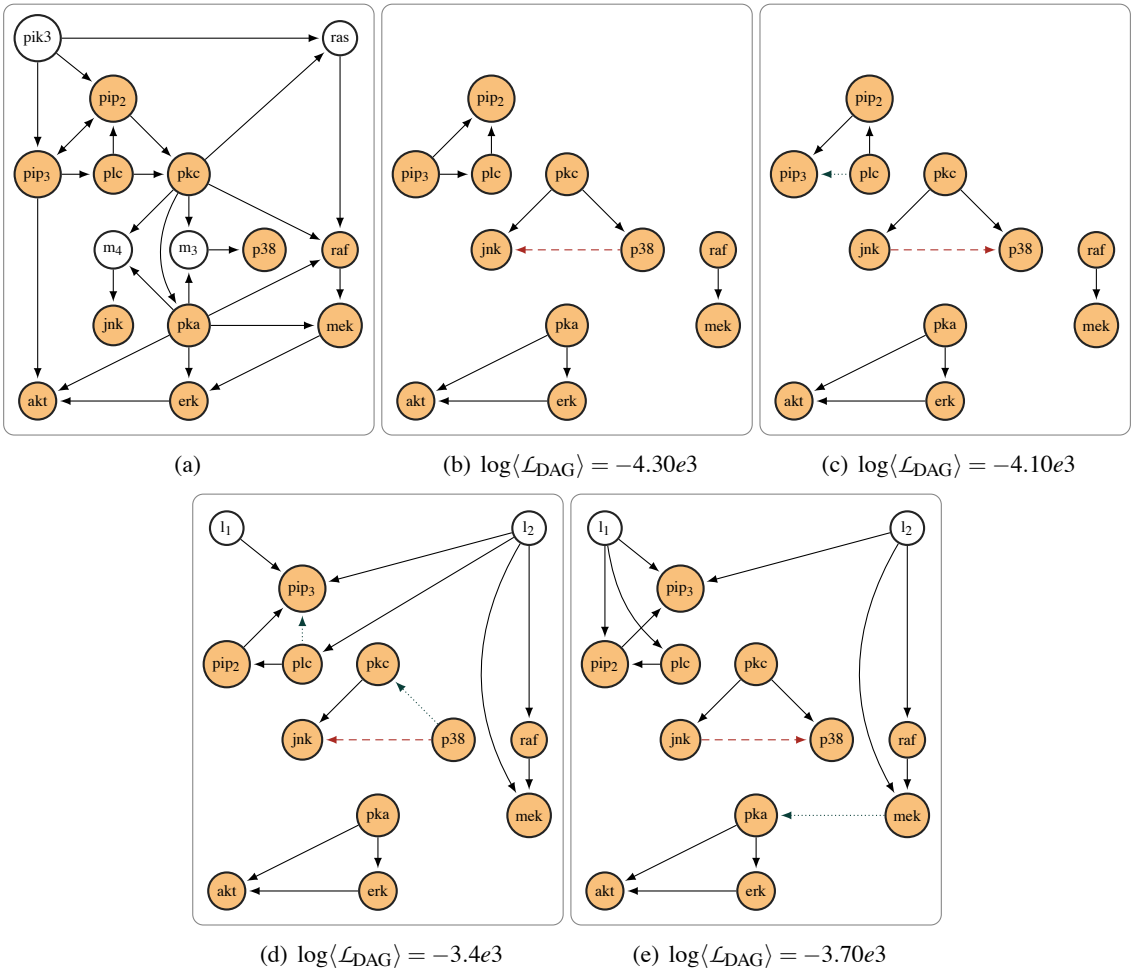
more information about the data at hand whereas the second option is orders of magnitude faster than the other three because it only tests for directionality.

6.6 Protein-signaling Network

This experiment demonstrates a typical application of SLIM in a realistic biological large N, small d setting. The data set introduced by Sachs et al. (2005) consists of flow cytometry measurements of 11 phosphorylated proteins and phospholipids (raf, erk, p38, jnk, akt, mek, pka, pkc, pip_2, pip_3, plc). Each observation is a vector of quantitative amounts measured from single cells. Data was generated from a series of stimulatory cues and inhibitory interventions. Hence the data is composed of three kinds of perturbations: general activators, specific activators and specific inhibitors. Here we are only using the 1755 observations—clearly non-Gaussian, for example, see Figure 16(a), corresponding to general stimulatory conditions. It is clear that using the whole data set, that is, using specific perturbations, will produce a richer model, however handling interventional data is out of the scope of this paper mainly because handling that kind of data with a factor model is not an easy task. Thus our current order search procedure is not appropriate. Focused only on the observational data, we want to test all the possibilities of our model in this data set, namely, standard factor models, pure DAGs, DAGs with latent variables, non-linear DAGs and quantitative model comparison using test likelihoods. The textbook DAG structure taken from Sachs et al. (see Figure 2 and Table 3, 2005) is shown in Figure 15(a) and the models are estimated using the true ordering and SLIM in Figures 15(b) and 15(c), respectively.

The DAG found using the right ordering of the variables shown in Figure 15(b) turned out to be the same structure found by the discrete Bayesian network from Sachs et al. (2005) without using interventional data (see supplementary material, Figure 4(a)), with one important difference: the method presented by Sachs et al. (2005) is not able to infer the directionality of the links in the graph without interventional data, that is, their resulting graph is undirected. SLIM in Figure 15(c) finds a network almost equal to the one in Figure 15(b) apart from one reversed link, plc $\rightarrow$ pip3. Surprisingly this was also found reversed by Sachs et al. (2005) using interventional data. In addition, there is just one false positive, the pair $\{$jnk, p38$\}$, even with a dedicated latent variable in the factor model mixing matrix shown in Figure 16(b), thus we cannot attribute such a false positive to estimation errors. A total of 211 ordering candidates were produced during the inference out of approximately 10^7 possible and only $m_{\text{top}} = 10$ of them were used in the structure search step. Note from Figure 16(d) that the predictive densities for the DAGs correlate well with the structural accuracy, apart from candidate 8. Candidates 3 and 8 have the same number of structural errors, however candidate 8 has 3 reversed links instead of 1 as shown in Figure 15(c). The predictive densities for the best candidate, third in Figure 16(d) are shown in Figure 16(c) and suggest that the factor model fits the data better. This makes sense considering that estimated DAG in Figure 15(c) is a substructure of the ground truth. We also examined the estimated factor model in Figure 16(b) and we found that several factors could correspond respectively to three unmeasured proteins, namely pi3k in factors 9 and 11, m_3 (mapkkk, mek4/7) and m_4 (mapkkk, mek3/6) in factor 7, ras in factors 4 and 6.

We also wanted to assess the performance of our method and several others using this data set, including LiNGAM and those mentioned in the Bayesian network repository experiment, even knowing that this data set contains non-Gaussian data. We found that all of them have similar results in terms of true and false positive rates when comparing them to SLIM. However the number



(a) (b) $\log\langle \mathcal{L}_{\text{DAG}} \rangle = -4.30e3$ (c) $\log\langle \mathcal{L}_{\text{DAG}} \rangle = -4.10e3$

(d) $\log\langle \mathcal{L}_{\text{DAG}} \rangle = -3.4e3$ (e) $\log\langle \mathcal{L}_{\text{DAG}} \rangle = -3.70e3$

Figure 15: Result for protein-signaling network data. (a) Textbook signaling network as reported in Sachs et al. (2005). Estimated structure using SLIM: (b) using the true ordering, (c) obtaining the ordering from the stochastic search, (d) top DAG with 2 latent variables and (e) the runner-up (in test likelihood). False positives are shown in red dashed lines and reversed links in green dotted lines. Below each structure we also report the median test likelihood (larger is better).

of reversed links was not in any case less than 6, which corresponds to more than 50% of the true positives found in every case. This means that they are essentially able to find the skeleton in Figure 15(b). Besides, we do not have knowledge of any other method for DAG learning using only the observational data that also provides results substantially better than the ones shown in Figure 15(c). The poor performance of LiNGAM is difficult to explain but the large amount of reversed links may be due to the FastICA based deterministic ordering search procedure.

We also tried DAG models with latent variables in this data set. The results obtained by the DAG with 2 a priori assumed latent variables are shown in Figures 15(d) and 15(e), corresponding to the first and second DAG candidates in terms of test likelihoods. The first option is different to the pure

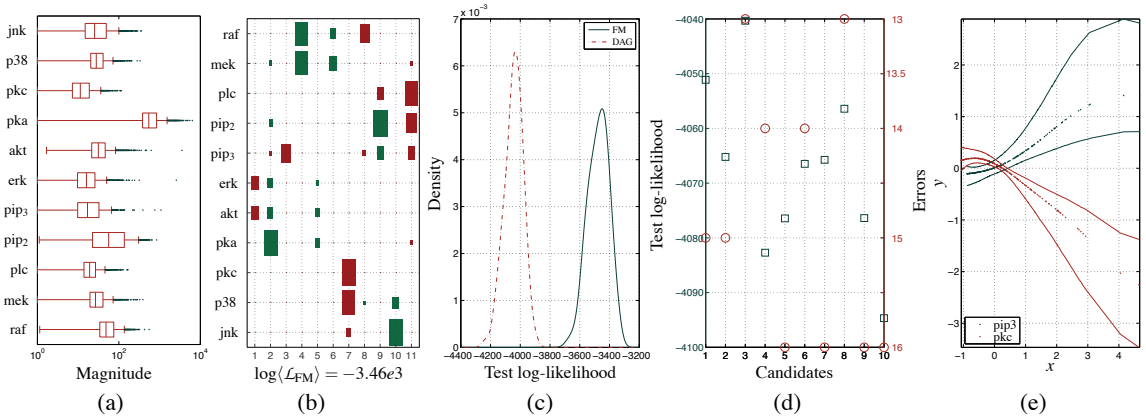


Figure 16: Results for protein-signaling network data. (a) Boxplot for each one of the 11 variables in the data set. (b) Estimated factor model. (c) Test likelihoods for the best DAG (dashed) and the factor model (solid). (d) Test likelihoods (squares) and structure errors (circles) included reversed links for all candidates. (e) Non-linear variables y obtained as a function of the observed variables x for pip3 and pkc. Each dot in the plot is an observation and the solid lines are 95% credible intervals.

DAG in Figure 15(c) only in the reversed link, p38 $\rightarrow$ pkc, but captures some of the behavior of pik3 and ras in l_1 and l_2 respectively. It is very interesting to see how, due to the link between pik3 and ras that is not possible to model with our model, the second inferred latent variable is detecting signals pointing towards pip_2 and plc. We also considered a second option because l_1 in the top model is only connected to a single variable pip_3 and thus could be regarded as an estimation error since it can be easily confounded with a driving signal. Comparing Figures 15(c) and 15(e) reveals two differences in the observed part, a false negative $pip_3 \rightarrow$ plc and a new true (reversed) positive mek $\rightarrow$ pka. This candidate is particularly interesting because the first latent variable captures the connectivity of pik3 while connecting itself to plc due to the lack of connectivity between pip_3 and plc. Moreover, the second latent variable resembles ras and the link between pik3 and ras as a link from itself to pip_3. In both solutions there is a connection between l_2 and mek that might be explained as a link through a phosphorylation of raf different to the observed one, that is, ras_{s259}. In terms of median test likelihoods, the model in Figure 15(d) is only marginally better than the factor model in Figure 16(b) and in turn marginally worse than the DAG in Figure 15(e).

For SNIM we started from the true ordering of the variables but we could not find any improvement compared to the structure in Figure 15(c). In particular there are only two differences, plc $\rightarrow$ pip_2 and jnk $\rightarrow$ p38 are missing, meaning that at least in this case there are no false positives in the non-linear DAG. Looking at the parameters of the covariance function used, $\boldsymbol{\upsilon}$ (not shown) with acceptance rates of approximately $\approx 20\%$ and reasonable credible intervals, we can say that our model found almost linear functions since all the parameters of the covariance functions are rather small. Figure 16(e) shows two particular non-linear variables learned by the model, corresponding to pip3 and plc. In each case the uncertainty of the estimation nicely increases with the magnitude of the observed variable and although the functions are fairly linear they resemble the saturation ef-

fect we can expect in this kind of biological data. From the three non-linear methods non-requiring exhaustive order search described in the previous section (DAG search, "ideal parent" and kPC), the best result we obtained was 11 structural errors, 10 true positives, 34 true negatives, 2 reversed and 6 bidirectional links for kPC vs 12, 9, 34, 1 and 0 by SLIM and 12, 8, 35, 0 and 0 by SNIM.

6.7 Time Series Data

We illustrate the use Correlated Sparse Linear Identifiable Modeling (CLSIM) on the data set introduced by Kao et al. (2004) consisting of temporal gene expression profiles of *E. coli* during transition from glucose to acetate measured using DNA microarrays. Samples from 100 genes were taken at 5, 10, 15, 30, 60 minutes and every hour until 6 hours after transition.[11] The general goal is to reconstruct the unknown transcription factor activities from the expression data and some prior knowledge. In Kao et al. (2004) the prior knowledge consisted of taking the set of transcription factors (ArcA, CRP, CysB, FadR, FruR, GatR, IcIR, LeuO, Lrp, NarL, PhoB, PurB, RpoE, RpoS, TrpR and TyrR) controlling the observed genes and the (up-to-date) connectivity between genes and transcription factors from RegulonDB[12] (Gama-Castro et al., 2008). From this setting, we can immediately relate the transcriptions factors with $\mathbf{Z}$, such a connectivity with $\mathbf{Q}_L$, and their relative strengths with $\mathbf{C}_L$, hence the problem can be seen as a standard factor model. In Kao et al. (2004) they applied a method called Network Component Analysis (NCA), that uses a least-squares based algorithm to solve a problem similar to the one in Equation (1), but assuming that the sparsity pattern (masking matrix $\mathbf{Q}_L$) of $\mathbf{C}_L$ is fixed and known. It is well-known that the information in RegulonDB is still incomplete and hard to obtain for organisms different than *E. coli*. Our goal here is thus to obtain similar transcription factor activities to those found by Kao et al. (2004) without using the information from RegulonDB, but taking into account that the data at hand is a time series by letting each transcription factor activity have an independent Gaussian process prior as described for CSLIM in Section 3.4. We will not attempt to use $\mathbf{Q}_L$ to recover the ground truth connectivity information since RegulonDB is collected from a wide range of experimental conditions and not only from the transcriptional activity produced by the *E. coli* during its transition from glucose to acetate. The results are shown in Figure 17.

Results in Figure 17(e) show the source matrix $\mathbf{Z}$ recovered by our model together with those from NCA.[13] In this experiment we ran a single chain and collected 6000 samples after a burn-in period of 2000 samples (approximately 10 minutes in a desktop machine). Most of the profiles obtained by our method are similar to those obtained by NCA (Kao et al., 2004). We ran two versions of our model, one with $\mathbf{Q}_L$ fixed to the RegulonDB values, that is, similar in spirit to NCA, and another when we infer $\mathbf{Q}_L$ without any restriction. The results of NCA and our model with fixed $\mathbf{Q}_L$ are directly comparable (up to scaling) whereas we had to match the permutation $\mathbf{P}_{\mathrm{f}}$ of the unrestricted model to those found by NCA in order to compare, using the Hungarian algorithm. Figure 17(a) shows the mixing matrices obtained by NCA and our two models. Figures 17(a) and 17(b) are very similar due to the restriction imposed on $\mathbf{Q}_L$. The mixing matrix obtained by our unrestricted model in Figure 17(c) is clearly denser than the other two, suggesting that there are different ways of connecting genes and transcription factors and still reconstruct the transcription factor activities given the observed gene expression data. When looking to the test log-likelihood

11. Data available at `http://www.seas.ucla.edu/~liaoj/NCA_module_Data`.
12. RegulonDB can be found at `http://regulondb.ccg.unam.mx/`.
13. Matlab package (v.2.3) available at `http://www.seas.ucla.edu/~liaoj/download.htm`.

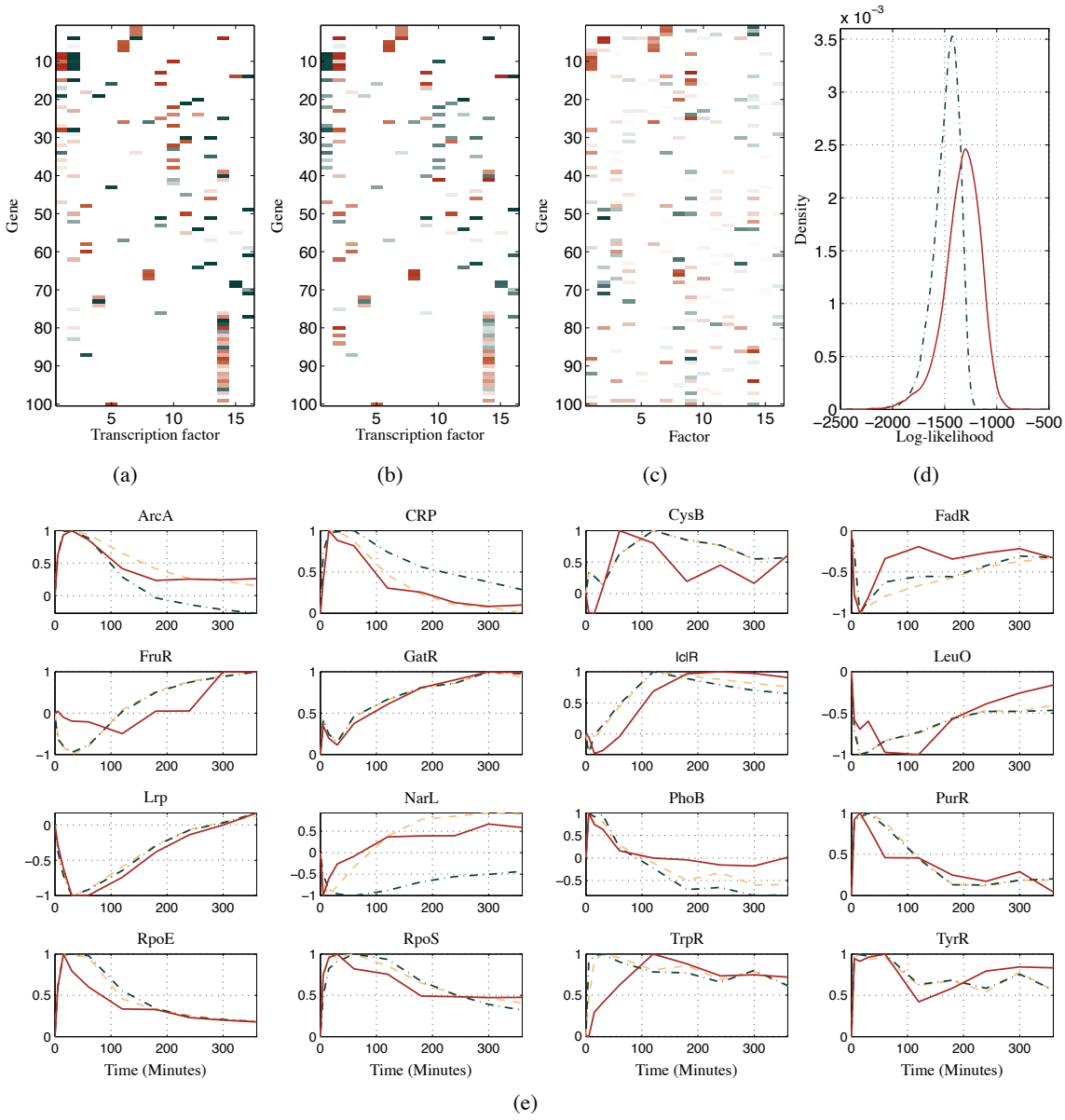


Figure 17: Results for *E. coli* data set. Mixing matrices estimated using: (a) NCA, (b) our formulation when restricting $\mathbf{Q}_L$ using RegulonDB information and (c) the factor model. (d) Model comparison results using test likelihoods. The restricted model (dash-dotted line) obtained a median negative log-likelihood of 1463.4 whereas the unrestricted model (solid line) obtained 1317.1, suggesting no significant model preferences. (e) Estimated transcription factor activities, $\mathbf{Z}$. Our methods (solid and dash-dotted lines for unrestricted and restricted model respectively) produce similar results to those produced by NCA (dashed line).

densities obtained by our two models in Figure 17(d) they are very similar, which suggests that there is no evidence that one of the models makes a better fit on test data. In terms of Mean Squared Error (MSE), NCA obtained 0.0146 while our model reached 0.0264 and 0.0218 on the restricted and unrestricted models, respectively, when using 90% of the data for inference. In addition, the 95% credible intervals for the MSE were $(0.0231, 0.0329)$ and $(0.0164, 0.0309)$ respectively. The latter shows again that there is no evidence that one of the three models is better than the other two, considering that: (i) NCA is trained on the entire data set and (ii) our unrestricted model could, in principle, produce mixing matrices arbitrarily denser than the connectivity matrix extracted from RegulonDB, and thus, again in principle, lower MSE values.

7. Discussion

We have proposed a novel approach called SLIM (Sparse Linear Identifiable Multivariate modeling) to perform inference and model comparison of general linear Bayesian networks within the same framework. The key ingredients for our Bayesian models are slab and spike priors to promote sparsity, heavy-tailed priors to ensure identifiability and predictive densities (test likelihoods) to perform the comparison. A set of candidate orderings is produced by stochastic search during the factor model inference. Subsequently, a linear DAG with or without latent variables is learned for each of the candidates. To the authors' knowledge this is the first time that a method for comparing such closely related linear models has been proposed. This setting can be very beneficial in situations where the prior evidence suggests both DAG structure and/or unmeasured variables in the data. We also show that the DAG with latent variables can be fully identifiable and that SLIM can be extended to the non-linear case (SNIM - Sparse Non-linear Identifiable Multivariate modeling), if the ordering of the variables is provided or can be tested by exhaustive enumeration. For example in the protein-signaling network (Sachs et al., 2005), the textbook ground truth suggests both DAG structure and a number of unmeasured proteins. The previous approach (Sachs et al., 2005) only performed structure learning in pure DAGs but our results using observational data alone suggest that the data is better explained by a (possibly non-linear) DAG with latent variables. Our extensive results on artificial data showed one by one the features of our model in each one of its variants, and demonstrated empirically their usefulness and potential applicability. When comparing against LiNGAM, our method always performed at least as well in every case with a comparable computational cost. The presented Bayesian framework also allows easy extension of our model to match different prior beliefs about the problems at hand without significantly changing the model and its conceptual foundations, as in CSLIM and SNIM.

We believe that the priors that give raise to sparse models in the fully Bayesian inference setting, like the two-level slab (continuous) and spike (point-mass in zero) priors used are very powerful tools for simultaneous model and parameter inference. They may be useful in many settings in machine learning where sparsity of parameters is desirable. Although the posterior distributions for slab and spike priors will be non-convex, it is our experience that inference with blocked Gibbs sampling actually has very good convergence properties. In the two-level approach, one uses a hierarchy of two slab and spike priors. The first is on the parameter and the second is on the mixture parameter (i.e., the probability that the parameter is non-zero). Instead of letting this parameter be controlled by a single Beta-distribution (one level approach) we have a slab and spike distribution on it with a Beta-distributed slab component biased towards one. This makes the model more

parsimonious, that is, the probability that parameters are zero or non-zero is closer to zero and one and parameter settings are more robust.

In the following we will discuss open questions and future directions. From the Bayesian network repository experiment it is clear that we need to improve our ordering search procedure if we want to use SLIM for problems with more than say 50 variables. This basically amounts to finding proposal distributions that better exploit the particularities of the model at hand. Another option could be to provide the proposal distribution with some notion of memory to avoid permutations with low probability and/or expand the coverage of the searching procedure.

It is well studied in the literature on sparse models that for increasing number of observations any model tends to loose its sparsity capabilities. This is because the likelihood starts dominating the inference, making the prior distribution less informative. The easiest way to handle such an effect is to make the hyperparameters of the sparsity prior dependent on N. We have not explored this phenomenon in SLIM but it should certainly be taken into account in the specification of sparsity priors.

Directly specifying the distributions of the latent variables in order to obtain identifiability in the general DAG with latent variables requires having different distributions for the driving signals of the observed variables and latent variables. This may introduce model mismatch or be restrictive in some cases as one will not have this kind of knowledge a priori. We thus need more principled ways to specify distributions for $\mathbf{z}$ ensuring identifiably, without restricting some of its components to having a particular behavior, like having heavier tails than the driving signals for instance. We conjecture that providing $\mathbf{z}$ with a parameterization of Dirichlet process priors with appropriate base measures would be enough but we are not certain whether this would be sufficient in practice.

We set a priori that the components of $\mathbf{z}$ are independent. Although this is a very reasonable assumption, it does not allow for connectivity between latent variables as we see for example in the protein signaling network, see Figure 15(a). It is straight forward to specify such a model, although identifiability becomes even harder to ensure in this case.

We do not have an ordering search procedure for the non-linear version of SLIM. This is a necessary step since exhaustive enumeration of all possible orderings is not an option beyond say 10 variables. The main problem is that the non-linear DAG has no equivalent factor model representation so we cannot directly exploit the permutation candidates we find in SLIM. However, as long as the non-linearities are weak, one might in principle use the permutation candidates found in a factor model, that is, the linear effects will determine the correct ordering of the variables.

SLIM cannot handle experimental (interventional) data, and consequently around 80% of the data from the Sachs et al. (2005) study is not used. It is well-established how to learn with interventions in DAGs (see Sachs et al., 2005). The problem remains of how to formulate effective inference with interventional data in the factor model.

Acknowledgments

We thank the editor and the three anonymous referees for their helpful comments and discussions that improved the presentation of this paper. This work is supported by a grant from the Novo Nordisk Foundation to the Bioinformatics Centre, University of Copenhagen.

Appendix A. Gibbs Sampling

Given a set of N observations in d dimensions, the data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ and m latent variables, MCMC analysis is standard and can be implemented through Gibbs sampling. Note that in the following, $\mathbf{X}_{i:}$ and $\mathbf{X}_{:i}$ are rows and columns of $\mathbf{X}$, respectively, and i, j, n are indexes for dimensions, factors and observations, respectively. In the following we describe the conditional distributions needed to sample from the standard factor model hierarchy. Below we will briefly discus the modifications needed for the DAG.

A.1 Noise Variance

We can sample each element of $\mathbf{\Psi}$ independently using

$$\psi_i^{-1}|\mathbf{X}_{i:}, \mathbf{C}_{i:}, \mathbf{Z}, \mathbf{V}_i, s_s, s_r \sim \text{Gamma}\left(\psi_i^{-1} \middle| s_s + \frac{N+d}{2}, s_r + u\right) , \tag{13}$$

where $\mathbf{V}_i$ is a diagonal matrix with entries τ_{ij} and

$$u = \frac{1}{2}(\mathbf{X}_{i:} - \mathbf{C}_{i:}\mathbf{Z})(\mathbf{X}_{i:} - \mathbf{C}_{i:}\mathbf{Z})^\top + \frac{1}{2}\mathbf{C}_{i:}\mathbf{V}_i^{-1}\mathbf{C}_{i:}^\top .$$

A.2 Factors

The conditional distribution of the latent variables $\mathbf{Z}$ using the scale mixtures of Gaussians representation can be computed independently for each element of z_{jn} using

$$z_{jn}|\mathbf{X}_{:n}, \mathbf{C}_{:j}, \mathbf{Z}_{:n}, \mathbf{\Psi}, \upsilon_{jn} \sim \mathcal{N}(z_{jn}|\mathbf{C}_{:j}^\top \mathbf{\Psi}^{-1}\boldsymbol{\varepsilon}_{\setminus jn}, u_{jn}) , \tag{14}$$

where $u_{jn} = (\mathbf{C}_{:j}^\top \mathbf{\Psi}^{-1}\mathbf{C}_{:j} + \upsilon_{jn}^{-1})^{-1}$ and $\boldsymbol{\varepsilon}_{\setminus jn} = \mathbf{X}_{:n} - \mathbf{C}\mathbf{Z}_{:n}|_{z_{jn}=0}$. If the latent factors are Laplace distributed the mixing variances υ_{jn} have exponential distribution, thus the resulting conditional is

$$\upsilon_{jn}^{-1}|z_{jn}, \lambda \sim \text{IG}\left(\upsilon_{jn}^{-1} \middle| \frac{\lambda}{|z_{jn}|}, \lambda^2\right) ,$$

and for the Student's t, with corresponding gamma densities as

$$\upsilon_{jn}^{-1}|z_{jn}, \sigma^2, \theta \sim \text{Gamma}\left(\upsilon_{jn}^{-1} \middle| \frac{\theta+1}{2}, \frac{\theta}{2} + \frac{z_{jn}^2}{2\sigma^2}\right) ,$$

where $\text{IG}(\cdot|\mu, \lambda)$ is the inverse Gaussian distribution with mean μ and scale parameter λ (Chhikara and Folks, 1989).

A.3 Gaussian Processes

In practice, the prior distribution for each row of the matrix $\mathbf{Z}$ in CSLIM has the form $z_{j1}, \ldots, z_{jN} \sim \mathcal{N}(0, \mathbf{K}_j)$, where $\mathbf{K}_j$ is a covariance matrix of size $N \times N$ built using $k_{\upsilon_j,n}(n, n')$. The conditional distribution for $z_{j1}, \ldots, z_{jN}$ can be computed using

$$z_{j1}, \ldots, z_{jN}|\mathbf{X}, \mathbf{C}_{j:}, \mathbf{Z}_{\setminus j}, \mathbf{\Psi} \sim \mathcal{N}(z_{j1}, \ldots, z_{jN}|\mathbf{C}_{:j}^\top \mathbf{\Psi}^{-1}\boldsymbol{\varepsilon}_{\setminus j}\mathbf{V}, \mathbf{V}) ,$$

where $\mathbf{Z}_{\setminus j}$ is $\mathbf{Z}$ without row j, $\mathbf{V} = (\mathbf{U} + \mathbf{K}_j^{-1})^{-1}$, $\mathbf{U}$ is a diagonal matrix with elements $\mathbf{C}_{:j}^\top \mathbf{\Psi}^{-1} \mathbf{C}_{:j}$ and $\boldsymbol{\varepsilon}_{\setminus j} = \mathbf{X} - \mathbf{C}\mathbf{Z}|_{z_{j1},\ldots,z_{jN}=0}$. The computation of $\mathbf{V}$ can be done in a numerically stable way by rewriting $\mathbf{V} = \mathbf{K}_j - \mathbf{K}_j(\mathbf{U}^{-1} + \mathbf{K}_j)^{-1}\mathbf{K}_j$ and then using Cholesky decomposition and back substitution to obtain in turn $\mathbf{L}\mathbf{L}^\top = \mathbf{U}^{-1} + \mathbf{K}_j$ and $\mathbf{L}^{-1}\mathbf{K}_j$. The hyperparameters of the covariance function in Equation (9) can be sampled using

$$\kappa|\boldsymbol{\upsilon}, k_s, k_r \sim \text{Gamma}\left(\kappa \middle| k_s + mu_s, k_r + \sum_{j=1}^{m} \upsilon_j\right) .$$

For the inverse length-scales we use Metropolis-Hastings updates with proposal $q(\upsilon_j^\star|\upsilon_j) = p(\upsilon_j^\star)$ and acceptance ratio

$$\xi_{\to\star} = \frac{\mathcal{N}(z_{j1},\ldots,z_{jN}|\mathbf{0},\mathbf{K}_j^\star)}{\mathcal{N}(z_{j1},\ldots,z_{jN}|\mathbf{0},\mathbf{K}_j)} ,$$

where $\mathbf{K}_j^\star$ is obtained using $k_{\upsilon_j^\star,n}(n,n')$. For SNIM, we only need to replace $\mathbf{C}$ by $\mathbf{B}$, $\mathbf{Z}$ by $\mathbf{Y} = [\mathbf{y}_1 \ \ldots \mathbf{y}_N]$ and $k_{\upsilon_j,n}(n,n')$ by $k_{\upsilon_i,x}(\mathbf{x},\mathbf{x}')$.

A.4 Mixing Matrix

In order to sample each c_{ij} from the conditional distribution of the matrix $\mathbf{C}$ we use

$$c_{ij}|\mathbf{X}_{i:}, \mathbf{C}_{\setminus ij}, \mathbf{Z}_{j:}, \psi_i, \tau_{ij} \sim \mathcal{N}(c_{ij}|u_{ij}\boldsymbol{\varepsilon}_{\setminus ij}\mathbf{Z}_{j:}^\top, u_{ij}\psi_i) , \tag{15}$$

where $u_{ij} = (\mathbf{Z}_{j:}\mathbf{Z}_{j:}^\top + \tau_{ij}^{-1})^{-1}$ and $\boldsymbol{\varepsilon}_{\setminus ij} = \mathbf{X}_{i:} - \mathbf{C}_{i:}\mathbf{Z}|_{d_{ij}=0}$. Note that we only need to sample those c_{ij} for which $r_{ij} = 1$, that is, just the slab distribution. Sampling from the conditional distributions for τ_{ij} can be done using

$$\tau_{ij}^{-1}|d_{jn}, t_s, t_r \sim \text{Gamma}\left(\tau_{ij}^{-1} \middle| t_s + \frac{1}{2}, t_r + \frac{d_{ij}^2}{2\psi_i}\right) . \tag{16}$$

The conditional distributions for the remaining parameters in the slab and spike prior can be written first for the masking matrix $\mathbf{Q}$ as

$$q_{ij}|\mathbf{X}_{i:}, \mathbf{D}_{i:}, \mathbf{Z}, \psi_i, \tau_{ij}, \eta_{ij} \sim \text{Bernoulli}\left(q_{ij} \middle| \frac{\xi_{\eta_{ij}}}{1+\xi_{\eta_{ij}}}\right) , \tag{17}$$

where

$$\xi_{\eta_{ij}} = \frac{\alpha_m \nu_j}{1-\alpha_m\nu_j} \frac{\psi_i^{1/2}}{(\mathbf{Z}_{j:}\mathbf{Z}_{j:}^\top + \tau_{ij}^{-1})^{1/2}} \exp\left(\frac{(\boldsymbol{\varepsilon}_{\setminus ij}\mathbf{Z}_{j:}^\top)^2}{2\psi_i(\mathbf{Z}_{j:}\mathbf{Z}_{j:}^\top + \tau_{ij}^{-1})}\right) ,$$

and the probability of each element of $\mathbf{C}$ of being non-zero as

$$\eta_{ij}|u_{ij}, q_{ij}, \alpha_p, \alpha_m \sim (1-u_{ij})\delta(\eta_{ij}) + u_{ij}\text{Beta}(\eta_{ij}|\alpha_p\alpha_m + q_{ij}, \alpha_p(1-\alpha_m) + 1 - q_{ij}) , \tag{18}$$

where $u_{ij} \sim \text{Bernoulli}(h_{ij}|r_{ij} + (1-r_{ij})\nu_j(1-\alpha_m)/(1-\nu_j\alpha_m))$, that is, we set $u_{ij} = 1$ if $q_{ij} = 1$. Finally, for the column-wise shared sparsity rate we have

$$\nu_j|\mathbf{u}_j, \beta_p, \beta_m \sim \text{Beta}\left(\nu_j \middle| \beta_p\beta_m + \sum_{i=1}^{d} u_{ij}, \beta_p(1-\beta_m) + \sum_{i=1}^{d}(1-u_{ij})\right) . \tag{19}$$

Sampling from the DAG model only requires minor changes in notation but the conditional posteriors are essentially the same. The changes mostly amount to replacing accordingly $\mathbf{C}$ by $\mathbf{B}$ and $\mathbf{Q}$ by $\mathbf{R}$. Note that $\mathbf{Q}_L$ is the identity and $\mathbf{R}$ is strictly lower triangular a priori, thus we only need to sample their active elements.

A.5 Inference with Missing Values

We introduce a binary masking matrix indicating whether an element of $\mathbf{X}$ is missing or not. For the factor model we have the following modified likelihood

$$p(\mathbf{X}_{\text{tr}}|\mathbf{C},\mathbf{Z},\mathbf{\Psi},\mathbf{M}_{\text{miss}}) = \mathcal{N}(\mathbf{M}_{\text{miss}} \odot \mathbf{X}|\mathbf{M}_{\text{miss}} \odot (\mathbf{CZ}),\mathbf{\Psi}) \ .$$

Testing on the missing values, $\mathbf{M}^{\star}_{\text{miss}} = \mathbf{1}\mathbf{1}^{\top} - \mathbf{M}$ requires averaging the test likelihood

$$p(\mathbf{X}^{\star}|\mathbf{C},\mathbf{Z},\mathbf{\Psi},\mathbf{M}^{\star}_{\text{miss}}) = \mathcal{N}(\mathbf{M}^{\star}_{\text{miss}} \odot \mathbf{X}|\mathbf{M}^{\star}_{\text{miss}} \odot (\mathbf{CZ}),\mathbf{\Psi}) \ ,$$

over $\mathbf{C},\mathbf{Z},\mathbf{\Psi}$ given $\mathbf{X}_{\text{tr}}$ (training). We can approximate the predictive density $p(\mathbf{X}^{\star}|\mathbf{X}_{\text{tr}},\cdot)$ by computing the likelihood above during sampling using the conditional posteriors of $\mathbf{C},\mathbf{Z}$ and $\mathbf{\Psi}$ and then summarizing using for example the median. Drawing from $\mathbf{C}$, $\mathbf{Z}$, $\mathbf{\Psi}$ can be achieved by sampling from their respective conditional distributions as described before with some minor modifications.

References

D. F. Andrews and C. L. Mallows. Scale mixtures of normal distributions. *Journal of the Royal Statistical Society: Series B (Methodology)*, 36(1):99–102, 1974.

A. Asuncion and D .J. Newman. UCI machine learning repository, 2007.

A. Azzalini and A. W. Bowman. A look at some data on the Old Faithful geyser. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 39(3):357–365, 1990.

P. Bekker and J. M. F. ten Berge. Generic global indentification in factor analysis. *Linear Algebra and its Applications*, 264(1–3):255–263, 1997.

M. Branco and D. K. Dey. A general class of multivariate skew-elliptical distributions. *Journal of Multivariate Analysis*, 79(1):99–113, 2001.

C. M. Carvalho, J. Chang, J. E. Lucas, J. R. Nevins, Q. Wang, and M. West. High-dimensional sparse factor modeling: Applications in gene expression genomics. *Journal of the American Statistical Association*, 103(484):1438–1456, 2008.

R. S. Chhikara and L. Folks. *The Inverse Gaussian Distribution: Theory, Methodology, and Applications*. M. Dekker, New York, 1989.

S. Chib. Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, 90(732):1313–1321, 1995.

D. M. Chickering. Learning Bayesian networks is NP-complete. In D. Fisher and H.-J. Lenz, editors, *Learning from Data: AI and Statistics*, pages 121–130. Springer-Verlag, 1996.

P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3):287–314, 1994.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

P. Daniusis, J. Janzing, J. Mooij, J. Zscheischler, B. Steudel, K. Zhang, and B. Schölkopf. Inferring deterministic causal relations. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 2010.

A. P Dawid and S. L Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, 21(3):1272–1317, 1993.

A. P. Dempster. Covariance selection. *Biometrics*, 28:157–175, 1972.

G. Elidan, I. Nachman, and N. Friedman. "Ideal Parent" structure learning for continuous variable Bayesian networks. *Journal of Machine Learning Research*, 8:1799–1833, 2007.

N. Friedman and D. Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1–2):95–125, 2003.

N. Friedman and I. Nachman. Gaussian process networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 211–219. 2000.

N. Friedman, I. Nachman, and D. Pe'er. Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm. In K. B. Laskey and H. Prade, editors, *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 206–215, 1999.

N. Friel and A. N. Pettitt. Marginal likelihood estimation via power posteriors. *Journal of the Royal Statistical Society: Series B (Methodology)*, 70(3):589–607, 2008.

S. Gama-Castro, V. Jiménez-Jacinto, M. Peralta-Gil, A. Santos-Zavaleta, M. I. Peñaloza-Spinola, B. Contreras-Moreira, J. Segura-Salazar, L. Muñiz-Rascado, I. Martínez-Flores, H. Salgado, C. Bonavides-Martínez, C. Abreu-Goodger, C. Rodríguez-Penagos, J. Miranda-Ríos, E. Morett, E. Merino, A. M. Huerta, L. Treviño-Quintanilla, and J. Collado-Vides. RegulonDB (version 6.0): gene regulation model of Escherichia coli K-12 beyond transcription, active (experimental) annotated promoters and textpresso navigation. *Nucleic Acids Research*, 36(Database Issue): 120–124, 2008.

E. I. George and R. E. McCulloch. Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881–889, 1993.

J. Geweke. Variable selection and model comparison in regression. In J. Berger, J. Bernardo, A. Dawid, and A. Smith, editors, *Bayesian Statistics 5*, pages 609–620. Oxford University Press, 1996.

Z. Ghahramani, T. L. Griffiths, and P. Sollich. Bayesian nonparametric latent feature models. In J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 8*, pages 201–226. Oxford University Press, 2006.

P. Giudici and P. J Green. Decomposable graphical Gaussian model determination. *Biometrika*, 86 (4):785–801, 1999.

A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 585–592. MIT Press, 2008.

D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.

R. Henao and O. Winther. Bayesian sparse factor models and DAGs inference and comparison. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 736–744. The MIT Press, 2009.

P. O. Hoyer, S. Shimizu, A. J. Kerminen, and M. Palviainen. Estimation of causal effects using linear non-Gaussian causal models with hidden variables. *Interantional Journal of Approximate Reasoning*, 49(2):362–378, 2008.

P .O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 689–696. 2009.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley-Interscience, 2001.

H. Ishwaran and J. S. Rao. Spike and slab variable selection: Frequentist and Bayesian strategies. *Annals of Statistics*, 33(2):730–773, 2005.

I. T. Jolliffe, N. T. Trendafilov, and M. Uddin. A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics*, 12(3):531–547, 2003.

A. M. Kagan, YU. V Linnik, and C. Radhakrishna Rao. *Characterization Problems in Mathematical Statistics*. Probability and Mathematical Statistics. Wiley, New York, 1973.

K. C. Kao, Y-L. Yang, R. Boscolo, C. Sabatti, V. Roychowdhury, and J. C. Liao. Transcriptome-based determination of multiple transcription regulator activities in Escherichia Coli by using network component analysis. *PNAS*, 101(2):641–646, 2004.

D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In M. E. Davies, C. C. James, S. A. Abdallah, and M. D. Plumbley, editors, *7th International Conference on Independent Component Analysis and Signal Separation*, volume 4666 of *Lecture Notes in Computer Science*, pages 381–388. Springer-Verlag, Berlin, 2007.

F. B. Lempers. *Posterior Probabilities of Alternative Linear Models*. Rotterdam University Press, 1971.

H. F. Lopes and M. West. Bayesian model assessment in factor analysis. *Statistica Sinica*, 14(1): 41–67, 2004.

J. Lucas, C. Carvalho, Q. Wang, A. Bild, J. R. Nevins, and M. West. *Bayesian Inference for Gene Expression and Proteomics*, chapter Sparse Statistical Modeling in Gene Expression Genomics, pages 155–176. Cambridge University Press, 2006.

J. K. Martin and R. P. McDonald. Bayesian estimation in unrestricted factor analysis: A treatment for heywood cases. *Psychometrika*, 40(4):505–517, 1975.

T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

J. Mooij and D. Janzing. Distinguishing between cause and effect. In *JMLR Workshop and Conference Proceedings*, volume 6, pages 147–156, 2010.

I. Murray. *Advances in Markov Chain Monte Carlo Methods*. PhD thesis, Gatsby computational neuroscience unit, University College London, 2007.

R. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

T. Park and G. Casella. The Bayesian lasso. *Journal of the American Statistical Association*, 103 (482):681–686, 2008.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

P. Rai and H. Daume III. The infinite hierarchical factor regression model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1321–1328. The MIT Press, 2009.

B. Rajaratman, H. Massam, and C. Carvalho. Flexible covariance estimation in graphical gaussian models. *Annals of Statistics*, 36(6):2818–2849, 2008.

J. M. Robins, R. Scheines, P. Spirtes, and L. Wasserman. Uniform consistency in causal inference. *Biometrika*, 90(3):491–515, 2003.

K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

M. W. Schmidt, A. Niculescu-Mizil, and K. P. Murphy. Learning graphical model structure using L1-regularization paths. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 1278–1283, 2007.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

R. Silva. *Causality in the Sciences*, chapter Measuring Latent Causal Structure. Oxford University Press, 2010.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. The MIT Press, second edition, 2001.

M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pages 548–549, 2005.

R. Thibaux and M. I. Jordan. Hierarchical beta processes and the indian buffet process. In M. Meila and X. Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, pages 564–571, 2007.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodology)*, 58(1):267–288, 1996.

R. Tillman, A. Gretton, and P. Spirtes. Nonlinear directed acyclic structure learning with weakly additive noise models. In *Advances in Neural Information Processing Systems 22*, pages 1847–1855. Y. Bengio and D. Schuurmans and J. Lafferty and C. K. I. Williams and A. Culotta, 2009.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

M. West. On scale mixtures of normal distributions. *Biometrika*, 74(3):646–648, 1987.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. In J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 7*, pages 723–732. Oxford University Press, 2003.

S. Yu, V. Tresp, and K. Yu. Robust multi-task learning with t-processes. In *Proceedings of the 24th International Conference on Machine Learning*, volume 227, pages 1103–1110, 2007.

K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pages 647–655. AUAI Press, 2009.

K. Zhang and A. Hyvärinen. Distinguishing causes from effect using nonlinear acyclic causal models. In *JMLR Workshop and Conference Proceedings*, volume 6, pages 157–164, 2010.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):262–286, 2006.

Journal of Machine Learning Research 12 (2011) 907-951 Submitted 1/10; Revised 10/10; Published 3/11

Forest Density Estimation

Han Liu HANLIU@CS.JHU.EDU
Department of Biostatistics and Computer Science
Johns Hopkins University
Baltimore, MD 21210, USA

Min Xu MINX@CS.CMU.EDU
Haijie Gu HAIJIE@CS.CMU.EDU
Anupam Gupta ANUPAMG@CS.CMU.EDU
John Lafferty* LAFFERTY@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Larry Wasserman LARRY@STAT.CMU.EDU
Department of Statistics
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Editor: Tong Zhang

Abstract

We study graph estimation and density estimation in high dimensions, using a family of density estimators based on forest structured undirected graphical models. For density estimation, we do not assume the true distribution corresponds to a forest; rather, we form kernel density estimates of the bivariate and univariate marginals, and apply Kruskal's algorithm to estimate the optimal forest on held out data. We prove an oracle inequality on the excess risk of the resulting estimator relative to the risk of the best forest. For graph estimation, we consider the problem of estimating forests with restricted tree sizes. We prove that finding a maximum weight spanning forest with restricted tree size is NP-hard, and develop an approximation algorithm for this problem. Viewing the tree size as a complexity parameter, we then select a forest using data splitting, and prove bounds on excess risk and structure selection consistency of the procedure. Experiments with simulated data and microarray data indicate that the methods are a practical alternative to Gaussian graphical models.

Keywords: kernel density estimation, forest structured Markov network, high dimensional inference, risk consistency, structure selection consistency

1. Introduction

One way to explore the structure of a high dimensional distribution P for a random vector $X = (X_1, \ldots, X_d)$ is to estimate its undirected graph. The undirected graph G associated with P has d vertices corresponding to the variables $X_1, \ldots, X_d$, and omits an edge between two nodes X_i and X_j if and only if X_i and X_j are conditionally independent given the other variables. Currently, the most popular methods for estimating G assume that the distribution P is Gaussian. Finding the

*. John Lafferty is also in the Department of Statistics at Carnegie Mellon University.

©2011 Han Liu, Min Xu, Haijie Gu, Anupam Gupta, John Lafferty and Larry Wasserman.

graphical structure in this case amounts to estimating the inverse covariance matrix Ω; the edge between X_j and X_k is missing if and only if $\Omega_{jk} = 0$. Algorithms for optimizing the ℓ_1-regularized log-likelihood have recently been proposed that efficiently produce sparse estimates of the inverse covariance matrix and the underlying graph (Banerjee et al., 2008; Friedman et al., 2007).

In this paper our goal is to relax the Gaussian assumption and to develop nonparametric methods for estimating the graph of a distribution. Of course, estimating a high dimensional distribution is impossible without making any assumptions. The approach we take here is to force the graphical structure to be a forest, where each pair of vertices is connected by at most one path. Thus, we relax the distributional assumption of normality but we restrict the family of undirected graphs that are allowed.

If the graph for P is a forest, then a simple conditioning argument shows that its density p can be written as

$$p(x) = \prod_{(i,j)\in E} \frac{p(x_i,x_j)}{p(x_i)p(x_j)} \prod_{k=1}^{d} p(x_k)$$

where E is the set of edges in the forest (Lauritzen, 1996). Here $p(x_i,x_j)$ is the bivariate marginal density of variables X_i and X_j, and $p(x_k)$ is the univariate marginal density of the variable X_k. With this factorization, we see that it is only necessary to estimate the bivariate and univariate marginals. Given any distribution P with density p, there is a tree T and a density p_T whose graph is T and which is closest in Kullback-Leibler divergence to p. When P is known, then the best fitting tree distribution can be obtained by Kruskal's algorithm (Kruskal, 1956), or other algorithms for finding a maximum weight spanning tree. In the discrete case, the algorithm can be applied to the estimated probability mass function, resulting in a procedure originally proposed by Chow. and Liu (1968). Here we are concerned with continuous random variables, and we estimate the bivariate marginals with nonparametric kernel density estimators.

In high dimensions, fitting a fully connected spanning tree can be expected to overfit. We regulate the complexity of the forest by selecting the edges to include using a data splitting scheme, a simple form of cross validation. In particular, we consider the family of forest structured densities that use the marginal kernel density estimates constructed on the first partition of the data, and estimate the risk of the resulting densities over a second, held out partition. The optimal forest in terms of the held out risk is then obtained by finding a maximum weight spanning forest for an appropriate set of edge weights.

A closely related approach is proposed by Bach and Jordan (2003), where a tree is estimated for the random vector $Y = WX$ instead of X, where W is a linear transformation, using an algorithm that alternates between estimating W and estimating the tree T. Kernel density estimators are used, and a regularization term that is a function of the number of edges in the tree is included to bias the optimization toward smaller trees. We omit the transformation W, and we use a data splitting method rather than penalization to choose the complexity of the forest.

While tree and forest structured density estimation has been long recognized as a useful tool, there has been little theoretical analysis of the statistical properties of such density estimators. The main contribution of this paper is an analysis of the asymptotic properties of forest density estimation in high dimensions. We allow both the sample size n and dimension d to increase, and prove oracle results on the risk of the method. In particular, we assume that the univariate and bivariate

marginal densities lie in a Hölder class with exponent β (see Section 4 for details), and show that

$$R(\widehat{p}_{\widehat{F}}) - \min_F R(\widehat{p}_F) = O_P\left(\sqrt{\log(nd)}\left(\frac{k^* + \widehat{k}}{n^{\beta/(2+2\beta)}} + \frac{d}{n^{\beta/(1+2\beta)}}\right)\right)$$

where R denotes the risk, the expected negative log-likelihood, $\widehat{k}$ is the number of edges in the estimated forest $\widehat{F}$, and k^* is the number of edges in the optimal forest F^* that can be constructed in terms of the kernel density estimates $\widehat{p}$.

In addition to the above results on risk consistency, we establish conditions under which

$$\mathbb{P}\left(\widehat{F}_d^{(k)} = F_d^{*(k)}\right) \to 1$$

as $n \to \infty$, where $F_d^{*(k)}$ is the *oracle forest*—the best forest with k edges; this result allows the dimensionality d to increase as fast as $o\left(\exp(n^{\beta/(1+\beta)})\right)$, while still having consistency in the selection of the oracle forest.

Among the only other previous work analyzing tree structured graphical models is Tan et al. (2011) and Chechetka and Guestrin (2007). Tan et al. (2011) analyze the error exponent in the rate of decay of the error probability for estimating the tree, in the fixed dimension setting, and Chechetka and Guestrin (2007) give a PAC analysis. An extension to the Gaussian case is given by Tan et al. (2010).

We also study the problem of estimating forests with restricted tree sizes. In many applications, one is interested in obtaining a graphical representation of a high dimensional distribution to aid in interpretation. For instance, a biologist studying gene interaction networks might be interested in a visualization that groups together genes in small sets. Such a clustering approach through density estimation is problematic if the graph is allowed to have cycles, as this can require marginal densities to be estimated with many interacting variables. Restricting the graph to be a forest circumvents the curse of dimensionality by requiring only univariate and bivariate marginal densities. The problem of clustering the variables into small interacting sets, each supported by a tree-structured density, becomes the problem of estimating a maximum weight spanning forest with a restriction on the size of each component tree. As we demonstrate, estimating restricted tree size forests can also be useful in model selection for the purpose of risk minimization. Limiting the tree size gives another way of regulating tree complexity that provides larger family of forest to select from in the data splitting procedure.

While the problem of finding a maximum weight forest with restricted tree size may be natural, it appears not to have been studied previously. We prove that the problem is NP-hard through a reduction from the problem of Exact 3-Cover (Garey and Johnson, 1979), where we are given a set X and a family $\mathcal{S}$ of 3-element subsets of X, and must choose a subfamily of disjoint 3-element subsets to cover X. While finding the exact optimum is hard, we give a practical 4-approximation algorithm for finding the optimal tree restricted forest; that is, our algorithm outputs a forest whose weight is guaranteed to be at least $\frac{1}{4}w(F^*)$, where $w(F^*)$ is the weight of the optimal forest. This approximation guarantee translates into excess risk bounds on the constructed forest using our previous analysis. Our experimental results with this approximation algorithm show that it can be effective in practice for forest density estimation.

In Section 2 we review some background and notation. In Section 3 we present a two-stage algorithm for estimating high dimensional densities supported by forests, and we provide a theoretical

analysis of the algorithm in Section 4, with the detailed proofs collected in an appendix. In Section 5, we explain how to estimate maximum weight forests with restricted tree size. In Section 6 we present experiments with both simulated data and gene microarray data sets, where the problem is to estimate the gene-gene association graphs.

2. Preliminaries and Notation

Let $p^*(x)$ be a probability density with respect to Lebesgue measure $\mu(\cdot)$ on $\mathbb{R}^d$ and let $X^{(1)},\ldots,X^{(n)}$ be n independent identically distributed $\mathbb{R}^d$-valued data vectors sampled from $p^*(x)$ where $X^{(i)} = (X_1^{(i)},\ldots,X_d^{(i)})$. Let $\mathcal{X}_j$ denote the range of $X_j^{(i)}$ and let $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_d$. For simplicity we assume that $\mathcal{X}_j = [0,1]$.

A graph is a forest if it is acyclic. If F is a d-node undirected forest with vertex set $V_F = \{1,\ldots,d\}$ and edge set $E(F) \subset \{1,\ldots,d\} \times \{1,\ldots,d\}$, the number of edges satisfies $|E(F)| < d$, noting that we do not restrict the graph to be connected. We say that a probability density function $p(x)$ is *supported by a forest* F if the density can be written as

$$p_F(x) = \prod_{(i,j)\in E(F)} \frac{p(x_i,x_j)}{p(x_i)\,p(x_j)} \prod_{k\in V_F} p(x_k), \tag{1}$$

where each $p(x_i,x_j)$ is a bivariate density on $\mathcal{X}_i \times \mathcal{X}_j$, and each $p(x_k)$ is a univariate density on $\mathcal{X}_k$. More details can be found in Lauritzen (1996).

Let $\mathcal{F}_d$ be the family of forests with d nodes, and let $\mathcal{P}_d$ be the corresponding family of densities:

$$\mathcal{P}_d = \left\{ p \geq 0 : \int_{\mathcal{X}} p(x)\,d\mu(x) = 1, \text{ and } p(x) \text{ satisfies (1) for some } F \in \mathcal{F}_d \right\}. \tag{2}$$

To bound the number of labeled spanning forests on d nodes, note that each such forest can be obtained by forming a labeled tree on $d+1$ nodes, and then removing node $d+1$. From Cayley's formula (Cayley, 1889; Aigner and Ziegler, 1998), we then obtain the following.

Proposition 1 *The size of the collection $\mathcal{F}_d$ of labeled forests on d nodes satisfies*

$$|\mathcal{F}_d| < (d+1)^{d-1}.$$

Define the oracle forest density

$$q^* = \arg\min_{q\in\mathcal{P}_d} D(p^*\|\,q) \tag{3}$$

where the Kullback-Leibler divergence $D(p\|\,q)$ between two densities p and q is

$$D(p\|\,q) = \int_{\mathcal{X}} p(x) \log \frac{p(x)}{q(x)} dx,$$

under the convention that $0\log(0/q) = 0$, and $p\log(p/0) = \infty$ for $p \neq 0$. The following is proved by Bach and Jordan (2003).

Proposition 2 *Let q^* be defined as in* (3). *There exists a forest $F^* \in \mathcal{F}_d$, such that*

$$q^* = p^*_{F^*} = \prod_{(i,j)\in E(F^*)} \frac{p^*(x_i,x_j)}{p^*(x_i)\,p^*(x_j)} \prod_{k\in V_{F^*}} p^*(x_k) \tag{4}$$

where $p^*(x_i,x_j)$ and $p^*(x_i)$ are the bivariate and univariate marginal densities of p^*.

For any density $q(x)$, the negative log-likelihood risk $R(q)$ is defined as

$$R(q) = -\mathbb{E}\log q(X) = -\int_{\mathcal{X}} p^*(x)\log q(x)\,dx$$

where the expectation is defined with respect to the distribution of X.

It is straightforward to see that the density q^* defined in (3) also minimizes the negative log-likelihood loss:

$$q^* = \arg\min_{q\in\mathcal{P}_d} D(p^*\|\,q) = \arg\min_{q\in\mathcal{P}_d} R(q).$$

Let $\widehat{p}(x)$ be the kernel density estimate, we also define

$$\widehat{R}(q) = -\int_{\mathcal{X}} \widehat{p}(x)\log q(x)\,dx.$$

We thus define the oracle risk as $R^* = R(q^*)$. Using Proposition 2 and Equation (1), we have

$$\begin{aligned}
R^* &= R(q^*) \;=\; R(p^*_{F^*}) \\
&= -\int_{\mathcal{X}} p^*(x)\left(\sum_{(i,j)\in E(F^*)} \log\frac{p^*(x_i,x_j)}{p^*(x_i)p^*(x_j)} + \sum_{k\in V_{F^*}} \log\left(p^*(x_k)\right)\right)dx \\
&= -\sum_{(i,j)\in E(F^*)} \int_{\mathcal{X}_i\times\mathcal{X}_j} p^*(x_i,x_j)\log\frac{p^*(x_i,x_j)}{p^*(x_i)p^*(x_j)}dx_i dx_j - \sum_{k\in V_{F^*}}\int_{\mathcal{X}_k} p^*(x_k)\log p^*(x_k)dx_k \\
&= -\sum_{(i,j)\in E(F^*)} I(X_i;X_j) + \sum_{k\in V_{F^*}} H(X_k),
\end{aligned} \tag{5}$$

where

$$I(X_i;X_j) = \int_{\mathcal{X}_i\times\mathcal{X}_j} p^*(x_i,x_j)\log\frac{p^*(x_i,x_j)}{p^*(x_i)\,p^*(x_j)}\,dx_i dx_j$$

is the mutual information between the pair of variables X_i, X_j and

$$H(X_k) = -\int_{\mathcal{X}_k} p^*(x_k)\log p^*(x_k)\,dx_k$$

is the entropy. While the best forest will in fact be a spanning tree, the densities $p^*(x_i,x_j)$ are in practice not known. We estimate the marginals using finite data, in terms of a kernel density estimates $\widehat{p}_{n_1}(x_i,x_j)$ over a training set of size n_1. With these estimated marginals, we consider all forest density estimates of the form

$$\widehat{p}_F(x) = \prod_{(i,j)\in E(F)} \frac{\widehat{p}_{n_1}(x_i,x_j)}{\widehat{p}_{n_1}(x_i)\,\widehat{p}_{n_1}(x_j)} \prod_{k\in V_F} \widehat{p}_{n_1}(x_k).$$

Within this family, the best density estimate may not be supported on a full spanning tree, since a full tree will in general be subject to overfitting. Analogously, in high dimensional linear regression, the optimal regression model will generally be a full d-dimensional fit, with a nonzero parameter for each variable. However, when estimated on finite data the variance of a full model will dominate the squared bias, resulting in overfitting. In our setting of density estimation we will regulate the complexity of the forest by cross validating over a held out set.

There are several different ways to judge the quality of a forest structured density estimator. In this paper we concern ourselves with prediction and structure estimation.

Definition 3 ((Risk consistency)) *For an estimator $\widehat{q}_n \in \mathcal{P}_d$, the excess risk is defined as $R(\widehat{q}_n) - R^*$. The estimator $\widehat{q}_n$ is risk consistent with convergence rate δ_n if*

$$\lim_{M\to\infty} \limsup_{n\to\infty} \mathbb{P}\left(R(\widehat{q}_n) - R^* \geq M\delta_n\right) = 0.$$

In this case we write $R(\widehat{q}_n) - R^* = O_P(\delta_n)$.

Definition 4 ((Estimation consistency)) *An estimator $\widehat{q}_n \in \mathcal{P}_d$ is estimation consistent with convergence rate δ_n, with respect to the Kullback-Leibler divergence, if*

$$\lim_{M\to\infty} \limsup_{n\to\infty} \mathbb{P}\left(D(p^*_{F^*} \| \widehat{q}_n) \geq M\delta_n\right) = 0.$$

Definition 5 ((Structure selection consistency)) *An estimator $\widehat{q}_n \in \mathcal{P}_d$ supported by a forest $\widehat{F}_n$ is structure selection consistent if*

$$\mathbb{P}\left(E(\widehat{F}_n) \neq E(F^*)\right) \to 0,$$

as n goes to infinity, where F^* is defined in (4).

Later we will show that estimation consistency is almost equivalent to risk consistency. If the true density is given, these two criteria are exactly the same; otherwise, estimation consistency requires stronger conditions than risk consistency.

It is important to note that risk consistency is an oracle property, in the sense that the true density $p^*(x)$ is not restricted to be supported by a forest; rather, the property assesses how well a given estimator $\widehat{q}$ approximates the best forest density (the oracle) within a class.

3. Kernel Density Estimation For Forests

If the true density $p^*(x)$ were known, by Proposition 2, the density estimation problem would be reduced to finding the best forest structure F^*_d, satisfying

$$F^*_d = \arg\min_{F\in\mathcal{F}_d} R(p^*_F) = \arg\min_{F\in\mathcal{F}_d} D(p^* \| p^*_F).$$

The optimal forest F^*_d can be found by minimizing the right hand side of (5). Since the entropy term $H(X) = \sum_k H(X_k)$ is constant across all forests, this can be recast as the problem of finding the maximum weight spanning forest for a weighted graph, where the weight $w(i,j)$ of the edge connecting nodes i and j is $I(X_i;X_j)$. Kruskal's algorithm (Kruskal, 1956) is a greedy algorithm

that is guaranteed to find a maximum weight spanning tree of a weighted graph. In the setting of density estimation, this procedure was proposed by Chow. and Liu (1968) as a way of constructing a tree approximation to a distribution. At each stage the algorithm adds an edge connecting that pair of variables with maximum mutual information among all pairs not yet visited by the algorithm, if doing so does not form a cycle. When stopped early, after $k < d-1$ edges have been added, it yields the best k-edge weighted forest.

Of course, the above procedure is not practical since the true density $p^*(x)$ is unknown. We replace the population mutual information $I(X_i;X_j)$ in (5) by the plug-in estimate $\widehat{I}_n(X_i,X_j)$, defined as

$$\widehat{I}_n(X_i,X_j) = \int_{\mathcal{X}_i\times\mathcal{X}_j} \widehat{p}_n(x_i,x_j)\log\frac{\widehat{p}_n(x_i,x_j)}{\widehat{p}_n(x_i)\,\widehat{p}_n(x_j)}\,dx_i dx_j$$

where $\widehat{p}_n(x_i,x_j)$ and $\widehat{p}_n(x_i)$ are bivariate and univariate kernel density estimates. Given this estimated mutual information matrix $\widehat{M}_n = \left[\widehat{I}_n(X_i,X_j)\right]$, we can then apply Kruskal's algorithm (equivalently, the Chow-Liu algorithm) to find the best forest structure $\widehat{F}_n$.

Since the number of edges of $\widehat{F}_n$ controls the number of degrees of freedom in the final density estimator, we need an automatic data-dependent way to choose it. We adopt the following two-stage procedure. First, randomly partition the data into two sets $\mathcal{D}_1$ and $\mathcal{D}_2$ of sizes n_1 and n_2; then, apply the following steps:

1. Using $\mathcal{D}_1$, construct kernel density estimates of the univariate and bivariate marginals and calculate $\widehat{I}_{n_1}(X_i,X_j)$ for $i,j\in\{1,\ldots,d\}$ with $i\neq j$. Construct a full tree $\widehat{F}_{n_1}^{(d-1)}$ with $d-1$ edges, using the Chow-Liu algorithm.

2. Using $\mathcal{D}_2$, prune the tree $\widehat{F}_{n_1}^{(d-1)}$ to find a forest $\widehat{F}_{n_1}^{(\widehat{k})}$ with $\widehat{k}$ edges, for $0\le\widehat{k}\le d-1$.

Once $\widehat{F}_{n_1}^{(\widehat{k})}$ is obtained in Step 2, we can calculate $\widehat{p}_{\widehat{F}_{n_1}^{(\widehat{k})}}$ according to (1), using the kernel density estimates constructed in Step 1.

3.1 Step 1: Estimating the Marginals

Step 1 is carried out on the data set $\mathcal{D}_1$. Let $K(\cdot)$ be a univariate kernel function. Given an evaluation point (x_i,x_j), the bivariate kernel density estimate for (X_i,X_j) based on the observations $\{X_i^{(s)},X_j^{(s)}\}_{s\in\mathcal{D}_1}$ is defined as

$$\widehat{p}_{n_1}(x_i,x_j) = \frac{1}{n_1}\sum_{s\in\mathcal{D}_1}\frac{1}{h_2^2}K\left(\frac{X_i^{(s)}-x_i}{h_2}\right)K\left(\frac{X_j^{(s)}-x_j}{h_2}\right), \tag{6}$$

where we use a product kernel with $h_2>0$ be the bandwidth parameter. The univariate kernel density estimate $\widehat{p}_{n_1}(x_k)$ for X_k is

$$\widehat{p}_{n_1}(x_k) = \frac{1}{n_1}\sum_{s\in\mathcal{D}_1}\frac{1}{h_1}K\left(\frac{X_k^{(s)}-x_k}{h_1}\right), \tag{7}$$

Algorithm 1 Chow-Liu (Kruskal)

1: **Input** data $\mathcal{D}_1 = \{X^{(1)}, \ldots, X^{(n_1)}\}$.
2: Calculate $\widehat{M}_{n_1}$, according to (6), (7), and (8).
3: Initialize $E^{(0)} = \emptyset$
4: **for** $k = 1, \ldots, d-1$ **do**
5: $\quad (i^{(k)}, j^{(k)}) \leftarrow \arg\max_{(i,j)} \widehat{M}_{n_1}(i,j)$ such that $E^{(k-1)} \cup \{(i^{(k)}, j^{(k)})\}$ does not contain a cycle
6: $\quad E^{(k)} \leftarrow E^{(k-1)} \cup \{(i^{(k)}, j^{(k)})\}$
7: **Output** tree $\widehat{F}_{n_1}^{(d-1)}$ with edge set $E^{(d-1)}$.

where $h_1 > 0$ is the univariate bandwidth. Detailed specifications for $K(\cdot)$ and h_1, h_2 will be discussed in the next section.

We assume that the data lie in a d-dimensional unit cube $\mathcal{X} = [0,1]^d$. To calculate the empirical mutual information $\widehat{I}_{n_1}(X_i, X_j)$, we need to numerically evaluate a two-dimensional integral. To do so, we calculate the kernel density estimates on a grid of points. We choose m evaluation points on each dimension, $x_{1i} < x_{2i} < \cdots < x_{mi}$ for the ith variable. The mutual information $\widehat{I}_{n_1}(X_i, X_j)$ is then approximated as

$$\widehat{I}_{n_1}(X_i, X_j) = \frac{1}{m^2} \sum_{k=1}^{m} \sum_{\ell=1}^{m} \widehat{p}_{n_1}(x_{ki}, x_{\ell j}) \log \frac{\widehat{p}_{n_1}(x_{ki}, x_{\ell j})}{\widehat{p}_{n_1}(x_{ki})\, \widehat{p}_{n_1}(x_{\ell j})}. \tag{8}$$

The approximation error can be made arbitrarily small by choosing m sufficiently large. As a practical concern, care needs to be taken that the factors $\widehat{p}_{n_1}(x_{ki})$ and $\widehat{p}_{n_1}(x_{\ell j})$ in the denominator are not too small; a truncation procedure can be used to ensure this. Once the $d \times d$ mutual information matrix $\widehat{M}_{n_1} = \left[\widehat{I}_{n_1}(X_i, X_j)\right]$ is obtained, we can apply the Chow-Liu (Kruskal) algorithm to find a maximum weight spanning tree.

3.2 Step 2: Optimizing the Forest

The full tree $\widehat{F}_{n_1}^{(d-1)}$ obtained in Step 1 might have high variance when the dimension d is large, leading to overfitting in the density estimate. In order to reduce the variance, we prune the tree; that is, we choose forest with $k \leq d-1$ edges. The number of edges k is a tuning parameter that induces a bias-variance tradeoff.

In order to choose k, note that in stage k of the Chow-Liu algorithm we have an edge set $E^{(k)}$ (in the notation of the Algorithm 1) which corresponds to a forest $\widehat{F}_{n_1}^{(k)}$ with k edges, where $\widehat{F}_{n_1}^{(0)}$ is the union of d disconnected nodes. To select k, we choose among the d trees $\widehat{F}_{n_1}^{(0)}$, $\widehat{F}_{n_1}^{(1)}$, $\ldots$, $\widehat{F}_{n_1}^{(d-1)}$.

Let $\widehat{p}_{n_2}(x_i, x_j)$ and $\widehat{p}_{n_2}(x_k)$ be defined as in (6) and (7), but now evaluated solely based on the held-out data in $\mathcal{D}_2$. For a density p_F that is supported by a forest F, we define the held-out negative log-likelihood risk as

$$\begin{aligned} &\widehat{R}_{n_2}(p_F) \\ &\quad = \quad - \sum_{(i,j) \in E_F} \int_{\mathcal{X}_i \times \mathcal{X}_j} \widehat{p}_{n_2}(x_i, x_j) \log \frac{p(x_i, x_j)}{p(x_i)\, p(x_j)}\, dx_i dx_j - \sum_{k \in V_F} \int_{\mathcal{X}_k} \widehat{p}_{n_2}(x_k) \log p(x_k)\, dx_k. \end{aligned} \tag{9}$$

The selected forest is then $\widehat{F}_{n_1}^{(\widehat{k})}$ where

$$\widehat{k} = \operatorname*{arg\,min}_{k\in\{0,\ldots,d-1\}} \widehat{R}_{n_2}\left(\widehat{p}_{\widehat{F}_{n_1}^{(k)}}\right)$$

and where $\widehat{p}_{\widehat{F}_{n_1}^{(k)}}$ is computed using the density estimate $\widehat{p}_{n_1}$ constructed on $\mathcal{D}_1$.

For computational simplicity, we can also estimate $\widehat{k}$ as

$$\begin{aligned}\widehat{k} &= \operatorname*{arg\,max}_{k\in\{0,\ldots,d-1\}} \frac{1}{n_2}\sum_{s\in\mathcal{D}_2}\log\left(\prod_{(i,j)\in E^{(k)}}\frac{\widehat{p}_{n_1}(X_i^{(s)},X_j^{(s)})}{\widehat{p}_{n_1}(X_i^{(s)})\,\widehat{p}_{n_1}(X_j^{(s)})}\prod_{k\in V_{\widehat{F}_{n_1}^{(k)}}}\widehat{p}_{n_1}(X_k^{(s)})\right)\\ &= \operatorname*{arg\,max}_{k\in\{0,\ldots,d-1\}} \frac{1}{n_2}\sum_{s\in\mathcal{D}_2}\log\left(\prod_{(i,j)\in E^{(k)}}\frac{\widehat{p}_{n_1}(X_i^{(s)},X_j^{(s)})}{\widehat{p}_{n_1}(X_i^{(s)})\,\widehat{p}_{n_1}(X_j^{(s)})}\right).\end{aligned}$$

This minimization can be efficiently carried out by iterating over the $d-1$ edges in $\widehat{F}_{n_1}^{(d-1)}$.

Once $\widehat{k}$ is obtained, the final forest density estimate is given by

$$\widehat{p}_n(x) = \prod_{(i,j)\in E^{(\widehat{k})}}\frac{\widehat{p}_{n_1}(x_i,x_j)}{\widehat{p}_{n_1}(x_i)\,\widehat{p}_{n_1}(x_j)}\prod_k \widehat{p}_{n_1}(x_k).$$

Remark 6 *For computational efficiency, Step 1 can be carried out simultaneously with Step 2. In particular, during the Chow-Liu iteration, whenever an edge is added to $E^{(k)}$, the log-likelihood of the resulting density estimator on $\mathcal{D}_2$ can be immediately computed. A more efficient algorithm to speed up the computation of the mutual information matrix is discussed in Appendix B.*

3.3 Building a Forest on Held-out Data

Another approach to estimating the forest structure is to estimate the marginal densities on the training set, but only build graphs on the held-out data. To do so, we first estimate the univariate and bivariate kernel density estimates using $\mathcal{D}_1$, denoted by $\widehat{p}_{n_1}(x_i)$ and $\widehat{p}_{n_1}(x_i,x_j)$. We also construct a new set of univariate and bivariate kernel density estimates using $\mathcal{D}_2$, $\widehat{p}_{n_2}(x_i)$ and $\widehat{p}_{n_2}(x_i,x_j)$. We then estimate the "cross-entropies" of the kernel density estimates $\widehat{p}_{n_1}$ for each pair of variables by computing

$$\begin{aligned}\widehat{I}_{n_2,n_1}(X_i,X_j) &= \int \widehat{p}_{n_2}(x_i,x_j)\log\frac{\widehat{p}_{n_1}(x_i,x_j)}{\widehat{p}_{n_1}(x_i)\widehat{p}_{n_1}(x_j)}\,dx_i\,dx_j\\ &\approx \frac{1}{m^2}\sum_{k=1}^{m}\sum_{\ell=1}^{m}\widehat{p}_{n_2}(x_{ki},x_{\ell j})\log\frac{\widehat{p}_{n_1}(x_{ki},x_{\ell j})}{\widehat{p}_{n_1}(x_{ki})\,\widehat{p}_{n_1}(x_{\ell j})}.\end{aligned} \tag{10}$$

Our method is to use $\widehat{I}_{n_2,n_1}(X_i,X_j)$ as edge weights on a full graph and run Kruskal's algorithm until we encounter edges with negative weight. Let $\mathcal{F}$ be the set of all forests and $\widehat{w}_{n_2}(i,j)=\widehat{I}_{n_2,n_1}(X_i,X_j)$. The final forest is then

$$\widehat{F}_{n_2} = \operatorname*{arg\,max}_{F\in\mathcal{F}}\widehat{w}_{n_2}(F) = \operatorname*{arg\,min}_{F\in\mathcal{F}}\widehat{R}_{n_2}(\widehat{p}_F)$$

By building a forest on held-out data, we directly cross-validate over *all* forests.

4. Statistical Properties

In this section we present our theoretical results on risk consistency, structure selection consistency, and estimation consistency of the forest density estimate $\widehat{p}_n = \widehat{p}_{\widehat{F}_d^{(\widehat{k})}}$.

To establish some notation, we write $a_n = \Omega(b_n)$ if there exists a constant c such that $a_n \geq c b_n$ for sufficiently large n. We also write $a_n \asymp b_n$ if there exists a constant c such that $a_n \leq c\, b_n$ and $b_n \leq c\, a_n$ for sufficiently large n. Given a d-dimensional function f on the domain $\mathcal{X}$, we denote its $L_2(P)$-norm and sup-norm as

$$\|f\|_{L_2(P)} = \sqrt{\int_{\mathcal{X}} f^2(x) dP_X(x)}, \qquad \|f\|_\infty = \sup_{x \in \mathcal{X}} |f(x)|$$

where P_X is the probability measure induced by X. Throughout this section, all constants are treated as generic values, and as a result they can change from line to line.

In our use of a data splitting scheme, we always adopt equally sized splits for simplicity, so that $n_1 = n_2 = n/2$, noting that this does not affect the final rate of convergence.

4.1 Assumptions on the Density

Fix $\beta > 0$. For any d-tuple $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}^d$ and $x = (x_1, \ldots, x_d) \in \mathcal{X}$, we define $x^\alpha = \prod_{j=1}^d x_j^{\alpha_j}$. Let D^α denote the differential operator

$$D^\alpha = \frac{\partial^{\alpha_1 + \cdots + \alpha_d}}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}.$$

For any real-valued d-dimensional function f on $\mathcal{X}$ that is $\lfloor \beta \rfloor$-times continuously differentiable at point $x_0 \in \mathcal{X}$, let $P^{(\beta)}_{f,x_0}(x)$ be its Taylor polynomial of degree $\lfloor \beta \rfloor$ at point x_0:

$$P^{(\beta)}_{f,x_0}(x) = \sum_{\alpha_1 + \cdots + \alpha_d \leq \lfloor \beta \rfloor} \frac{(x - x_0)^\alpha}{\alpha_1! \cdots \alpha_d!} D^\alpha f(x_0).$$

Fix $L > 0$, and denote by $\Sigma(\beta, L, r, x_0)$ the set of functions $f : \mathcal{X} \to \mathbb{R}$ that are $\lfloor \beta \rfloor$-times continuously differentiable at x_0 and satisfy

$$\left| f(x) - P^{(\beta)}_{f,x_0}(x) \right| \leq L \|x - x_0\|_2^\beta, \ \ \forall x \in \mathcal{B}(x_0, r)$$

where $\mathcal{B}(x_0, r) = \{x : \|x - x_0\|_2 \leq r\}$ is the L_2-ball of radius r centered at x_0. The set $\Sigma(\beta, L, r, x_0)$ is called the (β, L, r, x_0)-locally Hölder class of functions. Given a set A, we define

$$\Sigma(\beta, L, r, A) = \cap_{x_0 \in A} \Sigma(\beta, L, r, x_0).$$

The following are the regularity assumptions we make on the true density function $p^*(x)$.

Assumption 1 *For any $1 \leq i < j \leq d$, we assume*

(D1) there exist $L_1 > 0$ and $L_2 > 0$ such that for any $c > 0$ the true bivariate and univariate densities satisfy

$$p^*(x_i, x_j) \in \Sigma\left(\beta, L_2, c\,(\log n / n)^{\frac{1}{2\beta+2}}, \mathcal{X}_i \times \mathcal{X}_j\right)$$

and

$$p^*(x_i) \in \Sigma\left(\beta, L_1, c\,(\log n / n)^{\frac{1}{2\beta+1}}, \mathcal{X}_i\right);$$

(D2) there exists two constants c_1 and c_2 such that

$$c_1 \leq \inf_{x_i,x_j \in \mathcal{X}_i \times \mathcal{X}_j} p^*(x_i,x_j) \leq \sup_{x_i,x_j \in \mathcal{X}_i \times \mathcal{X}_j} p^*(x_i,x_j) \leq c_2$$

μ-almost surely.

These assumptions are mild, in the sense that instead of adding constraints on the joint density $p^*(x)$, we only add regularity conditions on the bivariate and univariate marginals.

4.2 Assumptions on the Kernel

An important ingredient in our analysis is an exponential concentration result for the kernel density estimate, due to Giné and Guillou (2002). We first specify the requirements on the kernel function $K(\cdot)$.

Let $(\Omega, \mathcal{A})$ be a measurable space and let $\mathcal{F}$ be a uniformly bounded collection of measurable functions.

Definition 7 *$\mathcal{F}$ is a bounded measurable VC class of functions with characteristics A and v if it is separable and for every probability measure P on $(\Omega, \mathcal{A})$ and any $0 < \varepsilon < 1$,*

$$N\left(\varepsilon \|F\|_{L_2(P)}, \mathcal{F}, \|\cdot\|_{L_2(P)}\right) \leq \left(\frac{A}{\varepsilon}\right)^v,$$

where $F(x) = \sup_{f \in \mathcal{F}} |f(x)|$ and $N(\varepsilon, \mathcal{F}, \|\cdot\|_{L_2(P)})$ denotes the ε-covering number of the metric space $(\Omega, \|\cdot\|_{L_2(P)})$; that is, the smallest number of balls of radius no larger than ε (in the norm $\|\cdot\|_{L_2(P)}$) needed to cover $\mathcal{F}$.

The one-dimensional density estimates are constructed using a kernel K, and the two-dimensional estimates are constructed using the product kernel

$$K_2(x,y) = K(x) \cdot K(y).$$

Assumption 2 *The kernel K satisfies the following properties.*

(K1) $\int K(u)\,du = 1$, $\int_{-\infty}^{\infty} K^2(u)\,du < \infty$ *and* $\sup_{u \in \mathbb{R}} K(u) \leq c$ *for some constant c.*

(K2) K is a finite linear combination of functions g whose epigraphs $epi(g) = \{(s,u) : g(s) \geq u\}$, can be represented as a finite number of Boolean operations (union and intersection) among sets of the form $\{(s,u) : Q(s,u) \geq \phi(u)\}$, where Q is a polynomial on $\mathbb{R} \times \mathbb{R}$ and ϕ is an arbitrary real function.

(K3) K has a compact support and for any $\ell \geq 1$ and $1 \leq \ell' \leq \lfloor \beta \rfloor$

$$\int |t|^\beta |K(t)|\,dt < \infty, \text{ and } \int |K(t)|^\ell dt < \infty, \quad \int t^{\ell'} K(t) dt = 0.$$

Assumptions (K1), (K2) and (K3) are mild. As pointed out by Nolan and Pollard (1987), both the pyramid (truncated or not) kernel and the boxcar kernel satisfy them. It follows from (K2) that the classes of functions

$$\begin{aligned}\mathcal{F}_1 &= \left\{\frac{1}{h_1}K\left(\frac{u-\cdot}{h_1}\right) : u \in \mathbb{R},\, h_1 > 0\right\} \\ \mathcal{F}_2 &= \left\{\frac{1}{h_2^2}K\left(\frac{u-\cdot}{h_2}\right)K\left(\frac{t-\cdot}{h_2}\right) : u,t \in \mathbb{R},\, h_2 > 0\right\} \end{aligned} \tag{11}$$

are bounded VC classes, in the sense of Definition 7. Assumption (K3) essentially says that the kernel $K(\cdot)$ should be β-*valid*; see Tsybakov (2008) and Definition 6.1 in Rigollet and Vert (2009) for further details about this assumption. Kernels satisfying (K2) include finite linear combinations of functions of the form $\phi(p(x))$ where p is a polynomial and ϕ is a bounded function of bounded variation (Giné and Guillou, 2002; Nolan and Pollard, 1987). Therefore, the kernels constructed in terms of Legendre polynomials as in Riggolet and Vert (2009) and Tsybakov (2008), satisfy (K2) and (K3).

We choose the bandwidths h_1 and h_2 used in the one-dimensional and two-dimensional kernel density estimates to satisfy

$$h_1 \asymp \left(\frac{\log n}{n}\right)^{\frac{1}{1+2\beta}} \tag{12}$$

$$h_2 \asymp \left(\frac{\log n}{n}\right)^{\frac{1}{2+2\beta}}. \tag{13}$$

This choice of bandwidths ensures the optimal rate of convergence.

4.3 Risk Consistency

Given the above assumptions, we first present a key lemma that establishes the rates of convergence of bivariate and univariate kernel density estimates in the sup norm. The proof of this and our other technical results are provided in Appendix A.

Lemma 8 *Under Assumptions 1 and 2, and choosing bandwidths satisfying* (12) *and* (13)*, the bivariate and univariate kernel density estimates $\widehat{p}(x_i,x_j)$ and $\widehat{p}(x_k)$ in* (6) *and* (7) *satisfy*

$$\begin{aligned}\mathbb{P}\left(\max_{(i,j)\in\{1,\ldots,d\}\times\{1,\ldots,d\}}\sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j}|\widehat{p}(x_i,x_j)-p^*(x_i,x_j)| \geq \varepsilon\right) \\ \leq\ c_2 d^2 \exp\left(-c_3 n^{\frac{\beta}{1+\beta}}(\log n)^{\frac{1}{1+\beta}}\varepsilon^2\right)\end{aligned}$$

for $\varepsilon \geq 4c_4 h_2^\beta$. Hence, choosing

$$\varepsilon = \Omega\left(4c_4\sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}}\right)$$

we have that

$$\max_{(i,j)\in\{1,\dots,d\}\times\{1,\dots,d\}} \sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |\widehat{p}(x_i,x_j) - p^*(x_i,x_j)| = O_P\left(\sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}}\right). \tag{14}$$

Similarly,

$$\mathbb{P}\left(\max_{i\in\{1,\dots,d\}} \sup_{x_i\in\mathcal{X}_i} |\widehat{p}(x_i) - p^*(x_i)| \geq \varepsilon\right) \leq c_5 d \exp\left(-c_6 n^{\frac{2\beta}{1+2\beta}} (\log n)^{\frac{1}{1+2\beta}} \varepsilon^2\right)$$

and

$$\max_{k\in\{1,\dots,d\}} \sup_{x_k\in\mathcal{X}_k} |\widehat{p}(x_k) - p^*(x_k)| = O_P\left(\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right). \tag{15}$$

To describe the risk consistency result, let $\mathcal{P}_d^{(d-1)} = \mathcal{P}_d$ be the family of densities that are supported by forests with at most $d-1$ edges, as already defined in (2). For $0 \leq k \leq d-1$, we define $\mathcal{P}_d^{(k)}$ as the family of d-dimensional densities that are supported by forests with at most k edges. Then

$$\mathcal{P}_d^{(0)} \subset \mathcal{P}_d^{(1)} \subset \cdots \subset \mathcal{P}_d^{(d-1)}. \tag{16}$$

Now, due to the nesting property (16), we have

$$\inf_{q_F\in\mathcal{P}_d^{(0)}} R(q_F) \geq \inf_{q_F\in\mathcal{P}_d^{(1)}} R(q_F) \geq \cdots \geq \inf_{q_F\in\mathcal{P}_d^{(d-1)}} R(q_F).$$

We first analyze the forest density estimator obtained using a fixed number of edges $k < d$; specifically, consider stopping the Chow-Liu algorithm in Stage 1 after k iterations. This is in contrast to the algorithm described in 3.2, where the pruned tree size is automatically determined on the held out data. While this is not very realistic in applications, since the tuning parameter k is generally hard to choose, the analysis in this case is simpler, and can be directly exploited to analyze the more complicated data-dependent method.

Theorem 9 (Risk consistency) *Let $\widehat{p}_{\widehat{F}_d^{(k)}}$ be the forest density estimate with $|E(\widehat{F}_d^{(k)})| = k$, obtained after the first k iterations of the Chow-Liu algorithm, for some $k \in \{0,\dots,d-1\}$. Under Assumptions 1 and 2, we have*

$$R(\widehat{p}_{\widehat{F}_d^{(k)}}) - \inf_{q_F\in\mathcal{P}_d^{(k)}} R(q_F) = O_P\left(k\sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}} + d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right).$$

Note that this result allows the dimension d to increase at a rate $o\left(\sqrt{n^{2\beta/(1+2\beta)}/\log n}\right)$ and the number of edges k to increase at a rate $o\left(\sqrt{n^{\beta/(1+\beta)}/\log n}\right)$, with the excess risk still decreasing to zero asymptotically.

The above results can be used to prove a risk consistency result for the data-dependent pruning method using the data-splitting scheme described in Section 3.2.

Theorem 10 *Let $\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}$ be the forest density estimate using the data-dependent pruning method in Section 3.2, and let $\widehat{p}_{\widehat{F}_d^{(k)}}$ be the estimate with $|E(\widehat{F}_d^{(k)})| = k$ obtained after the first k iterations of the Chow-Liu algorithm. Under Assumptions 1 and 2, we have*

$$R(\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}) - \min_{0\le k\le d-1} R(\widehat{p}_{\widehat{F}_d^{(k)}}) = O_P\left((k^*+\widehat{k})\sqrt{\frac{\log n+\log d}{n^{\beta/(1+\beta)}}} + d\sqrt{\frac{\log n+\log d}{n^{2\beta/(1+2\beta)}}}\right)$$

where $k^* = \arg\min_{0\le k\le d-1} R(\widehat{p}_{\widehat{F}_d^{(k)}})$.

The proof of this theorem is given in the appendix. A parallel result can be obtained for the method described in Section 3.3, which builds the forest by running Kruskal's algorithm on the heldout data.

Theorem 11 *Let $\widehat{F}_{n_2}$ be the forest obtained using Kruskal's algorithm on held-out data, and let $\widehat{k} = |\widehat{F}_{n_2}|$ be the number of edges in $\widehat{F}_{n_2}$. Then*

$$R(\widehat{p}_{\widehat{F}_{n_2}}) - \min_{F\in\mathcal{F}} R(\widehat{p}_F) = O_P\left((k^*+\widehat{k})\sqrt{\frac{\log n+\log d}{n^{\beta/(1+\beta)}}} + d\sqrt{\frac{\log n+\log d}{n^{2\beta/(1+2\beta)}}}\right)$$

where $k^* = |F^*|$ is the number of edges in the optimal forest $F^* = \arg\min_{F\in\mathcal{F}} R(\widehat{p}_F)$.

4.4 Structure Selection Consistency

In this section, we provide conditions guaranteeing that the procedure is structure selection consistent. Again, we do not assume the true density $p^*(x)$ is consistent with a forest; rather, we are interested in comparing the estimated forest structure to the oracle forest which minimizes the risk. In this way our result differs from that in Tan et al. (2011), although there are similarities in the analysis.

By Proposition 2, we can define

$$p^*_{F_d^{(k)}} = \arg\min_{q_F\in\mathcal{P}_d^{(k)}} R(q_F).$$

Thus $F_d^{(k)}$ is the optimal forest within $\mathcal{P}_d^{(k)}$ that minimizes the negative log-likelihood loss. Let $\widehat{F}_d^{(k)}$ be the estimated forest structure, fixing the number of edges at k; we want to study conditions under which

$$\mathbb{P}\left(\widehat{F}_d^{(k)} = F_d^{(k)}\right) \to 1.$$

Let us first consider the population version of the algorithm—if the algorithm cannot recover the best forest $F_d^{(k)}$ in this ideal case, there is no hope for stable recovery in the data version. The key observation is that the graph selected by the Chow-Liu algorithm only depends on the relative order of the edges with respect to mutual information, not on the specific mutual information values. Let

$$\mathcal{E} = \left\{\{(i,j),(k,\ell)\} : i<j \text{ and } k<\ell,\ j\neq\ell \text{ and } i,j,k,\ell\in\{1,\dots,d\}\right\}.$$

The cardinality of $\mathcal{E}$ is

$$|\mathcal{E}| = O(d^4).$$

Let $e = (i, j)$ be an edge; the corresponding mutual information associated with e is denoted as I_e. If for all $(e, e') \in \mathcal{E}$, we have $I_e \neq I_{e'}$, the population version of the Chow-Liu algorithm will always obtain the unique solution $F_d^{(k)}$. However, this condition is, in a sense, both too weak and too strong. It is too weak because the sample estimates of the mutual information values will only approximate the population values, and could change the relative ordering of some edges. However, the assumption is too strong because, in fact, the relative order of many edge pairs might be changed without affecting the graph selected by the algorithm. For instance, when $k \geq 2$ and I_e and $I_{e'}$ are the largest two mutual information values, it is guaranteed that e and e' will both be included in the learned forest $F_d^{(k)}$ whether $I_e > I_{e'}$ or $I_e < I_{e'}$.

Define the *crucial set* $\mathcal{J} \subset \mathcal{E}$ to be a set of pairs of edges (e, e') such that $I_e \neq I_{e'}$ and flipping the relative order of I_e and $I_{e'}$ changes the learned forest structure in the population Chow-Liu algorithm, with positive probability. Here, we assume that the Chow-Liu algorithm randomly selects an edge when a tie occurs.

The cardinality $|\mathcal{J}|$ of the crucial set is a function of the true density $p^*(x)$, and we can expect $|\mathcal{J}| \ll |\mathcal{E}|$. The next assumption provides a sufficient condition for the two-stage procedure to be structure selection consistent.

Assumption 3 *Let the crucial set $\mathcal{J}$ be defined as before. Suppose that*

$$\min_{((i,j),(k,\ell))\in\mathcal{J}} |I(X_i;X_j) - I(X_k;X_\ell)| \geq 2L_n$$

where $L_n = \Omega\left(\sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}}\right)$.

This assumption is strong, but is satisfied in many cases. For example, in a graph with population mutual informations differing by a constant, the assumption holds. Assumption 3 is trivially satisfied if $\frac{n^{\beta/(1+\beta)}}{\log n + \log d} \to \infty$.

Theorem 12 (Structure selection consistency) *Let $F_d^{(k)}$ be the optimal forest within $\mathcal{P}_d^{(k)}$ that minimizes the negative log-likelihood loss. Let $\widehat{F}_d^{(k)}$ be the estimated forest with $|E_{\widehat{F}_d^{(k)}}| = k$. Under Assumptions 1, 2, and 3, we have*

$$\mathbb{P}\left(\widehat{F}_d^{(k)} = F_d^{(k)}\right) \to 1$$

as $n \to \infty$.

The proof shows that our method is structure selection consistent as long as the dimension increases as $d = o\left(\exp(n^{\beta/(1+\beta)})\right)$; in this case the error decreases at the rate

$$o\left(\exp\left(4\log d - c(\log n)^{\frac{1}{1+\beta}}\log d\right)\right).$$

4.5 Estimation Consistency

Estimation consistency can be easily established using the structure selection consistency result above. Define the event $\mathcal{M}_k = \{\widehat{F}_d^{(k)} = F_d^{(k)}\}$. Theorem 12 shows that $\mathbb{P}(\mathcal{M}_k^c) \to 0$ as n goes to infinity.

Lemma 13 *Let $\widehat{p}_{\widehat{F}_d^{(k)}}$ be the forest-based kernel density estimate for some fixed $k \in \{0,\ldots,d-1\}$, and let*

$$p^*_{F_d^{(k)}} = \arg\min_{q_F \in \mathcal{P}_d^{(k)}} R(q_F).$$

Under the assumptions of Theorem 12,

$$D(p^*_{F_d^{(k)}} \| \widehat{p}_{\widehat{F}_d^{(k)}}) = R(\widehat{p}_{\widehat{F}_d^{(k)}}) - R(p^*_{F_d^{(k)}})$$

on the event $\mathcal{M}_k$.

Proof According to Bach and Jordan (2003), for a given forest F and a target distribution $p^*(x)$,

$$D(p^* \| q_F) = D(p^* \| p^*_F) + D(p^*_F \| q_F) \tag{17}$$

for all distributions q_F that are supported by F. We further have

$$D(p^* \| q) = \int_{\mathcal{X}} p^*(x) \log p^*(x) - \int_{\mathcal{X}} p^*(x) \log q(x) dx = \int_{\mathcal{X}} p^*(x) \log p^*(x) dx + R(q) \tag{18}$$

for any distribution q. Using (17) and (18), and conditioning on the event $\mathcal{M}_k$, we have

$$\begin{aligned} D(p^*_{F_d^{(k)}} \| \widehat{p}_{\widehat{F}_d^{(k)}}) &= D(p^* \| \widehat{p}_{\widehat{F}_d^{(k)}}) - D(p^* \| p^*_{F_d^{(k)}}) \\ &= \int_{\mathcal{X}} p^*(x) \log p^*(x) dx + R(\widehat{p}_{\widehat{F}_d^{(k)}}) - \int_{\mathcal{X}} p^*(x) \log p^*(x) dx - R(p^*_{F_d^{(k)}}) \\ &= R(\widehat{p}_{\widehat{F}_d^{(k)}}) - R(p^*_{F_d^{(k)}}), \end{aligned}$$

which gives the desired result. ■

The above lemma combined with Theorem 9 allows us to obtain the following estimation consistency result, the proof of which is omitted.

Corollary 14 (Estimation consistency) *Under Assumptions 1, 2, and 3, we have*

$$D(p^*_{F_d^{(k)}} \| \widehat{p}_{\widehat{F}_d^{(k)}}) = O_P\left(k \sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}} + d \sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}} \right)$$

conditioned on the event $\mathcal{M}_k$.

5. Tree Restricted Forests

We now turn to the problem of estimating forests with restricted tree sizes. As discussed in the introduction, clustering problems motivate the goal of constructing forest structured density estimators where each connected component has a restricted number of edges. But estimating restricted tree size forests can also be useful in model selection for the purpose of risk minimization, since the maximum subtree size can be viewed as an additional complexity parameter.

Algorithm 2 Approximate Max Weight t-Restricted Forest

1: **Input** graph G with positive edge weights, and positive integer $t \geq 2$.

2: Sort edges in decreasing order of weight.

3: Greedily add edges in decreasing order of weight such that

   (a) the degree of any node is at most $t+1$;

   (b) no cycles are formed.

   The resulting forest is $F' = \{T_1, T_2, \ldots, T_m\}$.

4: **Output** $F_t = \cup_j \texttt{TreePartition}(T_j, t)$.

Definition 15 *A t-restricted forest of a graph G is a subgraph F_t such that*

1. *F_t is the disjoint union of connected components $\{T_1, ..., T_m\}$, each of which is a tree;*
2. *$|T_i| \leq t$ for each $i \leq m$, where $|T_i|$ denotes the number of edges in the ith component.*

Given a weight w_e assigned to each edge of G, an optimal t-restricted forest F_t^* satisfies

$$w(F_t^*) = \max_{F \in \mathcal{F}_t(G)} w(F)$$

where $w(F) = \sum_{e \in F} w_e$ is the weight of a forest F and $\mathcal{F}_t(G)$ denotes the collection of all t-restricted forests of G.

For $t = 1$, the problem is maximum weighted matching. However, for $t \geq 7$, we show that finding an optimal t-restricted forest is an NP-hard problem; however, this problem appears not to have been previously studied. Our reduction is from Exact 3-Cover (X3C), shown to be NP-complete by Garey and Johnson 1979). In X3C, we are given a set X, a family $\mathcal{S}$ of 3-element subsets of X, and we must choose a subfamily of disjoint 3-element subsets to cover X. Our reduction constructs a graph with special tree-shaped subgraphs called *gadgets*, such that each gadget corresponds to a 3-element subset in $\mathcal{S}$. We show that finding a maximum weight t-restricted forest on this graph would allow us to then recover a solution to X3C by analyzing how the optimal forest must partition each of the gadgets.

Given the NP-hardness for finding optimal t-restricted forest, it is of interest to study approximation algorithms for the problem. Our first algorithm is Algorithm 2, which runs in two stages. In the first stage, a forest is greedily constructed in such a way that each node has degree no larger than t (a property that is satisfied by all t-restricted forests). However, the trees in the forest may have more than t edges; hence, in the second stage, each tree in the forest is partitioned in an optimal way by removing edges, resulting in a collection of trees, each of which has size at most t. The second stage employs a procedure we call `TreePartition` that takes a tree and returns the optimal t-restricted subforest. `TreePartition` is a divide-and-conquer procedure of Lukes (1974) that finds a carefully chosen set of forest partitions for each child subtree. It then merges these sets with the parent node one subtree at a time. The details of the `TreePartition` procedure are given in Appendix A.

Theorem 16 *Let F_t be the output of Algorithm 2, and let F_t^* be the optimal t-restricted forest. Then* $w(F_t) \geq \frac{1}{4} w(F_t^*)$.

In Appendix A.7, we present a proof of the above result. In that section, we also present an improved approximation algorithm, one based on solving linear programs, that finds a t-restricted forest F_t' such that $w(F_t') \geq \frac{1}{2} w(F_t^*)$. Although we cannot guarantee optimality in theory, algorithm 2 performs very well in practice. In Figure 1, we can see that the approximation picks out a t-restricted forest that is close to optimal among the set of all t-restricted forests.

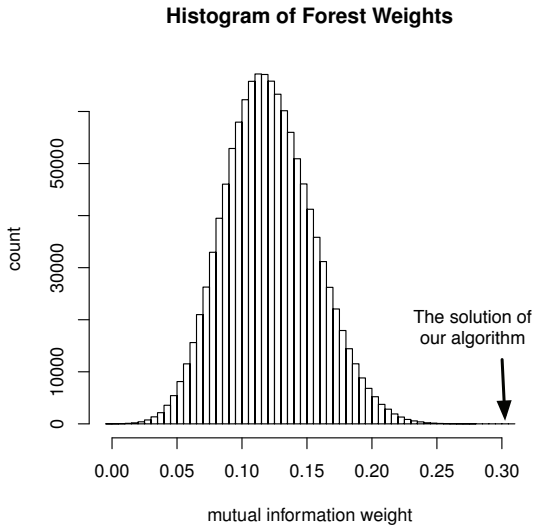


Figure 1: Histogram distribution of weights of all t-restricted forests on 11 nodes with $t = 7$. Edge weights are the mutual informations computed on the training data.

5.1 Pruning Based on t-Restricted Forests

For a given t, after producing an approximate maximum weight t-restricted forest $\widehat{F}_t$ using $\mathcal{D}_1$, we prune away edges using $\mathcal{D}_2$. To do so, we first construct a new set of univariate and bivariate kernel density estimates using $\mathcal{D}_2$, as before, $\widehat{p}_{n_2}(x_i)$ and $\widehat{p}_{n_2}(x_i, x_j)$. Recall that we define the "cross-entropies" of the kernel density estimates $\widehat{p}_{n_1}$ for each pair of variables as

$$\begin{aligned} \widehat{I}_{n_2,n_1}(X_i, X_j) &= \int \widehat{p}_{n_2}(x_i, x_j) \log \frac{\widehat{p}_{n_1}(x_i, x_j)}{\widehat{p}_{n_1}(x_i)\widehat{p}_{n_1}(x_j)} \, dx_i \, dx_j \\ &\approx \frac{1}{m^2} \sum_{k=1}^{m} \sum_{\ell=1}^{m} \widehat{p}_{n_2}(x_{ki}, x_{\ell j}) \log \frac{\widehat{p}_{n_1}(x_{ki}, x_{\ell j})}{\widehat{p}_{n_1}(x_{ki})\, \widehat{p}_{n_1}(x_{\ell j})}. \end{aligned}$$

We then eliminate all edges (i, j) in $\widehat{F}_t$ for which $\widehat{I}_{n_2,n_1}(X_i, X_j) \leq 0$. For notational simplicity, we denote the resulting pruned forest again by $\widehat{F}_t$.

Algorithm 3 t-Restricted Forest Density Estimation

1: Divide data into two halves $\mathcal{D}_1$ and $\mathcal{D}_2$.

2: Compute kernel density estimators $\widehat{p}_{n_1}$ and $\widehat{p}_{n2}$ for all pairs and single variable marginals.

3: For all pairs (i, j) compute $\widehat{I}_{n_1}(X_i, X_j)$ according to (8) and $\widehat{I}_{n_2,n_1}(X_i, X_j)$ according to (10).

4: For $t = 0, \ldots, t_{\text{final}}$ where t_{final} is chosen based on the application

1. Compute or approximate (for $t \geq 2$) the optimal t-restricted forest $\widehat{F}_t$ using $\widehat{I}_{n_1}$ as edge weights.

2. Prune $\widehat{F}_t$ to eliminate all edges with negative weights $\widehat{I}_{n_2,n_1}$.

5: Among all pruned forests $\widehat{p}_{F^t}$, select $\widehat{t} = \arg\min_{0 \leq t \leq t_{\text{final}}} \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_t})$.

To estimate the risk, we simply use $\widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_t})$ as defined in (9), and select the forest $\widehat{F}_{\widehat{t}}$ according to

$$\widehat{t} = \underset{0 \leq t \leq d-1}{\arg\min}\, \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_t}).$$

The resulting procedure is summarized in Algorithm 3.

Using the approximation guarantee and our previous analysis, we have that the population weights of the approximate t-restricted forest and the optimal forest satisfy the following inequality. We state the result for a general c-approximation algorithm; for the algorithm given above, $c = 4$, but tighter approximations are possible.

Theorem 17 *Assume the conditions of Theorem 9. For $t \geq 2$, let $\widehat{F}_t$ be the forest constructed using a c-approximation algorithm, and let F_t^* be the optimal forest; both constructed with respect to finite sample edge weights $\widehat{w}_{n_1} = \widehat{I}_{n_1}$. Then*

$$w(\widehat{F}_t) \geq \frac{1}{c} w(F_t^*) + O_P\left((k^* + \widehat{k}) \sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}} \right)$$

where $\widehat{k}$ and k^* are the number of edges in $\widehat{F}_t$ and F_t^*, respectively, and w denotes the population weights, given by the mutual information.

As seen below, although the approximation algorithm has weaker theoretical guarantees, it outperforms other approaches in experiments.

6. Experimental Results

In this section, we report numerical results on both synthetic data sets and microarray data. We mainly compare the forest density estimator with sparse Gaussian graphical models, fitting a multivariate Gaussian with a sparse inverse covariance matrix. The sparse Gaussian models are estimated using the graphical lasso algorithm (glasso) of Friedman et al. (2007), which is a refined version of an algorithm first derived by Banerjee et al. (2008). Since the glasso typically results in a large parameter bias as a consequence of the ℓ_1 regularization, we also compare with a method that we call

the *refit glasso*, which is a two-step procedure—in the first step, a sparse inverse covariance matrix is obtained by the glasso; in the second step, a Gaussian model is refit without ℓ_1 regularization, but enforcing the sparsity pattern obtained in the first step.

To quantitatively compare the performance of these estimators, we calculate the log-likelihood of all methods on a held-out data set $\mathcal{D}_2$. With $\widehat{\mu}_{n_1}$ and $\widehat{\Omega}_{n_1}$ denoting the estimates from the Gaussian model, the held-out log-likelihood can be explicitly evaluated as

$$\ell_{\text{gauss}} = -\frac{1}{n_2}\sum_{s\in\mathcal{D}_2}\left\{\frac{1}{2}(X^{(s)}-\widehat{\mu}_{n_1})^F\widehat{\Omega}_{n_1}(X^{(s)}-\widehat{\mu}_{n_1})+\frac{1}{2}\log\left(\frac{|\widehat{\Omega}_{n_1}|}{(2\pi)^d}\right)\right\}.$$

For a given tree structure $\widehat{F}$, the held-out log-likelihood for the forest density estimator is

$$\ell_{\text{fde}} = \frac{1}{n_2}\sum_{s\in\mathcal{D}_2}\log\left(\prod_{(i,j)\in E(\widehat{F})}\frac{\widehat{p}_{n_1}(X_i^{(s)},X_j^{(s)})}{\widehat{p}_{n_1}(X_i^{(s)})\widehat{p}_{n_1}(X_j^{(s)})}\prod_{k\in V_{\widehat{F}}}\widehat{p}_{n_1}(X_k^{(s)})\right),$$

where $\widehat{p}_{n_1}(\cdot)$ are the corresponding kernel density estimates, using a Gaussian kernel with plug-in bandwidths.

Since the held-out log-likelihood of the forest density estimator is indexed by the number of edges included in the tree, while the held-out log-likelihoods of the glasso and the refit glasso are indexed by a continuously varying regularization parameter, we need to find a way to calibrate them. To address this issue, we plot the held-out log-likelihood of the forest density estimator as a step function indexed by the tree size. We then run the full path of the glasso and discretize it according to the corresponding sparsity level, that is, how many edges are selected for each value of the regularization parameter. The size of the forest density estimator and the sparsity level of the glasso (and the refit glasso) can then be aligned for a fair comparison.

6.1 Synthetic Data

We use a procedure to generate high dimensional Gaussian and non-Gaussian data which are consistent with an undirected graph. We generate high dimensional graphs that contain cycles, and so are not forests. In dimension $d = 100$, we sample $n_1 = n_2 = 400$ data points from a multivariate Gaussian distribution with mean vector $\mu = (0.5,\ldots,0.5)$ and inverse covariance matrix Ω. The diagonal elements of Ω are all 62. We then randomly generate many connected subgraphs containing no more than eight nodes each, and set the corresponding non-diagonal elements in Ω at random, drawing values uniformly from -30 to -10. To obtain non-Gaussian data, we simply transform each dimension of the data by its empirical distribution function; such a transformation preserves the graph structure but the joint distribution is no longer Gaussian (see Liu et al., 2009).

To calculate the pairwise mutual information $\widehat{I}(X_i;X_j)$, we need to numerically evaluate two-dimensional integrals. We first rescale the data into $[0,1]^d$ and calculate the kernel density estimates on a grid of points; we choose $m = 128$ evaluation points $x_i^{(1)} < x_i^{(2)} < \cdots < x_i^{(m)}$ for each dimension i, and then evaluate the bivariate and the univariate kernel density estimates on this grid.

There are three different kernel density estimates that we use—the bivariate kde, the univariate kde, and the marginalized bivariate kde. Specifically, the bivariate kernel density estimate on x_i, x_j

based on the observations $\{X_i^{(s)}, X_j^{(s)}\}_{s\in\mathcal{D}_1}$ is defined as

$$\widehat{p}(x_i,x_j) = \frac{1}{n_1}\sum_{s\in\mathcal{D}_1}\frac{1}{h_{2i}h_{2j}}K\left(\frac{X_i^{(s)}-x_i}{h_{2i}}\right)K\left(\frac{X_j^{(s)}-x_j}{h_{2j}}\right),$$

using a product kernel. The bandwidths h_{2i}, h_{2j} are chosen as

$$h_{2k} = 1.06\cdot\min\left\{\widehat{\sigma}_k, \frac{\widehat{q}_{k,0.75}-\widehat{q}_{k,0.25}}{1.34}\right\}\cdot n^{-1/(2\beta+2)},$$

where $\widehat{\sigma}_k$ is the sample standard deviation of $\{X_k^{(s)}\}_{s\in\mathcal{D}_1}$ and $\widehat{q}_{k,0.75}$, $\widehat{q}_{k,0.25}$ are the 75% and 25% sample quantiles of $\{X_k^{(s)}\}_{s\in\mathcal{D}_1}$.

In all the experiments, we set $\beta = 2$, such a choice of β and the "plug-in" bandwidth h_{2k} (and h_{1k} in the following) is a very common practice in nonparametric Statistics. For more details, see Fan and Gijbels (1996) and Tsybakov (2008).

Given an evaluation point x_k, the univariate kernel density estimate $\widehat{p}(x_k)$ based on the observations $\{X_k^{(s)}\}_{s\in\mathcal{D}_1}$ is defined as

$$\widehat{p}(x_k) = \frac{1}{n_1}\sum_{s\in\mathcal{D}_1}\frac{1}{h_{1k}}K\left(\frac{X_k^{(s)}-x_k}{h_{1k}}\right),$$

where $h_{1k} > 0$ is defined as

$$h_{1k} = 1.06\cdot\min\left\{\widehat{\sigma}_k, \frac{\widehat{q}_{k,0.75}-\widehat{q}_{k,0.25}}{1.34}\right\}\cdot n^{-1/(2\beta+1)}.$$

Finally, the marginal univariate kernel density estimate $\widehat{p}_M(x_k)$ based on the observations $\{X_k^{(s)}\}_{s\in\mathcal{D}_1}$ is defined by integrating the irrelevant dimension out of the bivariate kernel density estimates $\widehat{p}(x_j,x_k)$ on the unit square $[0,1]^2$. Thus,

$$\widehat{p}_M(x_k) = \frac{1}{m-1}\sum_{\ell=1}^{m}\widehat{p}(x_j^{(\ell)},x_k).$$

With the above definitions of the bivariate and univariate kernel density estimates, we consider estimating the mutual information $I(X_i;X_j)$ in three different ways, depending on which estimates for the univariate densities are employed.

$$\begin{aligned}
\widehat{I}_{\text{fast}}(X_i,X_j) &= \frac{1}{(m-1)^2}\sum_{k'=1}^{m}\sum_{\ell'=1}^{m}\widehat{p}(x_i^{(k')},x_j^{(\ell')})\log\widehat{p}(x_i^{(k')},x_j^{(\ell')}) \\
&\quad \frac{1}{m-1}\sum_{k'=1}^{m}\widehat{p}(x_i^{(k')})\log\widehat{p}(x_i^{(k')}) - \frac{1}{m-1}\sum_{\ell'=1}^{m}\widehat{p}(x_j^{(\ell')})\log\widehat{p}(x_j^{(\ell')}) \\
\widehat{I}_{\text{medium}}(X_i,X_j) &= \frac{1}{(m-1)^2}\sum_{k'=1}^{m}\sum_{\ell'=1}^{m}\widehat{p}(x_i^{(k')},x_j^{(\ell')})\log\frac{\widehat{p}(x_i^{(k')},x_j^{(\ell')})}{\widehat{p}(x_i^{(k')})\,\widehat{p}(x_j^{(\ell')})}. \\
\widehat{I}_{\text{slow}}(X_i,X_j) &= \frac{1}{(m-1)^2}\sum_{k'=1}^{m}\sum_{\ell'=1}^{m}\widehat{p}(x_i^{(k')},x_j^{(\ell')})\log\widehat{p}(x_i^{(k')},x_j^{(\ell')}) - \\
&\quad \frac{1}{m-1}\sum_{k'=1}^{m}\widehat{p}_M(x_i^{(k')})\log\widehat{p}_M(x_i^{(k')}) - \frac{1}{m-1}\sum_{\ell'=1}^{m}\widehat{p}_M(x_j^{(\ell')})\log\widehat{p}_M(x_j^{(\ell')}).
\end{aligned}$$

The terms "fast," "medium" and "slow" refer to the theoretical statistical rates of convergence of the estimators. The "fast" estimate uses one-dimensional univariate kernel density estimators wherever possible. The "medium" estimate uses the one-dimensional kernel density estimates in the denominator of $p(x_i,x_j)/(p(x_i)p(x_j)$, but averages with respect to the bivariate density. Finally, the "slow" estimate marginalizes the bivariate densities to estimate the univariate densities. While the rate of convergence is the two-dimensional rate, the "slow" estimate ensures the consistency of the bivariate and univariate densities.

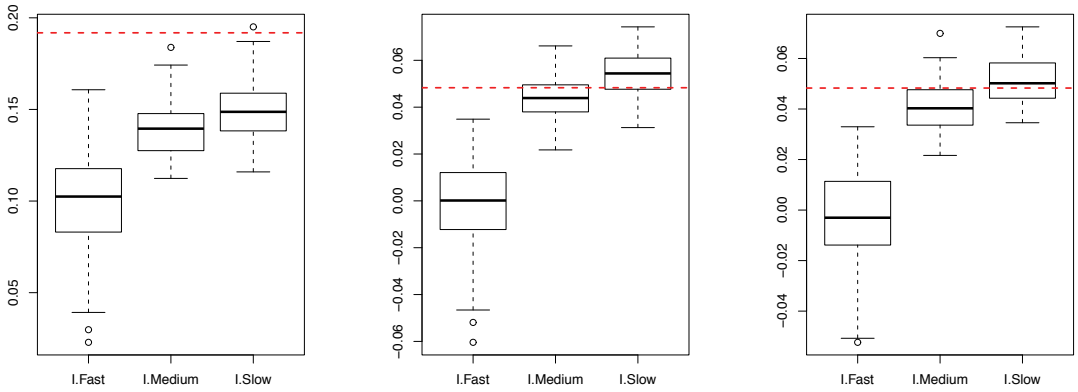


Figure 2: (Gaussian example) Boxplots of $\widehat{I}_{\text{fast}}$, $\widehat{I}_{\text{medium}}$, and $\widehat{I}_{\text{slow}}$ on three different pairs of variables. The red-dashed horizontal lines represent the population values.

Figure 2 compares $\widehat{I}_{\text{fast}}$, $\widehat{I}_{\text{medium}}$, and $\widehat{I}_{\text{slow}}$ on different pairs of variables. The boxplots are based on 100 trials. Compared to the ground truth, which can be computed exactly in the Gaussian case, we see that the performance of $\widehat{I}_{\text{medium}}$ and $\widehat{I}_{\text{slow}}$ is better than that of $\widehat{I}_{\text{fast}}$. This is due to the fact that simply replacing the population density with a "plug-in" version can lead to biased estimates; in fact, $\widehat{I}_{\text{fast}}$ is not even guaranteed to be non-negative. In what follows, we employ $\widehat{I}_{\text{medium}}$ for all the calculations, due to its ease of computation and good finite sample performance. Figure 3 compares the bivariate fits of the kernel density estimates and the Gaussian models over four edges. For the Gaussian fits of each edge, we directly calculate the bivariate sample covariance and sample mean and plug them into the bivariate Gaussian density function. From the perspective and contour plots, we see that the bivariate kernel density estimates provide reasonable fits for these bivariate components.

A typical run showing the held-out log-likelihood and estimated graphs is provided in Figure 4. We see that for the Gaussian data, the refit glasso has a higher held-out log-likelihood than the forest density estimator and the glasso. This is expected, since the Gaussian model is correct. For very sparse models, however, the performance of the glasso is worse than that of the forest density estimator, due to the large parameter bias resulting from the ℓ_1 regularization. We also observe an efficiency loss in the nonparametric forest density estimator, compared to the refit glasso. The graphs are automatically selected using the held-out log-likelihood, and we see that the nonparametric forest-based kernel density estimator tends to select a sparser model, while the parametric

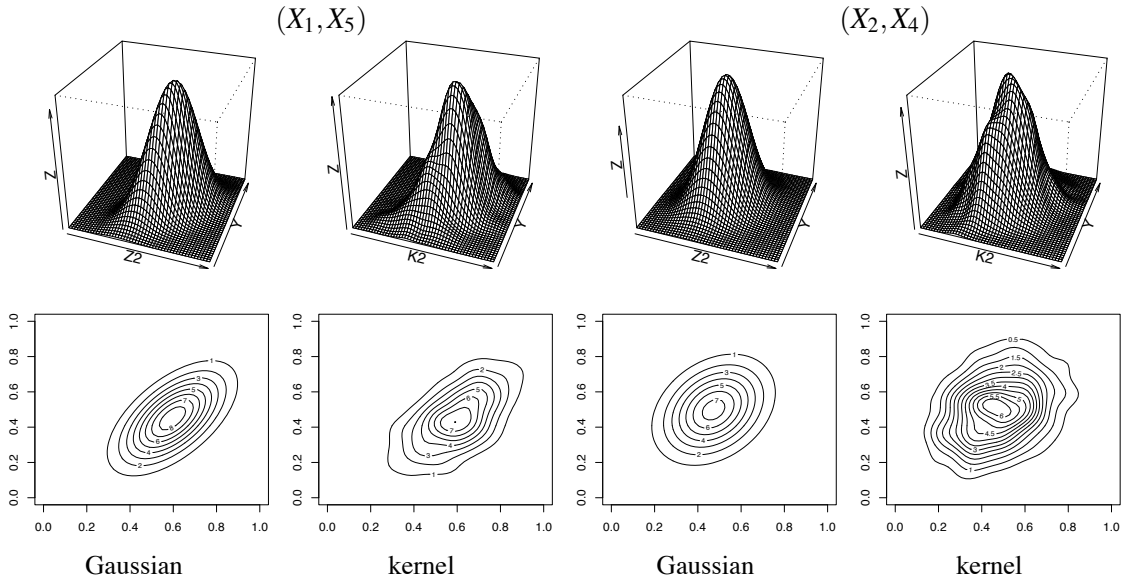


Figure 3: Perspective and contour plots of the bivariate Gaussian fits vs. the kernel density estimates for two edges of a Gaussian graphical model.

Gaussian models tend to overselect. This observation is new and is quite typical in our simulations. Another observation is that the held-out log-likelihood curve of the glasso becomes flat for less sparse models but never goes down. This suggests that the held-out log-likelihood is not a good model selection criterion for the glasso. For the non-Gaussian data, even though the refit glasso results in a reasonable graph, the forest density estimator performs much better in terms of held-out log-likelihood risk and graph estimation accuracy.

To compare with t-restricted forests, we generated additional Gaussian and non-Gaussian synthetic data as before except on a different graph structure. In Figure 5, we use 400 training examples while varying the size of heldout data to compare the log-likelihoods of four different methods; the log-likelihood is evaluated on a third large data set. In Figure 6, we consider only non-Gaussian data, use 400 training data and 400 heldout data, and generate graphs with best heldout log-likelihood across the four methods. We compute bandwidth, heldout log-likelihood, and mutual information same as before.

We observe that although creating a maximum spanning tree (MST) on the held-out data is asymptotically optimal; it can perform quite poorly. Unless there are copious amount of heldout data, held-out MST overfits on the heldout data and tend to give large graphs; in contrast, t-restricted forest has the weakest theoretical guarantee but it gives the best log-likelihood and produces sparser graphs. It is not surprising to note that MST on heldout data improves as heldout data size increases. Somewhat surprisingly though, Training-MST-with-pruning and t-restricted forest appear to be insensitive to the heldout data size.

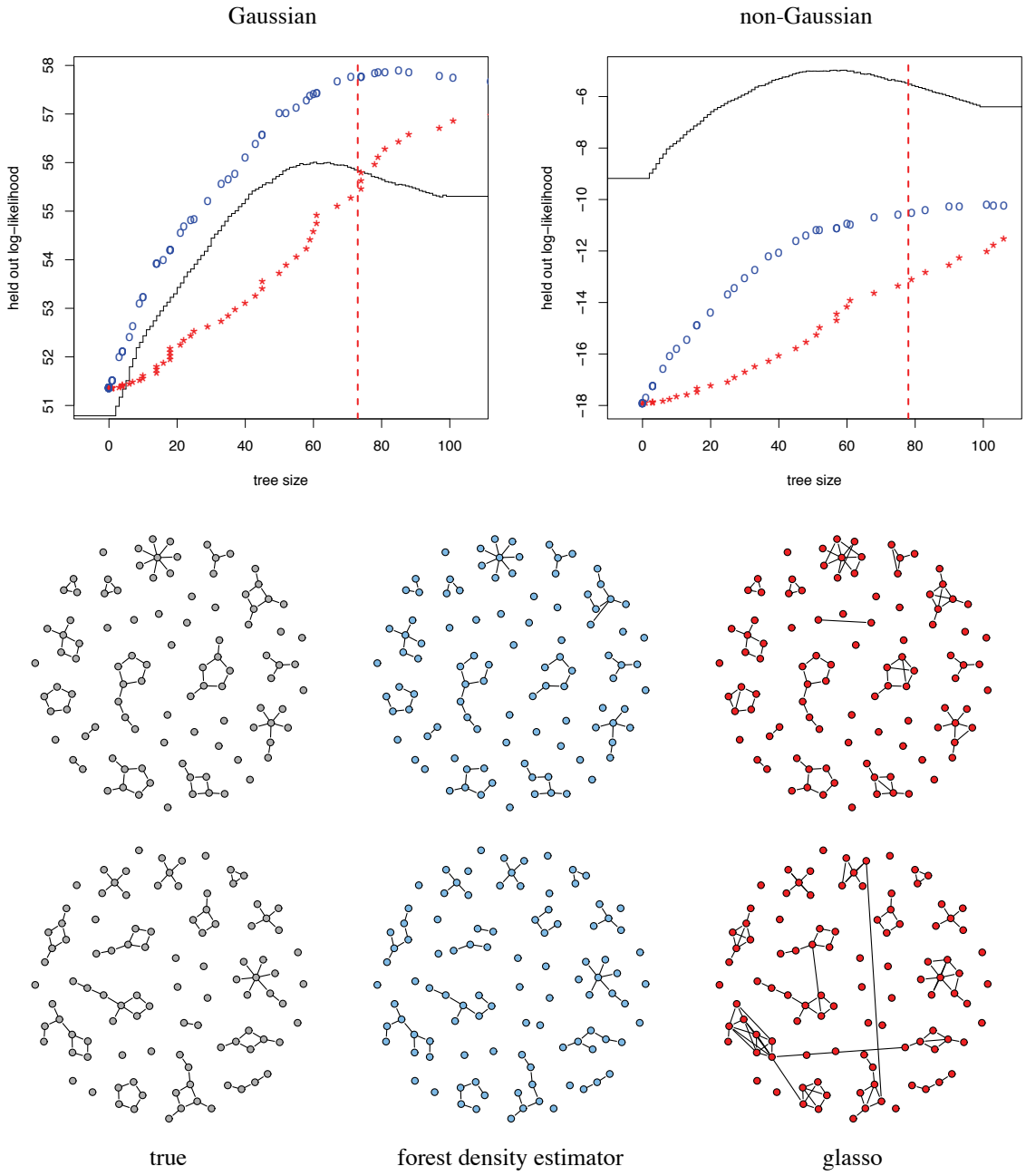
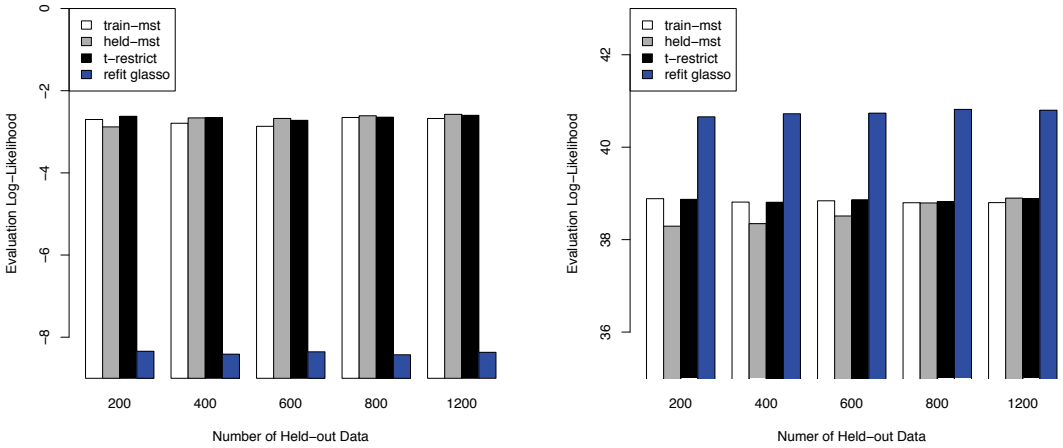


Figure 4: Synthetic data. Top-left Gaussian, and top-right non-Gaussian: Held-out log-likelihood plots of the forest density estimator (black step function), glasso (red stars), and refit glasso (blue circles), the vertical dashed red line indicates the size of the true graph. Bottom plots show the true and estimated graphs for the Gaussian (second row) and non-Gaussian data (third row).



(a) Non-Gaussian Data (b) Gaussian Data

Figure 5: Log-likelihood comparison of various methods: (left white) MST on Training Data with Pruning (gray) MST on Heldout Data (black) t-Restricted Graph (blue) Refit Glasso

6.2 Microarray Data

In this example, we study the empirical performance of the algorithms on a microarry dataset.

6.2.1 ARABIDOPSIS THALIANA DATA

We consider a data set based on Affymetrix GeneChip microarrays for the plant *Arabidopsis thaliana*, (Wille et al., 2004). The sample size is $n = 118$. The expression levels for each chip are pre-processed by a log-transformation and standardization. A subset of 40 genes from the isoprenoid pathway are chosen, and we study the associations among them using the glasso, the refit glasso, and the forest-based kernel density estimator.

From the held-out log-likelihood curves in Figure 7, we see that the tree-based kernel density estimator has a better generalization performance than the glasso and the refit glasso. This is not surprising, given that the true distribution of the data is not Gaussian. Another observation is that for the tree-based kernel density estimator, the held-out log-likelihood curve achieves a maximum when there are only 35 edges in the model. In contrast, the held-out log-likelihood curves of the glasso and refit glasso achieve maxima when there are around 280 edges and 100 edges respectively, while their predictive estimates are still inferior to those of the tree-based kernel density estimator.

Figure 7 also shows the estimated graphs for the tree-based kernel density estimator and the glasso. The graphs are automatically selected based on held-out log-likelihood. The two graphs are clearly different; it appears that the nonparametric tree-based kernel density estimator has the potential to provide different biological insights than the parametric Gaussian graphical model.

6.2.2 HAPMAP DATA

This data set comes from Nayak et al. (2009). The data set contains Affymetrics chip measured expression levels of 4238 genes for 295 normal subjects in the *Centre d'Etude du Polymorphisme*

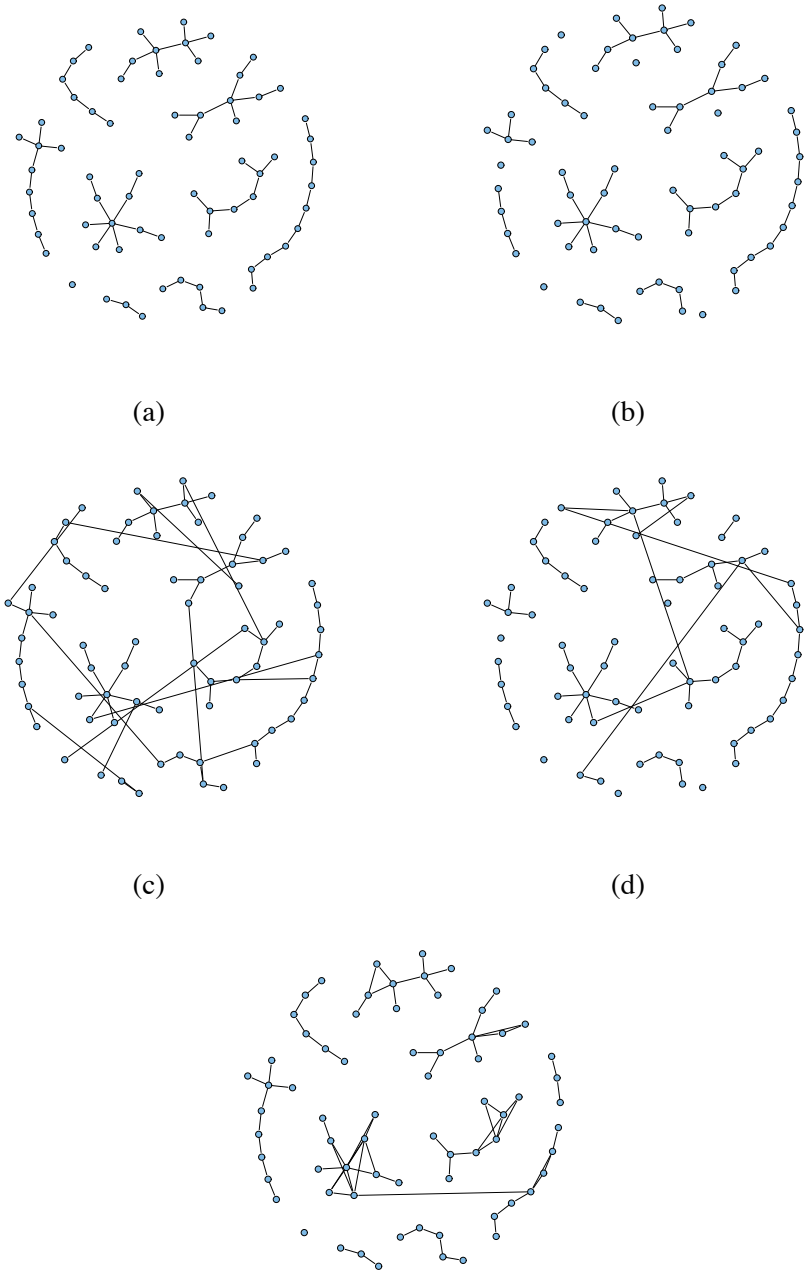


Figure 6: Graphs generated on non-Gaussian Data: (a) True Graph, (b) t-Restricted Forest (c) MST on Heldout Data (d) MST on Training Data with Pruning (e) Refit Glasso

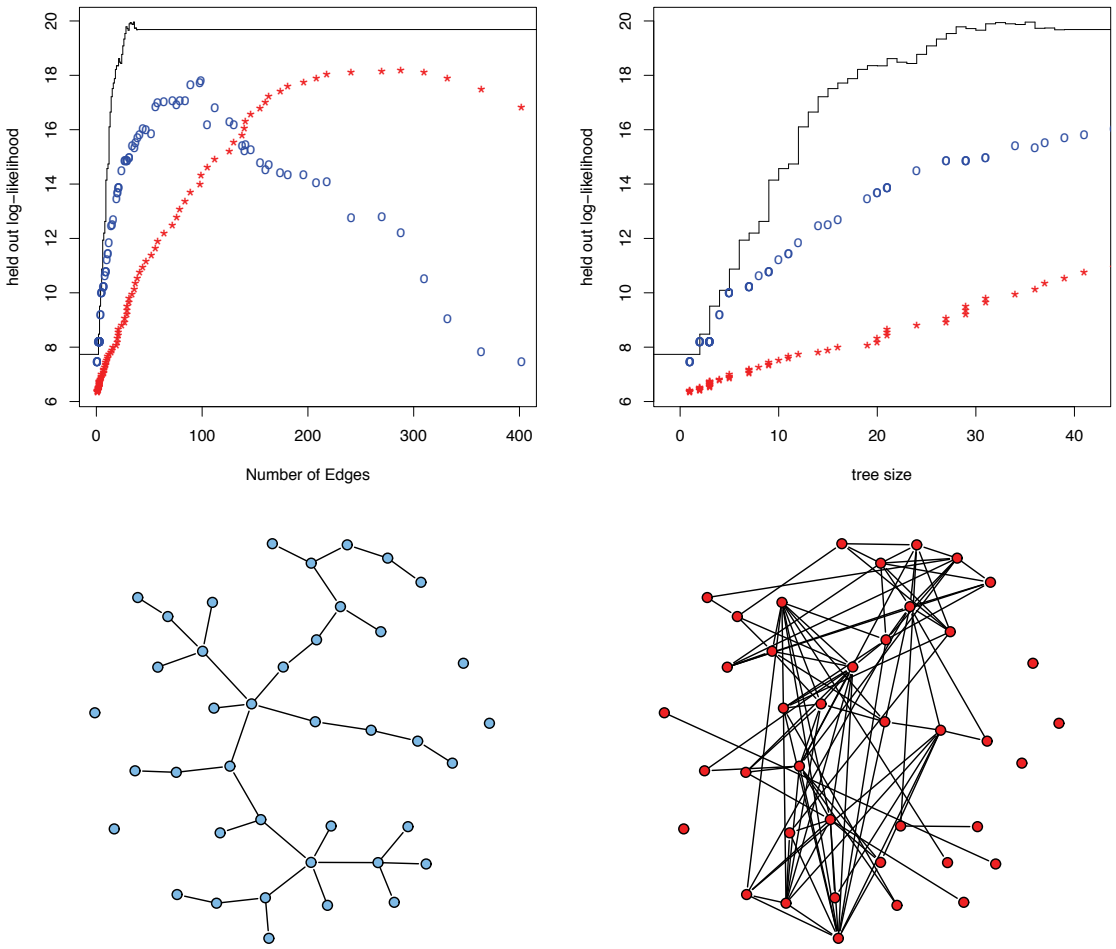


Figure 7: Results on microarray data. Top: held-out log-likelihood (left) and its zoom-in (right) of the tree-based kernel density estimator (black step function), glasso (red stars), and refit glasso (blue circles). Bottom: estimated graphs using the tree-based estimator (left) and glasso (right).

Humain (CEPH) and the International HapMap collections. The 295 subjects come from four different groups: 148 unrelated grandparents in the CEPH-Utah pedigrees, 43 Han Chinese in Beijing, 44 Japanese in Tokyo, and 60 Yoruba in Ibadan, Nigeria. Since we want to find common network patterns across different groups of subjects, we pooled the data together into a $n = 295$ by $d = 4238$ numerical matrix.

We estimate the full 4238 node graph using both the forest density estimator (described in Section 3.1 and 3.2) and the Meinshausen-Bühlmann neighborhood search method as proposed in Meinshausen and Bühlmann (2006) with regularization parameter chosen to give it about same number as edges as the forest graph.

To construct the kernel density estimates $\widehat{p}(x_i, x_j)$ we use an array of Nvidia graphical processing units (GPU) to parallelize the computation over the pairs of variables X_i and X_j. We discretize the domain of (X_i, X_j) into a 128×128 grid, and correspondingly employ 128×128 parallel cells in the GPU array, taking advantage of shared memory in CUDA. Parallelizing in this way increases the total performance by approximately a factor of 40, allowing the experiment to complete in a day.

The forest density estimated graph reveals one strongly connected component of more than 3000 genes and various isolated genes; this is consistent with the analysis in Nayak et al. (2009) and is realistic for the regulatory system of humans. The Gaussian graph contains similar component structure, but the set of edges differs significantly. We also ran the t-restricted forest algorithm for $t = 2000$ and it successfully separates the giant component into three smaller components. For visualization purposes, in Figure 8, we show only a 934 gene subgraph of the strongly connected component among the full 4238 node graphs we estimated. More detailed analysis of the biological implications of this work will left as a future study.

7. Conclusion

We have studied forest density estimation for high dimensional data. Forest density estimation skirts the curse of dimensionality by restricting to undirected graphs without cycles, while allowing fully nonparametric marginal densities. The method is computationally simple, and the optimal size of the forest can be robustly selected by a data-splitting scheme. We have established oracle properties and rates of convergence for function estimation in this setting. Our experimental results compared the forest density estimator to the sparse Gaussian graphical model in terms of both predictive risk and the qualitative properties of the estimated graphs for human gene expression array data. Together, these results indicate that forest density estimation can be a useful tool for relaxing the normality assumption in graphical modeling.

Acknowledgments

The research reported here was carried out at Carnegie Mellon University and was supported in part by NSF grant CCF-0625879, AFOSR contract FA9550-09-1-0373, and a grant from Google.

Appendix A. Proofs

In the following, we present the detailed proofs of all the technical results.

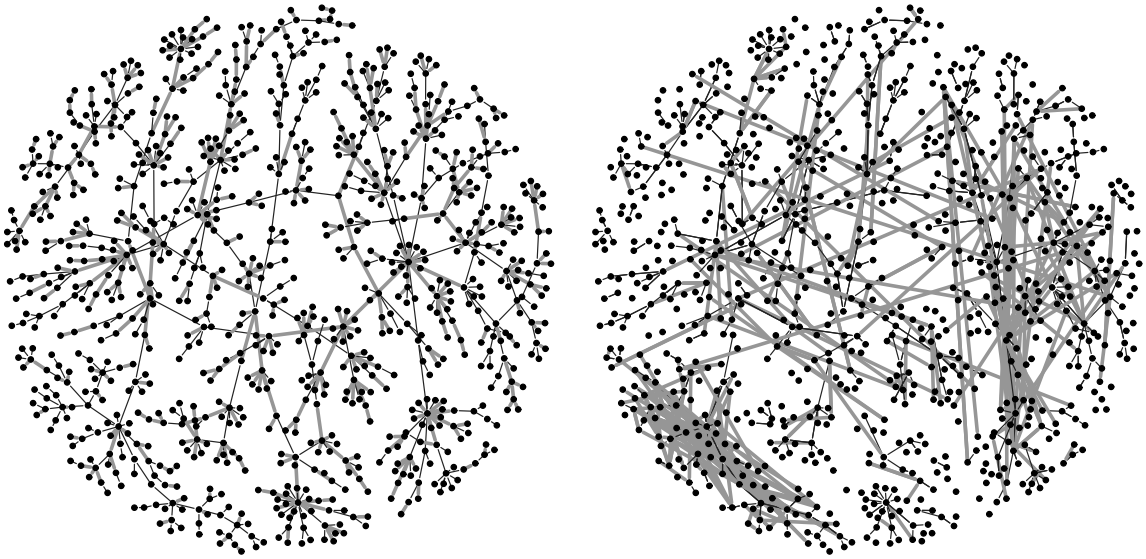


Figure 8: A 934 gene subgraph of the full estimated 4238 gene network. Left: estimated forest graph. Right: estimated Gaussian graph. The bold gray edges in the forest graph are missing from the Gaussian graph and vice versa; the thin black edges are shared by both graphs. Note that the layout of the genes is the same for both graphs.

A.1 Proof of Lemma 8

We only need to consider the more complicated bivariate case (14); the result in (15) follows from the same line of proof. First, given the assumptions, the following lemma can be obtained by an application of Corollary 2.2 of Giné and Guillou (2002). For a detailed proof, see Rinaldo and Wasserman (2010).

Lemma 18 *(Giné and Guillou, 2002) Let $\widehat{p}$ be a bivariate kernel density estimate using a kernel $K(\cdot)$ for which Assumption 2 holds and suppose that*

$$\sup_{t\in\mathcal{X}^2}\sup_{h_2>0}\int_{\mathcal{X}^2} K_2^2(u)p^*(t-uh_2)du \leq D < \infty. \tag{19}$$

1. *Let the bandwidth h_2 be fixed. Then there exit constants $L>0$ and $C>0$, which depend only on the VC characteristics of $\mathcal{F}_2$ in* (11)*, such that for any $c_1 \geq C$ and $0<\varepsilon\leq c_1 D/\|K_2\|_\infty$, there exists $n_0>0$ which depends on ε, D, $\|K_2\|_\infty$ and the VC characteristics of K_2, such that for all $n\geq n_0$,*

$$\mathbb{P}\left(\sup_{u\in\mathcal{X}^2}|\widehat{p}(u)-\mathbb{E}\widehat{p}(u)|>2\varepsilon\right)\leq L\exp\left\{-\frac{1}{L}\frac{\log(1+c_1/(4L))}{c_1}\frac{nh_2^2\varepsilon^2}{D}\right\}. \tag{20}$$

2. *Let $h_2 \to 0$ in such a way that $nh_2^2/\log h_2 \to \infty$, and let $\varepsilon \to 0$ so that*

$$\varepsilon = \Omega\left(\sqrt{\frac{\log r_n}{nh_2^2}}\right), \tag{21}$$

where $r_n = \Omega(h_2^{-1})$. Then (20) *holds for sufficiently large n.*

From $(D2)$ in Assumption 1 and $(K1)$ in Assumption 2, it is easy to see that (19) is satisfied. Also, since

$$h_2 \asymp \left(\frac{\log n}{n}\right)^{\frac{1}{2+2\beta}},$$

it is clear that $nh_2^2/\log h_2 \to \infty$. Part 2 of Lemma 18 shows that there exist c_2 and c_3 such that

$$\mathbb{P}\left(\sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |\widehat{p}(x_i,x_j) - \mathbb{E}\widehat{p}(x_i,x_j)| \geq \frac{\varepsilon}{2}\right) \leq c_2 \exp\left(-c_3 n^{\frac{\beta}{1+\beta}} (\log n)^{\frac{1}{1+\beta}} \varepsilon^2\right) \tag{22}$$

for all ε satisfying (21).

This shows that for any $i, j \in \{1,\ldots,d\}$ with $i \neq j$, the bivariate kernel density estimate $\widehat{p}(x_i,x_j)$ is uniformly close to $\mathbb{E}\widehat{p}(x_i,x_j)$. Note that $\mathbb{E}\widehat{p}(x_i,x_j)$ can be written as

$$\mathbb{E}\widehat{p}(x_i,x_j) = \int \frac{1}{h_2^2} K\left(\frac{u_i - x_i}{h_2}\right) K\left(\frac{v_j - x_j}{h_2}\right) p^*(u_i,v_j)\, du_i dv_j.$$

The next lemma, from Rigollet and Vert (2009), provides a uniform deviation bound on the bias term $\mathbb{E}\widehat{p}(x_i,x_j) - p^*(x_i,x_j)$.

Lemma 19 *(Rigollet and Vert, 2009) Under $(D1)$ in Assumption 1 and $(K3)$ in Assumption 2, we have*

$$\sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} \left|\mathbb{E}\widehat{p}(x_i,x_j) - p^*(x_i,x_j)\right| \leq L_1 h_2^{\beta} \int_{\mathcal{X}^2} (u^2+v^2)^{\beta/2} K(u)K(v)\, dudv.$$

where L is defined in $(D1)$ of Assumption 1.

Let $c_4 = L_1 \int_{\mathcal{X}^2} (u^2+v^2)^{\beta/2} K(u)K(v)\, dudv$. From the discussion of Example 6.1 in Rigollet and Vert (2009) and $(K1)$ in Assumption 2, we know that $c_4 < \infty$ and only depends on K and β. Therefore

$$\mathbb{P}\left(\sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |p^*(x_i,x_j) - \mathbb{E}\widehat{p}(x_i,x_j)| \geq \frac{\varepsilon}{2}\right) = 0 \tag{23}$$

for $\varepsilon \geq 4c_4 h_2^{\beta}$.

The desired result in Lemma 8 is an exponential probability inequality showing that $\widehat{p}(x_i,x_j)$ is close to $p^*(x_i,x_j)$. To obtain this, we use a union bound:

$$\begin{aligned}
&\mathbb{P}\left(\max_{(i,j)\in\{1,\ldots,d\}\times\{1,\ldots,d\}} \sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |\widehat{p}(x_i,x_j) - p^*(x_i,x_j)| \geq \varepsilon\right) \\
&\quad\leq\quad d^2\mathbb{P}\left(\sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |\widehat{p}(x_i,x_j) - \mathbb{E}\widehat{p}(x_i,x_j)| \geq \frac{\varepsilon}{2}\right) \\
&\qquad\quad + d^2\mathbb{P}\left(\sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |p^*(x_i,x_j) - \mathbb{E}\widehat{p}(x_i,x_j)| \geq \frac{\varepsilon}{2}\right).
\end{aligned} \tag{24}$$

The first result follows from (22) and (24).

Choosing

$$\varepsilon = \Omega\left(4c_4\sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}}\right),$$

the result directly follows by combining (22) and (23)

A.2 Proof of Theorem 9

First, from $(D2)$ in Assumption 1 and Lemma 8, we have for any $i \neq j$,

$$\max_{(i,j)\in\{1,\dots,d\}\times\{1,\dots,d\}} \sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} \left(\frac{\widehat{p}(x_i,x_j)}{p^*(x_i,x_j)} - 1\right) = O_P\left(\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}}\right).$$

The next lemma bounds the deviation of $\widehat{R}(\widehat{p}_F)$ from $R(p_F^*)$ over different choices of $F \in \mathcal{F}_d$ with $|E(F)| \leq k$. In the following, we let

$$\mathcal{F}_d^{(k)} = \{F \in \mathcal{F}_d : |E(F)| \leq k\}$$

denote the family of d-node forests with no more than k edges.

Lemma 20 *Under the assumptions of Theorem 9, we have*

$$\sup_{F\in\mathcal{F}_d^{(k)}} |\widehat{R}(\widehat{p}_F) - R(p_F^*)| = O_P\left(k\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} + d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right).$$

Proof For any $F \in \mathcal{F}_d^{(k)}$, we have

$$\begin{aligned}
&|\widehat{R}(\widehat{p}_F) - R(p_F^*)| \\
&\leq \underbrace{\left|\sum_{(i,j)\in E(F)} \left(\int_{\mathcal{X}_i\times\mathcal{X}_j} p^*(x_i,x_j)\log p^*(x_i,x_j)dx_i dx_j - \int_{\mathcal{X}_i\times\mathcal{X}_j} \widehat{p}(x_i,x_j)\log\widehat{p}(x_i,x_j)dx_i dx_j\right)\right|}_{A_1(F)} \\
&\quad + \underbrace{\left|\sum_{k\in V} (\deg_F(k) - 1)\left(\int_{\mathcal{X}_k} p^*(x_k)\log p^*(x_k)dx_k - \int_{\mathcal{X}_k} \widehat{p}(x_k)\log\widehat{p}(x_k)dx_k\right)\right|}_{A_2(F)}
\end{aligned}$$

where $\deg_F(k)$ is the degree of node k in F. Let $\varepsilon \geq 4c_4 h_2^\beta$ and let Ω_n be the event that

$$\max_{(i,j)\in\{1,\dots,d\}\times\{1,\dots,d\}} \sup_{(x_i,x_j)\in\mathcal{X}_i\times\mathcal{X}_j} |\widehat{p}(x_i,x_j) - p^*(x_i,x_j)| \leq \varepsilon.$$

By Lemma 8, Ω_n holds except on a set of probability at most

$$c_2 d^2 \exp\left(-c_3 n^{\frac{\beta}{1+\beta}} (\log n)^{\frac{1}{1+\beta}} \varepsilon^2\right).$$

From $(D2)$ in Assumption 1, and from the fact that $|\log(1+u)| \leq 2|u|$ for all small u, we have that, on the event Ω_n,

$$\sup_{F \in \mathcal{F}_d^{(k)}} A_1(F) \leq ck\varepsilon.$$

By choosing $\varepsilon = \Omega\left(4c_4 \sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}}\right)$ we conclude that

$$\sup_{F \in \mathcal{F}_d^{(k)}} A_1(F) = O_P\left(k \sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}}\right).$$

By a similar argument and using the fact that $\sum_k |\deg_F(k) - 1| = O(d)$, we have

$$\sup_{F \in \mathcal{F}_d^{(k)}} A_2(F) = O_P\left(d \sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right).$$

■

The next auxiliary lemma is also needed to obtain the main result. It shows that $\widehat{R}(\widehat{p}_F)$ does not deviate much from $R(\widehat{p}_F)$ uniformly over different choices of $F \in \mathcal{F}_d^{(k)}$.

Lemma 21 *Under the assumptions of Theorem 9, we have*

$$\sup_{F \in \mathcal{F}_d^{(k)}} |R(\widehat{p}_F) - \widehat{R}(\widehat{p}_F)| = O_P\left(k \sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} + d \sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right).$$

Proof The argument is similar to the proof of Lemma 20. ■

The proof of the main theorem follows by repeatedly applying the previous two lemmas. As in Proposition 2, with

$$p^*_{F_d^{(k)}} = \arg\min_{q_F \in \mathcal{P}_d^{(k)}} R(q_F),$$

we have

$$\begin{aligned}
& R(\widehat{p}_{\widehat{F}_d^{(k)}}) - R(p^*_{F_d^{(k)}}) \\
&= R(\widehat{p}_{\widehat{F}_d^{(k)}}) - \widehat{R}(\widehat{p}_{\widehat{F}_d^{(k)}}) + \widehat{R}(\widehat{p}_{\widehat{F}_d^{(k)}}) - R(p^*_{F_d^{(k)}}) \\
&= \widehat{R}(\widehat{p}_{\widehat{F}_d^{(k)}}) - R(p^*_{F_d^{(k)}}) + O_P\left(k\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} + d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right) \qquad (25) \\
&\leq \widehat{R}(\widehat{p}_{F_d^{(k)}}) - R(p^*_{F_d^{(k)}}) + O_P\left(k\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} + d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right) \qquad (26) \\
&= R(p^*_{F_d^{(k)}}) - R(p^*_{F_d^{(k)}}) + O_P\left(k\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} + d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right) \qquad (27) \\
&= O_P\left(k\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} + d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}\right).
\end{aligned}$$

where (25) follows from Lemma 21, (26) follows from the fact that $\widehat{p}_{\widehat{F}_d^{(k)}}$ is the minimizer of $\widehat{R}(\cdot)$, and (27) follows from Lemma 20.

A.3 Proof of Theorem 10

To simplify notation, we denote

$$\begin{aligned}
\phi_n(k) &= k\sqrt{\frac{\log n + \log d}{n^{\beta/(\beta+1)}}} \\
\psi_n(d) &= d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}.
\end{aligned}$$

Following the same proof as Lemma 21, we obtain the following.

Lemma 22 *Under the assumptions of Theorem 9, we have*

$$\sup_{F \in \mathcal{F}_d^{(k)}} |R(\widehat{p}_F) - \widehat{R}_{n_2}(\widehat{p}_F)| = O_P\Big(\phi_n(k) + \psi_n(d)\Big).$$

where $\widehat{R}_{n_2}$ is the held out risk.

To prove Theorem 10, we now have

$$\begin{aligned}
R(\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}) - R(\widehat{p}_{\widehat{F}_d^{(k^*)}}) &= R(\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}) - \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}) + \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}) - R(\widehat{p}_{\widehat{F}_d^{(k^*)}}) \\
&= O_P(\phi_n(\widehat{k}) + \psi_n(d)) + \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_d^{(\widehat{k})}}) - R(\widehat{p}_{\widehat{F}_d^{(k^*)}}) \\
&\leq O_P(\phi_n(\widehat{k}) + \psi_n(d)) + \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_d^{(k^*)}}) - R(\widehat{p}_{\widehat{F}_d^{(k^*)}}) \qquad (28) \\
&= O_P\Big(\phi_n(\widehat{k}) + \phi_n(k^*) + \psi_n(d)\Big).
\end{aligned}$$

where (28) follows from the fact that $\widehat{k}$ is the minimizer of $\widehat{R}_{n_2}(\cdot)$.

A.4 Proof of Theorem 11

Using the shorthand

$$\phi_n(k) = k\sqrt{\frac{\log n + \log d}{n^{\beta/(1+\beta)}}}$$
$$\psi_n(d) = d\sqrt{\frac{\log n + \log d}{n^{2\beta/(1+2\beta)}}}$$

We have that

$$\begin{aligned} R(\widehat{p}_{\widehat{F}_{n_2}}) - R(\widehat{p}_{F^*}) &= R(\widehat{p}_{\widehat{F}_{n_2}}) - \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_{n_2}}) + \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_{n_2}}) - R(\widehat{p}_{F^*}) \\ &= O_P(\phi_n(\widehat{k}) + \psi_n(d)) + \widehat{R}_{n_2}(\widehat{p}_{\widehat{F}_{n_2}}) - R(\widehat{p}_{F^*}) \\ &\leq O_P(\phi_n(\widehat{k}) + \psi_n(d)) + \widehat{R}_{n_2}(\widehat{p}_{F^*}) - R(\widehat{p}_{F^*}) \qquad (29) \\ &= O_P(\phi_n(\widehat{k}) + \phi_n(k^*) + \psi_n(d)) \end{aligned}$$

where (29) follows because $\widehat{F}_{n_2}$ is the minimizer of $\widehat{R}_{n_2}(\cdot)$.

A.5 Proof of Theorem 12

We begin by showing an exponential probability inequality on the difference between the empirical and population mutual informations.

Lemma 23 *Under Assumptions 1, 2, there exist generic constants c_5 and c_6 satisfying*

$$\mathbb{P}\left(|I(X_i;X_j) - \widehat{I}(X_i;X_j)| > \varepsilon\right) \leq c_5 \exp\left(-c_6 n^{\frac{\beta}{1+\beta}} (\log n)^{\frac{1}{1+\beta}} \varepsilon^2\right).$$

for arbitrary $i, j \in \{1,\ldots,d\}$ with $i \neq j$, and $\varepsilon \to 0$ so that

$$\varepsilon = \Omega\left(\sqrt{\frac{\log r_n}{n h_2^2}}\right),$$

where $r_n = \Omega(h_2^{-1})$.

Proof For any $\varepsilon = \Omega\left(\sqrt{\frac{\log r_n}{n h_2^2}}\right)$, we have

$$\begin{aligned} &\mathbb{P}\left(|I(X_i;X_j) - \widehat{I}(X_i;X_j)| > \varepsilon\right) \\ &= \mathbb{P}\left(\left|\int_{\mathcal{X}_i \times \mathcal{X}_j} p^*(x_i,x_j) \log \frac{p^*(x_i,x_j)}{p^*(x_i)p^*(x_j)} dx_i dx_j - \int_{\mathcal{X}_i \times \mathcal{X}_j} \widehat{p}(x_i,x_j) \log \frac{\widehat{p}(x_i,x_j)}{\widehat{p}(x_i)\widehat{p}(x_j)} dx_i dx_j\right| > \varepsilon\right) \\ &\leq \mathbb{P}\left(\left|\int_{\mathcal{X}_i \times \mathcal{X}_j} (p^*(x_i,x_j) \log p^*(x_i,x_j) - \widehat{p}(x_i,x_j) \log \widehat{p}(x_i,x_j))\, dx_i dx_j\right| > \frac{\varepsilon}{2}\right) \\ &\quad + \mathbb{P}\left(\left|\int_{\mathcal{X}_i \times \mathcal{X}_j} (p^*(x_i,x_j) \log p^*(x_i)p^*(x_j) - \widehat{p}(x_i,x_j) \log \widehat{p}(x_i)\widehat{p}(x_j))\, dx_i dx_j\right| > \frac{\varepsilon}{2}\right) \qquad (30) \end{aligned}$$

Since the second term of (30) only involves univariate kernel density estimates, this term is dominated by the first term, and we only need to analyze

$$\mathbb{P}\left(\left|\int_{\mathcal{X}_i\times\mathcal{X}_j}\left(p^*(x_i,x_j)\log p^*(x_i,x_j)-\widehat{p}(x_i,x_j)\log\widehat{p}(x_i,x_j)\right)dx_i dx_j\right|>\frac{\varepsilon}{2}\right).$$

The desired result then follows from the same analysis as in Lemma 20. ■

Let

$$L_n=\Omega\left(\sqrt{\frac{\log n+\log d}{n^{\beta/(1+\beta)}}}\right)$$

be defined as in Assumption 3. To prove the main theorem, we see the event $\widehat{F}_d^{(k)}\neq F_d^{(k)}$ implies that there must be at least exist two pairs of edges (i,j) and (k,ℓ), such that

$$\operatorname{sign}\Big(I(X_i,X_j)-I(X_k,X_\ell)\Big)\neq\operatorname{sign}\Big(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\Big). \tag{31}$$

Therefore, we have

$$\begin{aligned}&\mathbb{P}\left(\widehat{F}_d^{(k)}\neq F_d^{(k)}\right)\\ &\quad\leq\ \mathbb{P}\left(\Big(I(X_i,X_j)-I(X_k,X_\ell)\Big)\cdot\Big(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\Big)\leq 0,\ \text{for some } (i,j),(k,\ell)\right).\end{aligned}$$

With d nodes, there can be no more than $d^4/2$ pairs of edges; thus, applying a union bound yields

$$\begin{aligned}&\mathbb{P}\left(\Big(I(X_i,X_j)-I(X_k,X_\ell)\Big)\cdot\Big(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\Big)\leq 0,\ \text{for some } (i,j),(k,\ell)\right)\\ &\quad\leq\ \frac{d^4}{2}\max_{((i,j),(k,\ell))\in\mathcal{J}}\mathbb{P}\left(\Big(I(X_i,X_j)-I(X_k,X_\ell)\Big)\cdot\Big(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\Big)\leq 0\right).\end{aligned}$$

Assumption 3 specifies that

$$\min_{((i,j),(k,\ell))\in\mathcal{J}}|I(X_i,X_j)-I(X_k,X_\ell)|>2L_n.$$

Therefore, in order for (31) hold, there must exist an edge $(i,j)\in\mathcal{J}$ such that

$$|I(X_i,X_j)-\widehat{I}(X_i,X_j)|>L_n.$$

Thus, we have

$$\begin{aligned}&\max_{((i,j),(k,\ell))\in\mathcal{J}}\mathbb{P}\left(\Big(I(X_i,X_j)-I(X_k,X_\ell)\Big)\cdot\Big(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\Big)\leq 0\right)\\ &\quad\leq\ \max_{i,j\in\{1,\ldots,d\},i\neq j}\mathbb{P}\Big(|I(X_i,X_j)-\widehat{I}(X_i,X_j)|>L_n\Big)\\ &\quad\leq\ c_5\exp\left(-c_6 n^{\frac{\beta}{1+\beta}}(\log n)^{\frac{1}{1+\beta}}L_n^2\right).\end{aligned}\tag{32}$$

where (32) follows from Lemma 23.

Chaining together the above arguments, we obtain

$$\begin{aligned}
&\mathbb{P}\left(\widehat{F}_d^{(k)} \neq F_d^{(k)}\right) \\
&\leq \mathbb{P}\left(\left(I(X_i,X_j)-I(X_k,X_\ell)\right)\cdot\left(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\right)\leq 0, \text{ for some } (i,j),(k,\ell)\right) \\
&\leq \frac{d^4}{2}\max_{((i,j),(k,\ell))\in\mathcal{J}}\mathbb{P}\left(\left(I(X_i,X_j)-I(X_k,X_\ell)\right)\cdot\left(\widehat{I}(X_i,X_j)-\widehat{I}(X_k,X_\ell)\right)\leq 0\right) \\
&\leq d^4\max_{i,j\in\{1,\ldots,d\},i\neq j}\mathbb{P}\left(|I(X_i,X_j)-\widehat{I}(X_i,X_j)|>L_n\right) \\
&\leq d^4 c_5\exp\left(-c_6 n^{\frac{\beta}{1+\beta}}(\log n)^{\frac{1}{1+\beta}}L_n^2\right) \\
&= o\left(c_5\exp\left(4\log d - c_6(\log n)^{\frac{1}{1+\beta}}\log d\right)\right) \\
&= o(1).
\end{aligned}$$

The conclusion of the theorem now directly follows.

A.6 Proof of NP-hardness of t-Restricted Forest

We will reduce an instance of exact 3-cover (X3C) to an instance of finding a maximum weight t-restricted forest (t-RF).

Recall that in X3C, we are given a finite set X with $|X| = 3q$ and a family of 3-element subsets of X, $\mathcal{S} = \{S \subset X : |S| = 3\}$. The objective is to find a subfamily $\mathcal{S}' \subset \mathcal{S}$ such that every element of X occurs in exactly one member of $\mathcal{S}'$, or to determine that no such subfamily exists.

Suppose then we are given $X = \{x_1,\ldots,x_n\}$ and $\mathcal{S} = \{S \subset X : |S| = 3\}$, with $m = |\mathcal{S}|$. We construct the graph G in an instance of t-RF as follows, and as illustrated in Figure 9.

For each $x \in X$, add an *element node* to G. For each $S \in \mathcal{S}$, construct a *gadget*, which is a subgraph comprised of a *nexus node*, three *junction nodes*, and three *lure nodes*; see Figure 9. We assign weights to the edges in a gadget in the following manner:

$$\begin{aligned}
w(\text{element},\text{junction}) &= 2 \\
w(\text{nexus},\text{lure}_1) &= 5 \\
w(\text{lure}_1,\text{lure}_2) &= 10 \\
w(\text{lure}_2,\text{lure}_3) &= 10 \\
w(\text{nexus},\text{junction}) &= N > 31m.
\end{aligned}$$

Note that the weight N is chosen to be strictly greater than the weight all of the non-nexus-junction edges in the graph combined. To complete the instance of t-RF, let $t = 7$.

Lemma 24 *Suppose G is a graph constructed in the transformation from X3C described above. Then F_t^* must contain all the nexus-junction edges.*

Proof The set of all nexus-junction edges together form a well-defined t-restricted forest, since each subtree has a nexus node and 3 junction nodes. Call this forest F. If some forest F' is missing a nexus-junction edge, then F' must have weight strictly less than F, since N is larger than the sum of all of the non-nexus-junction edges. ■

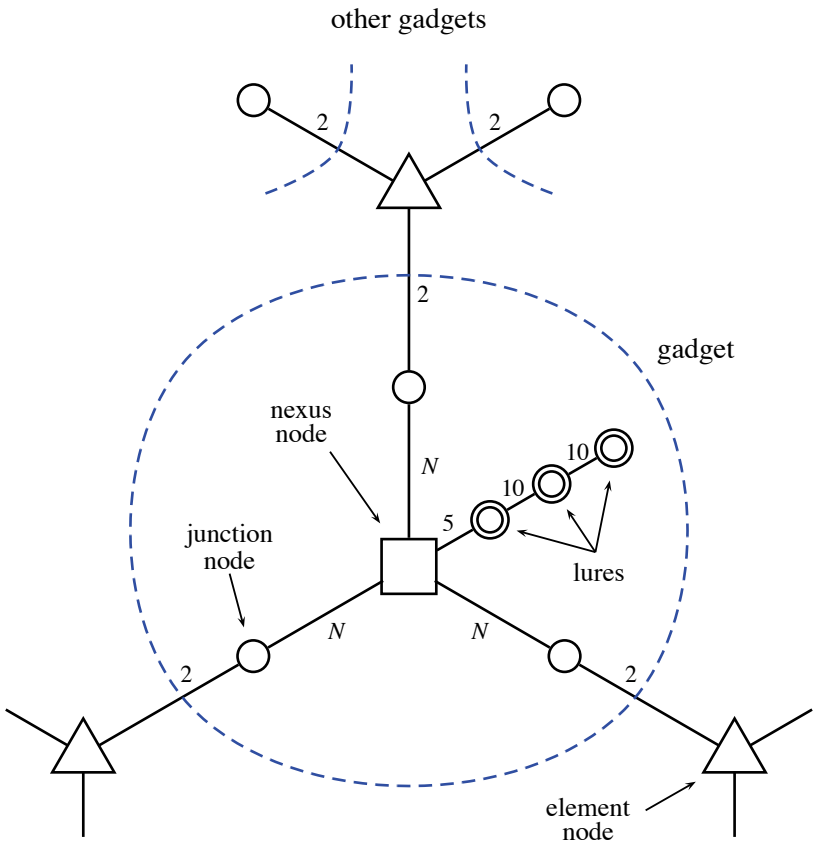


Figure 9: Gadget constructed in the reduction from X3C

Lemma 25 *Each subtree in F_t^* can contain at most one nexus node.*

Proof Suppose a subtree T in F_t^* contains two nexus nodes. Then it must contain 6 junction nodes by Lemma 24. Thus, T contains at least 8 nodes, and therefore violates the t-restriction constraint. ■

Lemma 26 *For each nexus node contained in F_t^*, the corresponding three junction nodes are either connected to all or none of the three neighboring element nodes.*

Proof By the previous two Lemmas 24 and 25, each subtree is associated with at most one gadget, and hence at most one $S \in \mathcal{S}$, and moreover each gadget has as least one associated subtree.

Without loss of generality, we consider a region of the graph local to some arbitrary subtree. By the size constraint, a subtree cannot contain all the adjacent element nodes and all the lure nodes.

We now perform a case analysis:

1. If a subtree contains no element nodes and all the lure nodes, then it has weight $3N+25$. Call this an OFF configuration.

2. If a subtree contains two element nodes, and a second subtree of three nodes contains all the lure nodes, then the total weight of both subtrees is $3N+24$. This is suboptimal because we can convert to an OFF configuration and gain additional weight without affecting any other subtrees. Hence, such a configuration cannot exist in F_t^*.
3. If a subtree contains two element nodes and lure_1, and a second subtree contains just lure_2 and lure_3, then the total weight of the two subtrees is $3N+19$. This is again suboptimal.
4. If a subtree contains an element node and both lure_1 and lure_2, then there cannot be a second subtree in region local to the gadget. The weight of this one subtree is $(3N+2+5+10) = 3N+17$, which is suboptimal.
5. If a subtree contains all three element nodes and no lure nodes, and a second subtree contains all the lure nodes, then the total weight is $(3N+6)+20 = 3N+26$. Call this an ON configuration.

Thus, we see that each gadget in F_t^* must be either an ON or an OFF configuration. ■

Recall that each gadget corresponds to a 3-element subset S in the family $\mathcal{S}$. Since a gadget in an ON configuration has greater weight than a gadget in an OFF configuration, an optimal t-RF will have as many gadgets in the ON configuration as possible. Thus, to solve X3C we can find the optimal t-RF and, to obtain a subcover $\mathcal{S}'$, we place all S into $\mathcal{S}'$ that correspond to ON gadgets in the forest. By Lemma 25 each subtree can contain at most one nexus node, which implies that each ON gadget is connected to element nodes that are not connected to any other ON gadgets. Thus, this results in a subcover for which each element of X appears in at most one $S \in \mathcal{S}'$.

A.7 Proof of Theorem 16

Recall that we want to show that Algorithm 2 returns a forest with weight that is at least a quarter of the weight of the optimal t-restricted forest. Let us distinguish two types of constraints:

(a) the degree of any node is at most t;
(b) the graph is acyclic.

Note that the optimal t-restricted forest F_t^* satisfies both the constraints above, and hence the maximum weight set of edges that satisfy both the constraints above has weight at least $w(F_t^*)$. Recall that the first stage of Algorithm 2 greedily adds edges subject to these two constraints—the next two lemmas show that the resulting forest has weight at least $\frac{1}{2}w(F_t^*)$.

Lemma 27 *The family of subgraphs satisfying the constraints (a) and (b) form a 2-independence family. That is, for any subgraph T satisfying (a) and (b), and for any edge $e \in G$, there exist at most two edges $\{e_1, e_2\}$ in T such that $T \cup \{e\} - \{e_1, e_2\}$ also satisfies constraints (a) and (b).*

Proof Let T be a subgraph satisfying (a) and (b) and suppose we add $e = (u,v)$ in T. Then the degrees of both u and v are at most $t+1$. If no cycles were created, then we can simply remove an edge in T containing u (if any) and an edge in T containing v (if any) to satisfy the degree constraint (a) as well. If adding e created a cycle of the form $\{\ldots,(u',u),(u,v),(v,v')\}$, then the edges (u',u) and (v,v') can be removed to satisfy both constraints (a) and (b). ■

Lemma 28 *Let F_1 be the forest output after Step 1 of algorithm 2. Then $w(F_1) \geq \frac{1}{2} w(F_t^*)$.*

Proof Let F^{**} be a maximum weight forest that obeys both constraints (a) and (b). Since the optimal t-restructed forest F_t^* obeys both these constraints, we have $w(F_t^*) \leq w(F^{**})$. By a theorem of Hausmann et al. (1980), in a p-independence family the greedy algorithm is a $\frac{1}{p}$-approximation to the maximum weight p-independent set. By Lemma 27, we know that the set of all subgraphs satisfying constraints (a) and (b) is a 2-independent family. Hence, $w(F_1) \geq \frac{1}{2} w(F^{**}) \geq \frac{1}{2} w(F_t^*)$. ■

We can now turn to the proof of Theorem 16.

Proof Given a graph G, let F_1 be the forest output by first step of Algorithm 2, and let F_A be the forest outputted by the second step. We claim that $w(F_A) \geq \frac{1}{2} w(F_1)$; combined with Lemma 28, this will complete the proof of the theorem.

To prove the claim, we first show that given any tree T with edge weights and maximum degree $t \geq 2$, we can obtain a sub-forest F with total weight $w(F) \geq \frac{1}{2} w(T)$, and where the number of edges in each tree in the forest F is at most $t-1$. Indeed, root the tree T at an arbitrary node of degree-1, and call an edge e *odd* or *even* depending on the parity of the number of edges in the unique path between e and the root. Note that the set of odd edges and the set of even edges partition T into subforests composed entirely of stars of maximum degree $t-1$, and one of these sub-forests contains half the weight of T, which is what we wanted to show.

Applying this procedure to each tree T in the forest F_1, we get the existence of a $t-1$-restricted subforest $F_1' \subseteq F_1$ that has weight at least $\frac{1}{2} w(F_1)$. Observe that a $t-1$-restricted subforest is *a fortiori* a k-restricted subforest, and since $w(F_A)$ is the best t-restricted subforest of F_1, we have

$$w(F_A) \geq w(F_1') \geq \frac{1}{2} w(F_1) \geq \frac{1}{4} w(F_t^*),$$

completing the proof. ■

A.7.1 AN IMPROVED APPROXIMATION ALGORITHM

We can get an improved approximation algorithm based on a linear programming approach. Recall that F^{**} is a maximum weight forest satisfying both (a) and (b). A result of Singh and Lau (2007) implies that given any graph G with non-negative edge weights, one can find in polynomial time a forest F_{SL} such that

$$w(F_{SL}) \geq w(F^{**}) \geq w(F_t^*), \tag{33}$$

but where the maximum degree in F_{SL} is $t+1$. Now applying the procedure from the proof of Theorem 16, we get a t-restricted forest F_{SL}' whose weight is at least half of $w(F_{SL})$. Combining this with (33) implies that $w(F_{SL}') \geq w(F_t^*)$, and completes the proof of the claimed improved approximation algorithm. We remark that the procedure of Singh and Lau (2007) to find the forest F_{SL} is somewhat computationally intensive, since it requires solving vertex solutions to large linear programs.

A.8 Proof of Theorem 17

Proceeding as in the proof of Theorem 10, we have that

$$\begin{aligned}\left|R(\widehat{p}_{\widehat{F}_t}) - R(\widehat{p}_{F_t^*})\right| &\leq R(\widehat{p}_{\widehat{F}_t}) - \widehat{R}_{n_1}(\widehat{p}_{\widehat{F}_t}) + \left|\widehat{R}_{n_1}(\widehat{p}_{\widehat{F}_t}) - R(\widehat{p}_{F_t^*})\right| \\ &= O_P\left(k\phi_n(d) + d\psi_n(d)\right) + \left|\widehat{R}_{n_1}(\widehat{p}_{\widehat{F}_t}) - R(\widehat{p}_{F_t^*})\right|.\end{aligned}$$

Now, let $\widehat{H}_{n1}$ denote the estimated entropy $H(X) = \sum_k H(X_k)$, constructed using the kernel density estimates $\widehat{p}_{n_1}(x_k)$. Since the risk is the negative expected log-likelihood, we have using the approximation guarantee that

$$\begin{aligned}\widehat{R}_{n_1}(\widehat{p}_{\widehat{F}_t}) - R(\widehat{p}_{F_t^*}) &= -\widehat{w}_{n_1}(\widehat{F}_t) + \widehat{H}_{n_1} - R(\widehat{p}_{F_t^*}) \\ &\leq -\frac{1}{c}\widehat{w}_{n_1}(F_t^*) + \widehat{H}_{n_1} - R(\widehat{p}_{F_t^*}) \\ &= \widehat{R}_{n_1}(\widehat{p}^*_{F_t}) + \frac{c-1}{c}\widehat{w}_{n_1}(F_t^*) - R(\widehat{p}_{F_t^*}) \\ &= O_P\left(k^*\phi_n(d) + d\psi_n(d) + \frac{c-1}{c}w(F_t^*)\right)\end{aligned}$$

and the result follows.

A.9 The TreePartition Subroutine

To produce the best t-restricted subforest of the forest F_1, we use a divide-and-conquer forest partition algorithm described by Lukes (1974), which we now describe in more detail.

To begin, note that finding an optimal subforest is equivalent to finding a partition of the nodes in the forest, where each disjoint tree in the subforest is a cluster in the partition. Since a forest contains a disjoint set of trees, it suffices to find the optimal t-restricted partition of each of the trees.

For every subtree T, with root v, we will find a *list of partitions* $v.P = \{v.P_0, v.P_1, ..., v.P_k\}$ such that

1. for $i \neq 0$, $v.P_i$ is a partition whose cluster containing root v has size i;
2. $v.P_i$ has the maximum weight among all partitions satisfying the above condition.

We define $v.P_0$ to be $\arg\max\{w(v.P_1), \ldots, w(v.P_t)\}$. The `Merge` subroutine used in `TreePartition` takes two lists of partitions $\{v.P, u_i.P\}$, where v is the parent of u_i, $v.P$ is a partition of node v unioned with subtrees of children $\{u_1, \ldots, u_{i-1}\}$, and $u_i.P$ is a partition of the subtree of child u_i; refer to Figure 10.

Since a partition is a list of clusters of nodes, we denote by $\texttt{Concat}(v.P_2, u.P_{k-2})$ the concatenation of clusters of partitions $v.P_2, u.P_{k-2}$. Note that the concatenation forms a partition if $v.P_2$ and $u.P_{k-2}$ are respectively partitions of two disjoint sets of vertices. The weight of a partition is denoted $w(v.P_2)$, that is, the weight of all edges between nodes of the same cluster in the partition $v.P_2$.

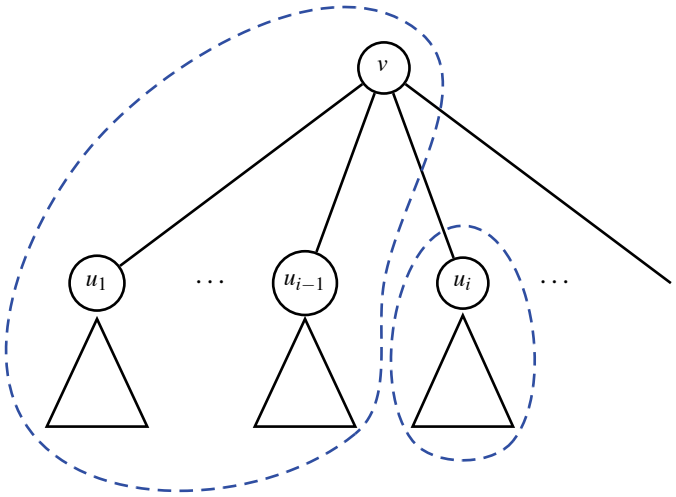


Figure 10: The `TreePartition` procedure to merge two subproblems.

Algorithm 4 `TreePartition`(T,t)

Input a tree T, a positive integer t

Returns an optimal partition into trees of size $\leq t$.

Initialize $v.P_1 = [\{v\}]$ where v is root of T, if v has no children, return $v.P_1$

For all children $\{u_1,...u_s\}$ of v, recursively call `TreePartition`(u_i,t) to get a collection of lists of partitions $\{u_1.P,u_2.P,...u_s.P\}$

For each child $u_i \in \{u_1,...u_s\}$ of v
Update $v.P \leftarrow$ `Merge`$(u_i.P,v.P)$

Output $v.P_0$

Algorithm 5 `Merge`$(v.P,u.P)$

Input a list of partitions $v.P$ and $u.P$, where v is a parent of u.

Returns a single list of partitions $v.P'$.

For $i = 1,\ldots,t$:

1. Let $(s^*,t^*) = \arg\max_{(s,t):s+t=i} w(\texttt{Concat}(v.P_s,u.P_t))$
2. Let $v.P'_i = \texttt{Concat}(v.P_{s^*},u.P_{t^*})$

Select $v.P'_0 = \arg\max_{v.P'_i} w(v.P'_i)$

Output $\{v.P'_0,v.P'_1,...v.P'_n\}$

Appendix B. Computation of the Mutual Information Matrix

In this appendix we explain different methods for computing the mutual information matrix, and making the tree estimation more efficient. One way to evaluate the empirical mutual information is to use

$$\widehat{I}(X_i;X_j) = \frac{1}{n_1} \sum_{s \in \mathcal{D}_1} \log \frac{\widehat{p}_{n_1}(X_i^{(s)}, X_j^{(s)})}{\widehat{p}_{n_1}(X_i^{(s)})\, \widehat{p}_{n_1}(X_j^{(s)})}. \tag{34}$$

Compared with our proposed method

$$\widehat{I}_{n_1}(X_i,X_j) = \frac{1}{m^2} \sum_{k=1}^{m} \sum_{\ell=1}^{m} \widehat{p}_{n_1}(x_{ki}, x_{\ell j}) \log \frac{\widehat{p}_{n_1}(x_{ki}, x_{\ell j})}{\widehat{p}_{n_1}(x_{ki})\, \widehat{p}_{n_1}(x_{\ell j})}, \tag{35}$$

(34) is somewhat easier to calculate. However, if the sample size in $\mathcal{D}_1$ is small, the approximation error can be large. A different analysis is needed to provide justification of the method based on (34), which would be more difficult since $\widehat{p}_{n_1}(\cdot)$ is dependent on $\mathcal{D}_1$. For these reasons we use the method in (35).

Also, note that instead of using the grid based method to evaluate the numerical integral, one could use sampling. If we can obtain m_1 i.i.d. samples from the bivariate density $\widehat{p}(X_i,X_j)$,

$$\left\{ (X_i^{(s)}, X_j^{(s)}) \right\}_{s=1}^{m_1} \overset{\text{i.i.d.}}{\sim} \widehat{p}_{n_1}(x_i,x_j),$$

then the empirical mutual information can be evaluated as

$$\widehat{I}(X_i;X_j) = \frac{1}{m_1} \sum_{s=1}^{m_1} \log \frac{\widehat{p}(X_i^{(s)}, X_j^{(s)})}{\widehat{p}(X_i^{(s)})\widehat{p}(X_j^{(s)})}.$$

Compared with (34), the main advantage of this approach is that the estimate can be arbitrarily close to (8) for large enough m_1 and m. Also, the computation can be easier compared to Algorithm 1. Let $\widehat{p}_{n_1}(X_i,X_j)$ be the bivariate kernel density estimator on $\mathcal{D}_1$. To sample a point from $\widehat{p}_{n_1}(X_i,X_j)$, we first random draw a sample $(X_i^{(k')}, X_j^{(\ell')})$ from $\mathcal{D}_1$, and then sample a point (X,Y) from the bivariate distribution

$$(X,Y) \sim \frac{1}{h_2^2} K\left(\frac{X_i^{(k')} - \cdot}{h_2} \right) K\left(\frac{X_j^{(\ell')} - \cdot}{h_2} \right).$$

Though this sampling strategy is superior to Algorithm 1, it requires evaluation of the bivariate kernel density estimates on many random points, which is time consuming; the grid-based method is preferred.

In our two-stage procedure, the stage requires calculation of the empirical mutual information $\widehat{I}(X_i;X_j)$ for $\binom{d}{2}$ entries. Each requires $O(m^2 n_1)$ work to evaluate the bivariate and univariate kernel density estimates on the $m \times m$ grid, in a naive implementation. Therefore, the total time to calculate the empirical mutual information matrix M is $O(m^2 n_1 d^2)$. In the second stage, the time complexity of the Chow-Liu algorithm is dominated by the first step. Therefore the total time complexity is $O\left(m^2 n_1 d^2\right)$. The first stage requires $O(d^2)$ space to store the matrix M and $O(m^2 n_1)$ space to

Algorithm 6 More efficient calculation of the mutual information matrix M.

Initialize $M = \mathbf{0}_{d\times d}$ and $H^{(i)} = \mathbf{0}_{n_1\times m}$ for $i = 1,\ldots,d$.
% calculate and pre-store the univariate KDE
for $k = 1,\ldots,d$ **do**
  for $k' = 1,\ldots,m$ **do**
    $\widehat{p}(x_k^{(k')}) \leftarrow \frac{1}{n_1}\sum_{s\in\mathcal{D}_1}\frac{1}{h_1}K\left(\frac{X_k^{(s)} - x_k^{(k')}}{h_1}\right)$
for $k' = 1,\ldots,m$ **do**
  % calculate the components used for the bivariate KDE
  for $i' = 1,\ldots,n_1$ **do**
    for $i = 1,\ldots,d$ **do**
      $H^{(i)}(i',k') \leftarrow \frac{1}{h_2}K\left(\frac{X_i^{i'} - x_i^{(k')}}{h_2}\right)$
  % calculate the mutual information matrix
  for $\ell' = 1,\ldots,m$ **do**
    for $i = 1,\ldots,d-1$ **do**
      for $j = i+1,\ldots,d$ **do**
        $\widehat{p}(x_i^{(k')}, x_j^{(\ell')}) \leftarrow 0$
      for $i' = 1,\ldots,n_1$ **do**
        $\widehat{p}(x_i^{(k')}, x_j^{(\ell')}) \leftarrow \widehat{p}(x_i^{(k')}, x_j^{(\ell')}) + H^{(i)}(i',k')\cdot H^{(j)}(i',\ell')$
      $\widehat{p}(x_i^{(k')}, x_j^{(\ell')}) \leftarrow \widehat{p}(x_i^{(k')}, x_j^{(\ell')})/n_1$
      $M(i,j) \leftarrow M(i,j) + \frac{1}{m^2}\widehat{p}(x_i^{(k')}, x_j^{(\ell')})\cdot\log\left(\widehat{p}(x_i^{(k')}, x_j^{(\ell')})/(\widehat{p}(x_i^{(k')})\cdot\widehat{p}(x_j^{(\ell')}))\right)$

evaluate the kernel density estimates on $\mathcal{D}_1$. The space complexity for the Chow-Liu algorithm is $O(d^2)$, and thus the total space complexity is $O(d^2 + m^2 n_1)$.

The quadratic time and space complexity in the number of variables d is acceptable for many practical applications but can be prohibitive when the dimension d is large. The main bottleneck is to calculate the empirical mutual information matrix M. Due to the use of the kernel density estimate, the time complexity is $O(d^2 m^2 n_1)$. The straightforward implementation in Algorithm 1 is conceptually easy but computationally inefficient, due to many redundant operations. For example, in the nested for loop, many components of the bivariate and univariate kernel density estimates are repeatedly evaluated. In Algorithm 6, we suggest an alternative method which can significantly reduce such redundancy at the price of increased but still affordable space complexity.

The main technique used in Algorithm 6 is to change the order of the multiple nested for loops, combined with some pre-calculation. This algorithm can significantly boost the empirical performance, although the worst case time complexity remains the same. An alternative suggested by Bach and Jordan (2003) is to approximate the mutual information, although this would require further analysis and justification.

References

Martin Aigner and Gïter Ziegler. *Proofs from THE BOOK*. Springer-Verlag, 1998.

Francis R. Bach and Michael I. Jordan. Beyond independent components: Trees and clusters. *Journal of Machine Learning Research*, 4:1205–1233, 2003.

Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. Model selection through sparse maximum likelihood estimation. *Journal of Machine Learning Research*, 9:485–516, March 2008.

Arthur Cayley. A theorem on trees. *Quart. J. Math.*, 23:376–378, 1889.

Anton Chechetka and Carlos Guestrin. Efficient principled learning of thin junction trees. In *In Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2007.

C. K. Chow. and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

Jianqing Fan and Irène Gijbels. *Local Polynomial Modelling and Its Applications*. Chapman and Hall, 1996.

Jerome H. Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2007.

Michael Garey and David Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979. ISBN 0-7167-1044-7.

Evarist Giné and Armell Guillou. Rates of strong uniform consistency for multivariate kernel density estimators. *Annales de l'institut Henri Poincaré (B), Probabilités et Statistiques*, 38:907–921, 2002.

Dirk Hausmann, Bernhard Korte, and Tom Jenkyns. Worst case analysis of greedy type algorithms for independence systems. *Math. Programming Studies*, 12:120–131, 1980.

Joseph B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.

Steffen L. Lauritzen. *Graphical Models*. Clarendon Press, 1996.

Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *Journal of Machine Learning Research*, 10:2295–2328, October 2009.

Joseph A. Lukes. Efficient algorithm for the partitioning of trees. *IBM Jour. of Res. and Dev.*, 18 (3):274, 1974.

Nicolai Meinshausen and Peter Bühlmann. High dimensional graphs and variable selection with the Lasso. *The Annals of Statistics*, 34:1436–1462, 2006.

Renuka Nayak, Michael Kearns, Richard Spielman, and Vivian Cheung. Coexpression network based on natural variation in human gene expression reveals gene interactions and functions. *Genome Research*, 19:1953–1962, 2009.

Deborah Nolan and David Pollard. U-processes: Rates of convergence. *The Annals of Statistics*, 15 (2):780 – 799, 1987.

Philippe Rigollet and Régis Vert. Optimal rates for plug-in estimators of density level sets. *Bernoulli*, 15(4):1154–1178, 2009.

Alessandro Rinaldo and Larry Wasserman. Generalized density clustering. *Ann. Statist.*, 38(5): 2678–2722, 2010.

Mohit Singh and Lap Chi Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *STOC'07—Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 661–670. ACM, New York, 2007.

Vincent Tan, Animashree Anandkumar, and A. Willsky. Learning Gaussian tree models: Analysis of error exponents and extremal structures. *IEEE Trans. on Signal Processing*, 58(5):2701–2714, 2010.

Vincent Tan, Animashree Anandkumar, Lang Tong, and Alan Willsky. A large-deviation analysis for the maximum likelihood learning of tree structures. *IEEE Trans. on Info. Theory*, 57(3): 1714–1735, 2011.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Publishing Company, Incorporated, 2008.

Anja Wille, Philip Zimmermann, Eva Vranová, Andreas Fürholz, Oliver Laule, Stefan Bleuler, Lars Hennig, Amela Prelić, Peter von Rohr, Lothar Thiele, Eckart Zitzler, Wilhelm Gruissem, and Peter Bühlmann. Sparse Gaussian graphical modelling of the isoprenoid gene network in *Arabidopsis thaliana*. *Genome Biology*, 5:R92, 2004.

Journal of Machine Learning Research 12 (2011) 953-997 Submitted 2/10; Revised 10/10; Published 3/11

ℓ_p-Norm Multiple Kernel Learning

Marius Kloft[*] MKLOFT@CS.BERKELEY.EDU
University of California
Computer Science Division
Berkeley, CA 94720-1758, USA

Ulf Brefeld BREFELD@YAHOO-INC.COM
Yahoo! Research
Avinguda Diagonal 177
08018 Barcelona, Spain

Sören Sonnenburg[†] SOEREN.SONNENBURG@TU-BERLIN.DE
Technische Universität Berlin
Franklinstr. 28/29
10587 Berlin, Germany

Alexander Zien[‡] ZIEN@LIFEBIOSYSTEMS.COM
LIFE Biosystems GmbH
Belfortstraße 2
69115 Heidelberg, Germany

Editor: Francis Bach

Abstract

Learning linear combinations of multiple kernels is an appealing strategy when the right choice of features is unknown. Previous approaches to multiple kernel learning (MKL) promote sparse kernel combinations to support interpretability and scalability. Unfortunately, this ℓ_1-norm MKL is rarely observed to outperform trivial baselines in practical applications. To allow for robust kernel mixtures that generalize well, we extend MKL to arbitrary norms. We devise new insights on the connection between several existing MKL formulations and develop two efficient *interleaved* optimization strategies for arbitrary norms, that is ℓ_p-norms with $p \geq 1$. This interleaved optimization is much faster than the commonly used wrapper approaches, as demonstrated on several data sets. A theoretical analysis and an experiment on controlled artificial data shed light on the appropriateness of sparse, non-sparse and ℓ_∞-norm MKL in various scenarios. Importantly, empirical applications of ℓ_p-norm MKL to three real-world problems from computational biology show that non-sparse MKL achieves accuracies that surpass the state-of-the-art.

Data sets, source code to reproduce the experiments, implementations of the algorithms, and further information are available at `http://doc.ml.tu-berlin.de/nonsparse_mkl/`.

Keywords: multiple kernel learning, learning kernels, non-sparse, support vector machine, convex conjugate, block coordinate descent, large scale optimization, bioinformatics, generalization bounds, Rademacher complexity

---

∗. Also at Machine Learning Group, Technische Universität Berlin, 10587 Berlin, Germany.

†. Parts of this work were done while SS was at the Friedrich Miescher Laboratory, Max Planck Society, 72076 Tübingen, Germany.

‡. Most contributions by AZ were done at the Fraunhofer Institute FIRST, 12489 Berlin, Germany.

©2011 Marius Kloft, Ulf Brefeld, Sören Sonnenburg and Alexander Zien.

1. Introduction

Kernels allow to decouple machine learning from data representations. Finding an appropriate data representation via a kernel function immediately opens the door to a vast world of powerful machine learning models (e.g., Schölkopf and Smola, 2002) with many efficient and reliable off-the-shelf implementations. This has propelled the dissemination of machine learning techniques to a wide range of diverse application domains.

Finding an appropriate data abstraction—or even engineering *the best* kernel—for the problem at hand is not always trivial, though. Starting with cross-validation (Stone, 1974), which is probably the most prominent approach to general model selection, a great many approaches to selecting the right kernel(s) have been deployed in the literature.

Kernel target alignment (Cristianini et al., 2002; Cortes et al., 2010b) aims at learning the entries of a kernel matrix by using the outer product of the label vector as the ground-truth. Chapelle et al. (2002) and Bousquet and Herrmann (2002) minimize estimates of the generalization error of support vector machines (SVMs) using a gradient descent algorithm over the set of parameters. Ong et al. (2005) study hyperkernels on the space of kernels and alternative approaches include selecting kernels by DC programming (Argyriou et al., 2008) and semi-infinite programming (Özögür-Akyüz and Weber, 2008; Gehler and Nowozin, 2008). Although finding non-linear kernel mixtures (Gönen and Alpaydin, 2008; Varma and Babu, 2009) generally results in non-convex optimization problems, Cortes et al. (2009b) show that convex relaxations may be obtained for special cases.

However, learning arbitrary kernel combinations is a problem too general to allow for a general optimal solution—by focusing on a restricted scenario, it is possible to achieve guaranteed optimality. In their seminal work, Lanckriet et al. (2004) consider training an SVM along with optimizing the linear combination of several positive semi-definite matrices, $K = \sum_{m=1}^{M} \theta_m K_m$, subject to the trace constraint $\mathrm{tr}(K) \leq c$ and requiring a valid combined kernel $K \succeq 0$. This spawned the new field of *multiple kernel learning* (MKL), the automatic combination of several kernel functions. Lanckriet et al. (2004) show that their specific version of the MKL task can be reduced to a convex optimization problem, namely a semi-definite programming (SDP) optimization problem. Though convex, however, the SDP approach is computationally too expensive for practical applications. Thus much of the subsequent research focuses on devising more efficient optimization procedures.

One conceptual milestone for developing MKL into a tool of practical utility is simply to constrain the mixing coefficients θ to be non-negative: by obviating the complex constraint $K \succeq 0$, this small restriction allows to transform the optimization problem into a quadratically constrained program, hence drastically reducing the computational burden. While the original MKL objective is stated and optimized in dual space, alternative formulations have been studied. For instance, Bach et al. (2004) found a corresponding primal problem, and Rubinstein (2005) decomposed the MKL problem into a min-max problem that can be optimized by mirror-prox algorithms (Nemirovski, 2004). The min-max formulation has been independently proposed by Sonnenburg et al. (2005). They use it to recast MKL training as a semi-infinite linear program. Solving the latter with column generation (e.g., Nash and Sofer, 1996) amounts to repeatedly training an SVM on a mixture kernel while iteratively refining the mixture coefficients θ. This immediately lends itself to a convenient implementation by a wrapper approach. These wrapper algorithms directly benefit from efficient SVM optimization routines (cf., Fan et al., 2005; Joachims, 1999) and are now commonly deployed in recent MKL solvers (e.g., Rakotomamonjy et al., 2008; Xu et al., 2009), thereby allowing for large-scale training (Sonnenburg et al., 2005, 2006a). However, the complete training of several

SVMs can still be prohibitive for large data sets. For this reason, Sonnenburg et al. (2005) also propose to interleave the SILP with the SVM training which reduces the training time drastically. Alternative optimization schemes include level-set methods (Xu et al., 2009) and second order approaches (Chapelle and Rakotomamonjy, 2008). Szafranski et al. (2010), Nath et al. (2009), and Bach (2009) study composite and hierarchical kernel learning approaches. Finally, Zien and Ong (2007) and Ji et al. (2009) provide extensions for multi-class and multi-label settings, respectively.

Today, there exist two major families of multiple kernel learning models. The first is characterized by Ivanov regularization (Ivanov et al., 2002) over the mixing coefficients (Rakotomamonjy et al., 2007; Zien and Ong, 2007). For the Tikhonov-regularized optimization problem (Tikhonov and Arsenin, 1977), there is an additional parameter controlling the regularization of the mixing coefficients (Varma and Ray, 2007).

All the above mentioned multiple kernel learning formulations promote *sparse* solutions in terms of the mixing coefficients. The desire for sparse mixtures originates in practical as well as theoretical reasons. First, sparse combinations are easier to interpret. Second, irrelevant (and possibly expensive) kernels functions do not need to be evaluated at testing time. Finally, sparseness appears to be handy also from a technical point of view, as the additional simplex constraint $\|\theta\|_1 \leq 1$ simplifies derivations and turns the problem into a linearly constrained program. Nevertheless, sparseness is not always beneficial in practice and sparse MKL is frequently observed to be outperformed by a regular SVM using an unweighted-sum kernel $K = \sum_m K_m$ (Cortes et al., 2008).

Consequently, despite all the substantial progress in the field of MKL, there still remains an unsatisfied need for an approach that is really useful for practical applications: a model that has a good chance of improving the accuracy (over a plain sum kernel) together with an implementation that matches today's standards (i.e., that can be trained on 10,000s of data points in a reasonable time). In addition, since the field has grown several competing MKL formulations, it seems timely to consolidate the set of models. In this article we argue that all of this is now achievable.

1.1 Outline of the Presented Achievements

On the theoretical side, we cast multiple kernel learning as a general regularized risk minimization problem for arbitrary convex loss functions, Hilbertian regularizers, and arbitrary norm-penalties on θ. We first show that the above mentioned Tikhonov and Ivanov regularized MKL variants are equivalent in the sense that they yield the same set of hypotheses. Then we derive a dual representation and show that a variety of methods are special cases of our objective. Our optimization problem subsumes state-of-the-art approaches to multiple kernel learning, covering sparse and non-sparse MKL by arbitrary p-norm regularization ($1 \leq p \leq \infty$) on the mixing coefficients as well as the incorporation of prior knowledge by allowing for non-isotropic regularizers. As we demonstrate, the p-norm regularization includes both important special cases (sparse 1-norm and plain sum ∞-norm) and offers the potential to elevate predictive accuracy over both of them.

With regard to the implementation, we introduce an appealing and efficient optimization strategy which grounds on an exact update in closed-form in the θ-step; hence rendering expensive semi-infinite and first- or second-order gradient methods unnecessary. By using proven working set optimization for SVMs, p-norm MKL can now be trained highly efficiently for all p; in particular, we outpace other current 1-norm MKL implementations. Moreover our implementation employs kernel caching techniques, which enables training on ten thousands of data points or thousands of kernels respectively. In contrast, most competing MKL software require all kernel matrices

to be stored completely in memory, which restricts these methods to small data sets with limited numbers of kernels. Our implementation is freely available within the SHOGUN machine learning toolbox available at `http://www.shogun-toolbox.org/`. See also our supplementary homepage: `http://doc.ml.tu-berlin.de/nonsparse_mkl/`.

Our claims are backed up by experiments on artificial and real world data sets representing diverse, relevant and challenging problems from the application domain of bioinformatics. Using artificial data, we investigate the impact of the p-norm on the test error as a function of the size of the true sparsity pattern. The real world problems include subcellular localization of proteins, transcription start site detection, and enzyme function prediction. The results demonstrate (i) that combining kernels is now tractable on large data sets, (ii) that it can provide cutting edge classification accuracy, and (iii) that depending on the task at hand, different kernel mixture regularizations are required for achieving optimal performance.

We also present a theoretical analysis of non-sparse MKL. We introduce a novel ℓ_1-to-ℓ_p conversion technique and use it to derive generalization bounds. Based on these, we perform a case study to compare an exemplary sparse with a non-sparse learning scenario. We show that in the sparse scenario $\ell_{p>1}$-norm MKL yields a strictly better generalization bound than ℓ_1-norm MKL, while in the non-sparse scenario it is the other way around.

The remainder is structured as follows. We derive non-sparse MKL in Section 2 and discuss relations to existing approaches in Section 3. Section 4.3 introduces the novel optimization strategy and its implementation. We report on theoretical results in Section 5 and on our empirical results in Section 6. Section 7 concludes.

1.1.1 RELATED WORK

A basic version of this work appeared in NIPS 2009 (Kloft et al., 2009a). The present article additionally offers a more general and complete derivation of the main optimization problem, exemplary applications thereof, a simple algorithm based on a closed-form solution, technical details of the implementation, a theoretical analysis, and additional experimental results. Parts of Section 5 are based on Kloft et al. (2010) the present analysis however extends the previous publication by a novel conversion technique, an illustrative case study, tighter bounds, and an improved presentation.

In related papers, non-sparse MKL has been applied, extended, and further analyzed by several researchers since its initial publication in Kloft et al. (2008), Cortes et al. (2009a), and Kloft et al. (2009a): Varma and Babu (2009) derive a projected gradient-based optimization method for ℓ_2-norm MKL. Yu et al. (2010) present a more general dual view of ℓ_2-norm MKL and show advantages of ℓ_2-norm over an unweighted-sum kernel SVM on six bioinformatics data sets. Cortes et al. (2010a) provide generalization bounds for ℓ_1- and $\ell_{p\leq 2}$-norm MKL. The analytical optimization method presented in this paper was independently and in parallel discovered by Xu et al. (2010) and has also been studied in Roth and Fischer (2007) and Ying et al. (2009) for ℓ_1-norm MKL, and in Szafranski et al. (2010) and Nath et al. (2009) for composite kernel learning on small and medium scales.

2. Multiple Kernel Learning—A Unifying View

In this section we cast multiple kernel learning into a unified framework: we present a regularized loss minimization formulation with additional norm constraints on the kernel mixing coefficients.

We show that it comprises many popular MKL variants currently discussed in the literature, including seemingly different ones.

We derive generalized dual optimization problems without making specific assumptions on the norm regularizers or the loss function, beside that the latter is convex. As a special case we derive ℓ_p-norm MKL in Section 4. In addition, our formulation covers binary classification and regression tasks and can easily be extended to multi-class classification and structural learning settings using appropriate convex loss functions and joint kernel extensions (cf. Section 3). Prior knowledge on kernel mixtures and kernel asymmetries can be incorporated by non-isotropic norm regularizers.

2.1 Preliminaries

We begin with reviewing the classical supervised learning setup. Given a labeled sample $\mathcal{D} = \{(x_i, y_i)\}_{i=1\ldots,n}$, where the x_i lie in some input space $\mathcal{X}$ and $y_i \in \mathcal{Y} \subset \mathbb{R}$, the goal is to find a hypothesis $h \in H$, that generalizes well on new and unseen data. Regularized risk minimization returns a minimizer h^*,

$$h^* \in \operatorname{argmin}_h \mathrm{R}_{\mathrm{emp}}(h) + \lambda\Omega(h),$$

where $\mathrm{R}_{\mathrm{emp}}(h) = \frac{1}{n}\sum_{i=1}^{n} V(h(x_i), y_i)$ is the empirical risk of hypothesis h w.r.t. a convex loss function $V : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$, $\Omega : H \to \mathbb{R}$ is a regularizer, and $\lambda > 0$ is a trade-off parameter. We consider linear models of the form

$$h_{\tilde{w},b}(x) = \langle \tilde{w}, \psi(x) \rangle + b, \tag{1}$$

together with a (possibly non-linear) mapping $\psi : \mathcal{X} \to \mathcal{H}$ to a Hilbert space $\mathcal{H}$ (e.g., Schölkopf et al., 1998; Müller et al., 2001) and constrain the regularization to be of the form $\Omega(h) = \frac{1}{2}\|\tilde{w}\|_2^2$ which allows to kernelize the resulting models and algorithms. We will later make use of kernel functions $k(x,x') = \langle \psi(x), \psi(x') \rangle_{\mathcal{H}}$ to compute inner products in $\mathcal{H}$.

2.2 Regularized Risk Minimization with Multiple Kernels

When learning with multiple kernels, we are given M different feature mappings $\psi_m : \mathcal{X} \to \mathcal{H}_m$, $m = 1,\ldots M$, each giving rise to a reproducing kernel k_m of $\mathcal{H}_m$. Convex approaches to multiple kernel learning consider linear kernel mixtures $k_\theta = \sum \theta_m k_m$, $\theta_m \geq 0$. Compared to Equation (1), the primal model for learning with multiple kernels is extended to

$$h_{\tilde{w},b,\theta}(x) = \sum_{m=1}^{M} \sqrt{\theta_m} \langle \tilde{w}_m, \psi_m(x) \rangle_{\mathcal{H}_m} + b = \langle \tilde{w}, \psi_\theta(x) \rangle_{\mathcal{H}} + b$$

where the parameter vector $\tilde{w}$ and the composite feature map ψ_θ have a block structure $\tilde{w} = (\tilde{w}_1^\top, \ldots, \tilde{w}_M^\top)^\top$ and $\psi_\theta = \sqrt{\theta_1}\psi_1 \times \ldots \times \sqrt{\theta_M}\psi_M$, respectively.

In learning with multiple kernels we aim at minimizing the loss on the training data w.r.t. the optimal kernel mixture $\sum_{m=1}^{M} \theta_m k_m$ in addition to regularizing θ to avoid overfitting. Hence, in terms of regularized risk minimization, the optimization problem becomes

$$\inf_{\tilde{w},b,\theta:\theta\geq\mathbf{0}} \quad \frac{1}{n}\sum_{i=1}^{n} V\left(\sum_{m=1}^{M} \sqrt{\theta_m}\langle \tilde{w}_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b,\ y_i\right) + \frac{\lambda}{2}\sum_{m=1}^{M} \|\tilde{w}_m\|_{\mathcal{H}_m}^2 + \tilde{\mu}\tilde{\Omega}[\theta], \tag{2}$$

for $\tilde{\mu} > 0$. Note that the objective value of Equation (2) is an upper bound on the training error. Previous approaches to multiple kernel learning employ regularizers of the form $\tilde{\Omega}(\theta) = \|\theta\|_1$ to promote sparse kernel mixtures. In contrast, we propose to use convex regularizers of the form $\tilde{\Omega}(\theta) = \|\theta\|^2$, where $\|\cdot\|^2$ is an arbitrary norm in $\mathbb{R}^M$, possibly allowing for non-sparse solutions and the incorporation of prior knowledge. The non-convexity arising from the $\sqrt{\theta_m}\tilde{w}_m$ product in the loss term of Equation (2) is not inherent and can be resolved by substituting $w_m \leftarrow \sqrt{\theta_m}\tilde{w}_m$. Furthermore, the regularization parameter and the sample size can be decoupled by introducing $\tilde{C} = \frac{1}{n\tilde{\lambda}}$ (and adjusting $\mu \leftarrow \frac{\tilde{\mu}}{\tilde{\lambda}}$) which has favorable scaling properties in practice. We obtain the following convex optimization problem (Boyd and Vandenberghe, 2004) that has also been considered by Varma and Ray (2007) for hinge loss and an ℓ_1-norm regularizer

$$\inf_{w,b,\theta:\theta\geq\mathbf{0}} \quad \tilde{C}\sum_{i=1}^{n} V\left(\sum_{m=1}^{M}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b,\ y_i\right) + \frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|^2_{\mathcal{H}_m}}{\theta_m} + \mu\|\theta\|^2, \tag{3}$$

where we use the convention that $\frac{t}{0} = 0$ if $t = 0$ and ∞ otherwise.

An alternative approach has been studied by Rakotomamonjy et al. (2007) and Zien and Ong (2007), again using hinge loss and ℓ_1-norm. They upper bound the value of the regularizer $\|\theta\|_1 \leq 1$ and incorporate the regularizer as an additional constraint into the optimization problem. For $C > 0$ and hinge loss, they arrive at the following problem which is the primary object of investigation in this paper.

2.2.1 GENERAL PRIMAL MKL OPTIMIZATION PROBLEM

$$\begin{aligned} \inf_{w,b,\theta:\theta\geq\mathbf{0}} \quad & C\sum_{i=1}^{n} V\Big(\sum_{m=1}^{M}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b,\ y_i\Big) + \frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|^2_{\mathcal{H}_m}}{\theta_m} \\ \text{s.t.} \quad & \|\theta\|^2 \leq 1. \end{aligned} \tag{4}$$

It is important to note here that, while the Tikhonov regularization in (3) has *two* regularization parameters (C and μ), the above Ivanov regularization (4) has only *one* (C only). Our first contribution shows that, despite the additional regularization parameter, both MKL variants are equivalent, in the sense that traversing the regularization paths yields the same binary classification functions.

Theorem 1 *Let $\|\cdot\|$ be a norm on $\mathbb{R}^M$ and V a convex loss function. Suppose for the optimal w^* in Optimization Problem* (4) *it holds $w^* \neq \mathbf{0}$. Then, for each pair $(\tilde{C},\mu)$ there exists $C > 0$ such that for each optimal solution (w,b,θ) of Equation* (3) *using $(\tilde{C},\mu)$, we have that $(w,b,\kappa\theta)$ is also an optimal solution of Optimization Problem* (4) *using C, and vice versa, where $\kappa > 0$ is a multiplicative constant.*

For the proof we need Prop. 12, which justifies switching from Ivanov to Tikhonov regularization, and back, if the regularizer is tight. We refer to Appendix A for the proposition and its proof.

Proof of Theorem 1 Let be $(\tilde{C},\mu) > 0$. In order to apply Prop. 12 to (3), we show that condition (31) in Prop. 12 is satisfied, that is, that the regularizer is tight.

Suppose on the contrary, that Optimization Problem (4) yields the same infimum regardless of whether we require

$$\|\theta\|^2 \leq 1,$$

or not. Then this implies that in the optimal point we have $\sum_{m=1}^{M} \frac{\|w_m^*\|_2^2}{\theta_m^*} = 0$, hence,

$$\frac{\|w_m^*\|_2^2}{\theta_m^*} = 0, \ \ \forall\, m = 1,\ldots,M. \tag{5}$$

Since all norms on $\mathbb{R}^M$ are equivalent (e.g., Rudin, 1991), there exists a $L < \infty$ such that $\|\theta^*\|_\infty \leq L\|\theta^*\|$. In particular, we have $\|\theta^*\|_\infty < \infty$, from which we conclude by (5), that $w_m = 0$ holds for all m, which contradicts our assumption.

Hence, Prop. 12 can be applied,[1] which yields that (3) is equivalent to

$$\begin{aligned} \inf_{w,b,\theta} \quad & \tilde{C} \sum_{i=1}^{n} V\Big(\sum_{m=1}^{M} \langle w_m, \psi_m(x) \rangle + b,\ y_i \Big) + \frac{1}{2} \sum_{m=1}^{M} \frac{\|w_m\|_2^2}{\theta_m} \\ \text{s.t.} \quad & \|\theta\|^2 \leq \tau, \end{aligned}$$

for some $\tau > 0$. Consider the optimal solution $(w^\star, b^\star, \theta^\star)$ corresponding to a given parametrization $(\tilde{C}, \tau)$. For any $\lambda > 0$, the bijective transformation $(\tilde{C}, \tau) \mapsto (\lambda^{-1/2}\tilde{C}, \lambda\tau)$ will yield $(w^\star, b^\star, \lambda^{1/2}\theta^\star)$ as optimal solution. Applying the transformation with $\lambda := 1/\tau$ and setting $C = \tilde{C}\tau^{\frac{1}{2}}$ as well as $\kappa = \tau^{-1/2}$ yields Optimization Problem (4), which was to be shown. ■

Zien and Ong (2007) also show that the MKL optimization problems by Bach et al. (2004), Sonnenburg et al. (2006a), and their own formulation are equivalent. As a main implication of Theorem 1 and by using the result of Zien and Ong it follows that the optimization problem of Varma and Ray (2007) lies in the same equivalence class as Bach et al. (2004), Sonnenburg et al. (2006a), Rakotomamonjy et al. (2007) and Zien and Ong (2007). In addition, our result shows the coupling between trade-off parameter C and the regularization parameter μ in Equation (3): tweaking one also changes the other and vice versa. Theorem 1 implies that optimizing C in Optimization Problem (4) implicitly searches the regularization path for the parameter μ of Equation (3). In the remainder, we will therefore focus on the formulation in Optimization Problem (4), as a single parameter is preferable in terms of model selection.

2.3 MKL in Dual Space

In this section we study the generalized MKL approach of the previous section in the dual space. Let us begin with rewriting Optimization Problem (4) by expanding the decision values into slack variables as follows

$$\begin{aligned} \inf_{w,b,t,\theta} \quad & C \sum_{i=1}^{n} V\left(t_i,\ y_i\right) + \frac{1}{2} \sum_{m=1}^{M} \frac{\|w_m\|_{\mathcal{H}_m}^2}{\theta_m} \\ \text{s.t.} \quad & \forall i: \ \sum_{m=1}^{M} \langle w_m, \psi_m(x_i) \rangle_{\mathcal{H}_m} + b = t_i \ ; \quad \|\theta\|^2 \leq 1 \ ; \quad \theta \geq \mathbf{0} \ , \end{aligned} \tag{6}$$

where $\|\cdot\|$ is an arbitrary norm in $\mathbb{R}^m$ and $\|\cdot\|_{\mathcal{H}_M}$ denotes the Hilbertian norm of $\mathcal{H}_m$. Applying Lagrange's theorem re-incorporates the constraints into the objective by introducing Lagrangian

1. Note that after a coordinate transformation, we can assume that $\mathcal{H}$ is finite dimensional (see Schölkopf et al., 1999).

multipliers $\alpha \in \mathbb{R}^n$, $\beta \in \mathbb{R}_+$, and $\gamma \in \mathbb{R}^M$. The Lagrangian saddle point problem is then given by

$$\sup_{\substack{\alpha,\beta,\gamma:\\ \beta\geq 0,\gamma\geq \mathbf{0}}} \inf_{w,b,t,\theta} \quad C\sum_{i=1}^{n} V(t_i,\, y_i) + \frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|_{\mathcal{H}_m}^2}{\theta_m} - \sum_{i=1}^{n}\alpha_i\left(\sum_{m=1}^{M}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b - t_i\right) + \beta\left(\frac{1}{2}\|\theta\|^2 - \frac{1}{2}\right) - \gamma^\top\theta.$$

Denoting the Lagrangian by $\mathcal{L}$ and setting its first partial derivatives with respect to w and b to 0 reveals the optimality conditions

$$\mathbf{1}^\top\alpha = 0;$$
$$w_m = \theta_m\sum_{i=1}^{n}\alpha_i\psi_m(x_i),\ \ \forall\, m = 1,\ldots,M.$$

Resubstituting the above equations yields

$$\sup_{\substack{\alpha,\beta,\gamma:\ \mathbf{1}^\top\alpha=0,\\ \beta\geq 0,\gamma\geq\mathbf{0}}} \inf_{t,\theta} \quad C\sum_{i=1}^{n}\left(V(t_i,\, y_i) + \alpha_i t_i\right) - \frac{1}{2}\sum_{m=1}^{M}\theta_m\alpha^\top K_m\alpha + \beta\left(\frac{1}{2}\|\theta\|^2 - \frac{1}{2}\right) - \gamma^\top\theta,$$

which can also be written as

$$\sup_{\substack{\alpha,\beta,\gamma:\ \mathbf{1}^\top\alpha=0,\\ \beta\geq 0,\gamma\geq\mathbf{0}}} -C\sum_{i=1}^{n}\sup_{t_i}\left(-\frac{\alpha_i}{C}t_i - V(t_i,\, y_i)\right) - \beta\sup_{\theta}\left(\frac{1}{\beta}\sum_{m=1}^{M}\left(\frac{1}{2}\alpha^\top K_m\alpha + \gamma_m\right)\theta_m - \frac{1}{2}\|\theta\|^2\right) - \frac{1}{2}\beta.$$

As a consequence, we now may express the Lagrangian as[2]

$$\sup_{\alpha,\beta,\gamma:\ \mathbf{1}^\top\alpha=0,\beta\geq 0,\gamma\geq\mathbf{0}} -C\sum_{i=1}^{n}V^*\left(-\frac{\alpha_i}{C},\, y_i\right) - \frac{1}{\beta}\left\|\left(\frac{1}{2}\alpha^\top K_m\alpha + \gamma_m\right)_{m=1}^{M}\right\|_*^2 - \frac{1}{2}\beta, \tag{7}$$

where $h^*(x) = \sup_u x^\top u - h(u)$ denotes the Fenchel-Legendre conjugate of a function h and $\|\cdot\|_*$ denotes the *dual norm*, that is, the norm defined via the identity $\frac{1}{2}\|\cdot\|_*^2 := \left(\frac{1}{2}\|\cdot\|^2\right)^*$. In the following, we call V^* the *dual loss*. Equation (7) now has to be maximized with respect to the dual variables α, β, subject to $\mathbf{1}^\top\alpha = 0$ and $\beta \geq 0$. Let us ignore for a moment the non-negativity constraint on β and solve $\partial\mathcal{L}/\partial\beta = 0$ for the unbounded β. Setting the partial derivative to zero allows to express the optimal β as

$$\beta = \left\|\left(\frac{1}{2}\alpha^\top K_m\alpha + \gamma_m\right)_{m=1}^{M}\right\|_*. \tag{8}$$

Obviously, at optimality, we always have $\beta \geq 0$. We thus discard the corresponding constraint from the optimization problem and plugging Equation (8) into Equation (7) results in the following *dual* optimization problem:

2. We employ the notation $s = (s_1,\ldots,s_M)^\top = (s_m)_{m=1}^M$ for $s \in \mathbb{R}^M$.

2.3.1 GENERAL DUAL MKL OPTIMIZATION PROBLEM

$$\sup_{\alpha,\gamma:\ \mathbf{1}^\top\alpha=0,\gamma\geq\mathbf{0}} \quad -C\sum_{i=1}^{n} V^*\left(-\frac{\alpha_i}{C},\, y_i\right) - \left\|\left(\frac{1}{2}\alpha^\top K_m\alpha+\gamma_m\right)_{m=1}^{M}\right\|_*. \tag{9}$$

The above dual generalizes multiple kernel learning to arbitrary convex loss functions and norms.[3] Note that for the most common choices of norms (for example, ℓ_p-norm, weighted ℓ_p-norms, and sum of ℓ_p-norms; but not the norms discussed in Section 3.5) it holds $\gamma^* = 0$ in the optimal point so that the γ-term can be discarded and the above reduces to an optimization problem that solely depends on α. Also note that if the loss function is continuous (e.g., hinge loss), the supremum is also a maximum. The threshold b can be recovered from the solution by applying the KKT conditions.

The above dual can be characterized as follows. We start by noting that the expression in Optimization Problem (9) is a composition of two terms, first, the left hand side term, which depends on the conjugate loss function V^*, and, second, the right hand side term which depends on the conjugate norm. The right hand side can be interpreted as a regularizer on the quadratic terms that, according to the chosen norm, smoothens the solutions. Hence we have a decomposition of the dual into a loss term (in terms of the dual loss) and a regularizer (in terms of the dual norm). For a specific choice of a pair $(V, \|\cdot\|)$ we can immediately recover the corresponding dual by computing the pair of conjugates $(V^*, \|\cdot\|_*)$ (for a comprehensive list of dual losses see Rifkin and Lippert, 2007, Table 3). In the next section, this is illustrated by means of well-known loss functions and regularizers.

At this point we would like to highlight some properties of Optimization Problem (9) that arise due to our dualization technique. While approaches that firstly apply the representer theorem and secondly optimize in the primal such as Chapelle (2006) also can employ general loss functions, the resulting loss terms depend on all optimization variables. By contrast, in our formulation the dual loss terms are of a much simpler structure and they only depend on a single optimization variable α_i. A similar dualization technique yielding singly-valued dual loss terms is presented in Rifkin and Lippert (2007); it is based on Fenchel duality and limited to strictly positive definite kernel matrices. Our technique, which uses Lagrangian duality, extends the latter by allowing for positive semi-definite kernel matrices.

3. Recovering Previous MKL Formulations as Special Instances

In this section we show that existing MKL-based learners are subsumed by the generalized formulation in Optimization Problem (9). It is helpful for what is coming up to note that for most (but not all; see Section 3.5) choices of norms it holds $\gamma^* = 0$ in the generalized dual MKL problem (9), so that it simplifies to:

$$\sup_{\alpha:\ \mathbf{1}^\top\alpha=0} \quad -C\sum_{i=1}^{n} V^*\left(-\frac{\alpha_i}{C},\, y_i\right) - \frac{1}{2}\left\|\left(\alpha^\top K_m\alpha\right)_{m=1}^{M}\right\|_*. \tag{10}$$

3. We can even employ non-convex losses and still the dual will be a convex problem; however, it might suffer from a duality gap.

3.1 Support Vector Machines with Unweighted-Sum Kernels

First, we note that the support vector machine with an unweighted-sum kernel can be recovered as a special case of our model. To see this, we consider the regularized risk minimization problem using the hinge loss function $V(t,y) = \max(0, 1-ty)$ and the regularizer $\|\theta\|_\infty$. We then can obtain the corresponding dual in terms of Fenchel-Legendre conjugate functions as follows.

We first note that the dual loss of the hinge loss is $V^*(t,y) = \frac{t}{y}$ if $-1 \leq \frac{t}{y} \leq 0$ and ∞ elsewise (Rifkin and Lippert, 2007, Table 3). Hence, for each i the term $V^*\left(-\frac{\alpha_i}{C},\, y_i\right)$ of the generalized dual, that is, Optimization Problem (9), translates to $-\frac{\alpha_i}{Cy_i}$, provided that $0 \leq \frac{\alpha_i}{y_i} \leq C$. Employing a variable substitution of the form $\alpha_i^{\text{new}} = \frac{\alpha_i}{y_i}$, Optimization Problem (9) translates to

$$\max_{\alpha,\gamma:\ \gamma\geq 0} \quad \mathbf{1}^\top \alpha - \left\| \left(\frac{1}{2}\alpha^\top Y K_m Y \alpha + \gamma_m \right)_{m=1}^{M} \right\|_*, \quad \text{s.t.} \quad y^\top \alpha = 0 \ \text{ and } \ \mathbf{0} \leq \alpha \leq C\mathbf{1}, \tag{11}$$

where we denote $Y = \operatorname{diag}(y)$. The primal ℓ_∞-norm penalty $\|\theta\|_\infty$ is dual to $\|\theta\|_1$, hence, via the identity $\|\cdot\|_* = \|\cdot\|_1$ the right hand side of the last equation translates to $\sum_{m=1}^M \alpha^\top Y K_m Y \alpha$, and we note that $\gamma^* = 0$ in the optimal point. Combined with (11) this leads to the dual

$$\max_{\alpha} \quad \mathbf{1}^\top \alpha - \sum_{m=1}^{M} \alpha^\top Y K_m Y \alpha, \quad \text{s.t.} \quad y^\top \alpha = 0 \ \text{ and } \ \mathbf{0} \leq \alpha \leq C\mathbf{1},$$

which is precisely an SVM with an unweighted-sum kernel.

3.2 QCQP MKL of Lanckriet et al. (2004)

A common approach in multiple kernel learning is to employ regularizers of the form

$$\Omega(\theta) = \|\theta\|_1. \tag{12}$$

This so-called ℓ_1-norm regularizers are specific instances of *sparsity-inducing* regularizers. The obtained kernel mixtures usually have a considerably large fraction of zero entries, and hence equip the MKL problem by the favor of interpretable solutions. Sparse MKL is a special case of our framework; to see this, note that the conjugate of (12) is $\|\cdot\|_\infty$. Recalling the definition of an ℓ_p-norm, the right hand side of Optimization Problem (9) translates to $\max_{m\in\{1,\ldots,M\}} \alpha^\top Y K_m Y \alpha$. The maximum can subsequently be expanded into a slack variable ξ, resulting in

$$\begin{aligned} &\sup_{\alpha,\xi} \quad \mathbf{1}^\top \alpha - \xi \\ &\text{s.t.} \quad \forall\, m: \ \frac{1}{2}\alpha^\top Y K_m Y \alpha \leq \xi\,; \quad y^\top \alpha = 0\,; \quad \mathbf{0} \leq \alpha \leq C\mathbf{1}, \end{aligned}$$

which is the original QCQP formulation of MKL, firstly given by Lanckriet et al. (2004).

3.3 A Smooth Variant of Group Lasso

Yuan and Lin (2006) studied the following optimization problem for the special case $\mathcal{H}_m = \mathbb{R}^{d_m}$ and $\psi_m = \mathrm{id}_{\mathbb{R}^{d_m}}$, also known as group lasso,

$$\min_{w} \quad \frac{C}{2}\sum_{i=1}^{n}\left(y_i - \sum_{m=1}^{M} \langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} \right)^2 + \frac{1}{2}\sum_{m=1}^{M} \|w_m\|_{\mathcal{H}_m}. \tag{13}$$

The above problem has been solved by active set methods in the primal (Roth and Fischer, 2008). We sketch an alternative approach based on dual optimization. First, we note that Equation (13) can be equivalently expressed as (Micchelli and Pontil, 2005, Lemma 26)

$$\inf_{w,\theta:\theta\geq\mathbf{0}} \quad \frac{C}{2}\sum_{i=1}^{n}\left(y_i - \sum_{m=1}^{M}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m}\right)^2 + \frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|_{\mathcal{H}_m}^2}{\theta_m}, \quad \text{s.t.} \quad \|\theta\|_1^2 \leq 1.$$

The dual of $V(t,y) = \frac{1}{2}(y-t)^2$ is $V^*(t,y) = \frac{1}{2}t^2 + ty$ and thus the corresponding group lasso dual can be written as

$$\max_{\alpha} \quad y^\top\alpha - \frac{1}{2C}\|\alpha\|_2^2 - \frac{1}{2}\left\|\left(\alpha^\top Y K_m Y \alpha\right)_{m=1}^{M}\right\|_\infty,$$

which can be expanded into the following QCQP

$$\begin{aligned}\sup_{\alpha,\xi} \quad & y^\top\alpha - \frac{1}{2C}\|\alpha\|_2^2 - \xi \\ \text{s.t.} \quad \forall m: \quad & \frac{1}{2}\alpha^\top Y K_m Y \alpha \leq \xi.\end{aligned}$$

For small n, the latter formulation can be handled efficiently by QCQP solvers. However, the quadratic constraints caused by the non-smooth ℓ_∞-norm in the objective still are computationally too demanding. As a remedy, we propose the following unconstrained variant based on ℓ_p-norms ($1 < p < \infty$), given by

$$\max_{\alpha} \quad y^\top\alpha - \frac{1}{2C}\|\alpha\|_2^2 - \frac{1}{2}\left\|\left(\alpha^\top Y K_m Y \alpha\right)_{m=1}^{M}\right\|_{p^*}.$$

It is straightforward to verify that the above objective function is differentiable in any $\alpha \in \mathbb{R}^n$ (in particular, notice that the ℓ_p-norm function is differentiable for $1 < p < \infty$) and hence the above optimization problem can be solved very efficiently by, for example, limited memory quasi-Newton descent methods (Liu and Nocedal, 1989).

3.4 Density Level-Set Estimation

Density level-set estimators are frequently used for anomaly/novelty detection tasks (Markou and Singh, 2003a,b). Kernel approaches, such as one-class SVMs (Schölkopf et al., 2001) and Support Vector Domain Descriptions (Tax and Duin, 1999) can be cast into our MKL framework by employing loss functions of the form $V(t) = \max(0, 1-t)$. This gives rise to the primal

$$\inf_{w,\theta:\theta\geq\mathbf{0}} \quad C\sum_{i=1}^{n}\max\left(0, \sum_{m=1}^{M}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m}\right) + \frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|_{\mathcal{H}_m}^2}{\theta_m}, \quad \text{s.t.} \quad \|\theta\|^2 \leq 1.$$

Noting that the dual loss is $V^*(t) = t$ if $-1 \leq t \leq 0$ and ∞ elsewise, we obtain the following generalized dual

$$\sup_{\alpha} \quad \mathbf{1}^\top\alpha - \frac{1}{2}\left\|\left(\alpha^\top K_m \alpha\right)_{m=1}^{M}\right\|_{p^*}, \quad \text{s.t.} \quad \mathbf{0} \leq \alpha \leq C\mathbf{1},$$

which has been studied by Sonnenburg et al. (2006a) and Rakotomamonjy et al. (2008) for ℓ_1-norm, and by Kloft et al. (2009b) for ℓ_p-norms.

3.5 Non-Isotropic Norms

In practice, it is often desirable for an expert to incorporate prior knowledge about the problem domain. For instance, an expert could provide estimates of the interactions of kernels $\{K_1,...,K_M\}$ in the form of an $M \times M$ matrix E. Alternatively, E could be obtained by computing pairwise kernel alignments $E_{ij} = \frac{<K_i,K_j>}{\|K_i\| \, \|K_j\|}$ given a dot product on the space of kernels such as the Frobenius dot product (Ong et al., 2005). In a third scenario, E could be a diagonal matrix encoding the a priori importance of kernels—it might be known from pilot studies that a subset of the employed kernels is inferior to the remaining ones.

All those scenarios can be handled within our framework by considering non-isotropic regularizers of the form[4]

$$\|\theta\|_{E^{-1}} = \sqrt{\theta^\top E^{-1}\theta} \;\text{ with }\; E \succ 0,$$

where E^{-1} is the matrix inverse of E.

However, this choice of a norm is quite different from what we have seen before: let us consider Optimization Problem (9); for non-isotropic norms we in general do not have $\gamma^* = 0$ in the optimal point so that this OP does not simplify to the dual (10) as in the subsections before. Instead we have to work with (9) directly. To this end, note that for the dual norm it holds $\left(\frac{1}{2}\|\cdot\|^2_{E^{-1}}\right)^* = \frac{1}{2}\|\cdot\|^2_E$, so that we obtain from (9) the following dual

$$\sup_{\alpha,\gamma:\, \mathbf{1}^\top\alpha=\mathbf{0},\gamma\geq\mathbf{0}} \; -C\sum_{i=1}^n V^*\left(-\frac{\alpha_i}{C},\, y_i\right) - \left\|\left(\frac{1}{2}\alpha^\top K_m\alpha + \gamma_m\right)_{m=1}^M\right\|_E,$$

which is the desired non-isotropic MKL problem.

4. ℓ_p-Norm Multiple Kernel Learning

In this work, we propose to use non-sparse and thus more robust kernel mixtures by employing an ℓ_p-norm constraint with $p > 1$, rather than the traditionally used ℓ_1-norm constraint, on the mixing coefficients (Kloft et al., 2009a). To this end, we employ non-sparse norms of the form $\|\theta\|_p = (\sum_{m=1}^M \theta_m^p)^{1/p}$, $1 < p < \infty$.[5] From the unifying framework of Section 2 we obtain the following ℓ_p-norm MKL primal:

4.1 Primal ℓ_p-norm MKL Optimization Problem

$$\begin{aligned} \inf_{w,b,\theta:\theta\geq\mathbf{0}} \quad & C\sum_{i=1}^n V\left(\sum_{m=1}^M \langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b,\, y_i\right) + \frac{1}{2}\sum_{m=1}^M \frac{\|w_m\|^2_{\mathcal{H}_m}}{\theta_m} \\ \text{s.t.} \quad & \|\theta\|_p^2 \leq 1. \end{aligned} \tag{14}$$

Using that the dual norm of the ℓ_p-norm is the ℓ_{p^*}-norm, where $p^* := \frac{p}{p-1}$, and noting that $\gamma^* = 0$ in the optimal point, we obtain from Optimization Problem (9) the following ℓ_p-norm MKL dual:

4. This idea is inspired by the Mahalanobis distance (Mahalanobis, 1936).

5. While the upcoming reasoning also holds for weighted ℓ_p-norms, the extension to more general norms, such as the ones described in Section 3.5, is left for future work.

4.2 Dual ℓ_p-norm MKL Optimization Problem

$$\sup_{\alpha:\mathbf{1}^\top\alpha=\mathbf{0}} \; -C\sum_{i=1}^{n} V^*\left(-\frac{\alpha_i}{C},\, y_i\right) - \frac{1}{2}\left\|\left(\alpha^\top K_m \alpha\right)_{m=1}^{M}\right\|_{p^*}.$$

In the special case of hinge loss minimization, we obtain the optimization problem

$$\sup_{\alpha} \; \mathbf{1}^\top\alpha - \frac{1}{2}\left\|\left(\alpha^\top Y K_m Y \alpha\right)_{m=1}^{M}\right\|_{p^*}, \quad \text{s.t.} \quad y^\top\alpha = 0 \;\text{ and }\; \mathbf{0} \le \alpha \le C\mathbf{1}. \tag{15}$$

In the subsequent sections, we will propose an efficient optimization algorithm for Optimization Problem (15) (Section 4.3) and proof its convergence (Section 4.3.3). Later we derive generalization bounds (Section 5), and analyze ℓ_p-norm MKL empirically using artificial and real-world data sets (Section 6).

4.3 Optimization Strategies

The dual as given in Optimization Problem (15) does not lend itself to efficient large-scale optimization in a straight-forward fashion, for instance by direct application of standard approaches like gradient descent. Instead, it is beneficial to exploit the structure of the MKL cost function by alternating between optimizing w.r.t. the mixings θ and w.r.t. the remaining variables. Most recent MKL solvers (e.g., Rakotomamonjy et al., 2008; Xu et al., 2009; Nath et al., 2009) do so by setting up a two-layer optimization procedure: a master problem, which is parameterized only by θ, is solved to determine the kernel mixture; to solve this master problem, repeatedly a slave problem is solved which amounts to training a standard SVM on a mixture kernel. Importantly, for the slave problem, the mixture coefficients are fixed, such that conventional, efficient SVM optimizers can be recycled. Consequently these two-layer procedures are commonly implemented as *wrapper* approaches. Albeit appearing advantageous, wrapper methods suffer from two shortcomings: (i) Due to kernel cache limitations, the kernel matrices have to be pre-computed and stored or many kernel computations have to be carried out repeatedly, inducing heavy wastage of either memory or time. (ii) The slave problem is always optimized to the end (and many convergence proofs seem to require this), although most of the computational time is spend on the non-optimal mixtures. Certainly suboptimal slave solutions would already suffice to improve far-from-optimal θ in the master problem.

Due to these problems, MKL is prohibitive when learning with a multitude of kernels and on large-scale data sets as commonly encountered in many data-intense real world applications such as bioinformatics, web mining, databases, and computer security. The optimization approach presented in this paper decomposes the MKL problem into smaller subproblems (Platt, 1999; Joachims, 1999; Fan et al., 2005) by establishing a wrapper-like scheme *within* the decomposition algorithm.

Our algorithm is embedded into the large-scale framework of Sonnenburg et al. (2006a) and extends it to the optimization of non-sparse kernel mixtures induced by an ℓ_p-norm penalty. Our strategy alternates between minimizing the primal problem (6) w.r.t. θ via a simple analytical update formula and with incomplete optimization w.r.t. all other variables which, however, is performed in terms of the dual variables α. Optimization w.r.t. α is performed by chunking optimizations with minor iterations. Convergence of our algorithm is proven under typical technical regularity assumptions.

4.3.1 A SIMPLE WRAPPER APPROACH BASED ON AN ANALYTICAL UPDATE

We first present an easy-to-implement wrapper version of our optimization approach to multiple kernel learning. The interleaved decomposition algorithm is deferred to the next section.

To derive the new algorithm, we divide the optimization variables of the primal problem (14) into two groups, (w,b) on one hand and θ on the other. Our algorithm will alternatingly operate on those two groups via a block coordinate descent algorithm, also known as the *non-linear block Gauss-Seidel method*. Thereby the optimization w.r.t. θ will be carried out analytically and the (w,b)-step will be computed in the dual, if needed.

The basic idea of our first approach is that for a given, fixed set of primal variables (w,b), the optimal θ in the primal problem (14) can be calculated analytically as the following proposition shows.

Proposition 2 *Let V be a convex loss function, be $p>1$. Given fixed (possibly suboptimal) $w\neq\mathbf{0}$ and b, the minimal θ in Optimization Problem* (14) *is attained for*

$$\theta_m = \frac{\|w_m\|_{\mathcal{H}_m}^{\frac{2}{p+1}}}{\left(\sum_{m'=1}^{M}\|w_{m'}\|_{\mathcal{H}_{m'}}^{\frac{2p}{p+1}}\right)^{1/p}}, \qquad \forall m=1,\ldots,M. \tag{16}$$

Proof [6] We start the derivation, by equivalently translating Optimization Problem (14) via Theorem 1 into

$$\inf_{w,b,\theta:\theta\geq\mathbf{0}} \quad \tilde{C}\sum_{i=1}^{n} V\left(\sum_{m=1}^{M}\langle w_m,\psi_m(x_i)\rangle_{\mathcal{H}_m}+b,\ y_i\right)+\frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|_{\mathcal{H}_m}^2}{\theta_m}+\frac{\mu}{2}\|\theta\|_p^2, \tag{17}$$

with $\mu>0$. Suppose we are given fixed (w,b), then setting the partial derivatives of the above objective w.r.t. θ to zero yields the following condition on the optimality of θ,

$$-\frac{\|w_m\|_{\mathcal{H}_m}^2}{2\theta_m^2}+\mu\cdot\frac{\partial\left(\frac{1}{2}\|\theta\|_p^2\right)}{\partial\theta_m}=0, \quad \forall m=1,\ldots,M. \tag{18}$$

The first derivative of the ℓ_p-norm with respect to the mixing coefficients can be expressed as

$$\frac{\partial\left(\frac{1}{2}\|\theta\|_p^2\right)}{\partial\theta_m}=\theta_m^{p-1}\|\theta\|_p^{2-p},$$

and hence Equation (18) translates into the following optimality condition,

$$\exists\zeta \quad \forall m=1,\ldots,M: \qquad \theta_m=\zeta\|w_m\|_{\mathcal{H}_m}^{\frac{2}{p+1}}. \tag{19}$$

Because $w\neq 0$, using the same argument as in the proof of Theorem 1, the constraint $\|\theta\|_p^2\leq 1$ in (17) is at the upper bound, that is, $\|\theta\|_p=1$ holds for an optimal θ. Inserting (19) in the latter equation leads to $\zeta=\left(\sum_{m=1}^{M}\|w_m\|_{\mathcal{H}_m}^{2p/p+1}\right)^{1/p}$. Resubstitution into (19) yields the claimed formula

6. We remark that a more general result can be obtained by an alternative proof using Hölder's inequality (see Lemma 26 in Micchelli and Pontil, 2005).

(16). ■

Second, we consider how to optimize Optimization Problem (14) w.r.t. the remaining variables (w,b) for a given set of mixing coefficients θ. Since optimization often is considerably easier in the dual space, we fix θ and build the partial Lagrangian of Optimization Problem (14) w.r.t. all other primal variables w, b. The resulting dual problem is of the form (detailed derivations omitted)

$$\sup_{\alpha:\mathbf{1}^\top\alpha=0} -C\sum_{i=1}^{n} V^*\left(-\frac{\alpha_i}{C},\, y_i\right) - \frac{1}{2}\sum_{m=1}^{M}\theta_m\alpha^\top K_m\alpha, \tag{20}$$

and the KKT conditions yield $w_m = \theta_m\sum_{i=1}^{n}\alpha_i\psi_m(x_i)$ in the optimal point, hence

$$\|w_m\|^2 = \theta_m^2\alpha K_m\alpha, \quad \forall\, m=1,\ldots,M. \tag{21}$$

We now have all ingredients (i.e., Equations (16), (20)–(21)) to formulate a simple macro-wrapper algorithm for ℓ_p-norm MKL training:

Algorithm 1 Simple $\ell_{p>1}$-norm MKL wrapper-based training algorithm. The analytical updates of θ and the SVM computations are optimized alternatingly.

1: **input:** feasible α and θ
2: **while** optimality conditions are not satisfied **do**
3: Compute α according to Equation (20) (e.g., SVM)
4: Compute $\|w_m\|^2$ for all $m=1,...,M$ according to Equation (21)
5: Update θ according to Equation (16)
6: **end while**

The above algorithm alternatingly solves a convex risk minimization machine (e.g., SVM) w.r.t. the actual mixture θ (Equation (20)) and subsequently computes the analytical update according to Equation (16) and (21). It can, for example, be stopped based on changes of the objective function or the duality gap within subsequent iterations.

4.3.2 TOWARDS LARGE-SCALE MKL—INTERLEAVING SVM AND MKL OPTIMIZATION

However, a disadvantage of the above wrapper approach still is that it deploys a full blown kernel matrix. We thus propose to interleave the SVM optimization of SVMlight with the θ- and α-steps at training time. We have implemented this so-called *interleaved* algorithm in Shogun for hinge loss, thereby promoting sparse solutions in α. This allows us to solely operate on a small number of active variables.[7] The resulting interleaved optimization method is shown in Algorithm 2. Lines 3-5 are standard in chunking based SVM solvers and carried out by SVM$^{\text{light}}$ (note that Q is chosen as described in Joachims, 1999). Lines 6-7 compute SVM-objective values. Finally, the analytical θ-step is carried out in Line 9. The algorithm terminates if the maximal KKT violation (cf. Joachims, 1999) falls below a predetermined precision ε and if the normalized maximal constraint violation $|1-\frac{\omega}{\omega_{old}}| < \varepsilon_{mkl}$ for the MKL-step, where ω denotes the MKL objective function value (Line 8).

7. In practice, it turns out that the kernel matrix of active variables typically is about of the size 40×40, even when we deal with ten-thousands of examples.

Algorithm 2 ℓ_p-Norm MKL chunking-based training algorithm via analytical update. Kernel weighting θ and (signed) SVM α are optimized interleavingly. The accuracy parameter ε and the subproblem size Q are assumed to be given to the algorithm.

Initialize: $g_{m,i} = \hat{g}_i = \alpha_i = 0, \forall i = 1,...,n; \quad L = S = -\infty; \quad \theta_m = \sqrt[p]{1/M}, \forall m = 1,...,M$
iterate
  Select Q variables $\alpha_{i_1}, \ldots, \alpha_{i_Q}$ based on the gradient $\hat{\mathbf{g}}$ of (20) w.r.t. α
  Store $\alpha^{old} = \alpha$ and then update α according to (20) with respect to the selected variables
  Update gradient $g_{m,i} \leftarrow g_{m,i} + \sum_{q=1}^{Q} (\alpha_{i_q} - \alpha_{i_q}^{old}) k_m(x_{i_q}, x_i), \ \forall\, m = 1, \ldots, M, i = 1, \ldots, n$
  Compute the quadratic terms $S_m = \frac{1}{2} \sum_i g_{m,i} \alpha_i, \ q_m = 2\theta_m^2 S_m, \ \forall m = 1, \ldots, M$
  $L_{old} = L, \quad L = \sum_i y_i \alpha_i, \quad S_{old} = S, \quad S = \sum_m \theta_m S_m$
  if $|1 - \frac{L-S}{L_{old}-S_{old}}| \geq \varepsilon$
    $\theta_m = (q_m)^{1/(p+1)} \Big/ \left(\sum_{m'=1}^{M} (q_{m'})^{p/(p+1)} \right)^{1/p}, \quad \forall\, m = 1, \ldots, M$
  else
    break
  end if
  $\hat{g}_i = \sum_m \theta_m g_{m,i}$ for all $i = 1, \ldots, n$

4.3.3 CONVERGENCE PROOF FOR $p > 1$

In the following, we exploit the primal view of the above algorithm as a nonlinear block Gauss-Seidel method, to prove convergence of our algorithms. We first need the following useful result about convergence of the nonlinear block Gauss-Seidel method in general.

Proposition 3 (Bertsekas, 1999, Prop. 2.7.1) *Let $\mathcal{X} = \bigotimes_{m=1}^{M} \mathcal{X}_m$ be the Cartesian product of closed convex sets $\mathcal{X}_m \subset \mathbb{R}^{d_m}$, be $f : \mathcal{X} \to \mathbb{R}$ a continuously differentiable function. Define the nonlinear block Gauss-Seidel method recursively by letting $x^0 \in \mathcal{X}$ be any feasible point, and be*

$$x_m^{k+1} = \operatorname*{argmin}_{\xi \in \mathcal{X}_m} f\left(x_1^{k+1}, \cdots, x_{m-1}^{k+1}, \xi, x_{m+1}^{k}, \cdots, x_M^{k}\right), \ \ \forall m = 1, \ldots, M. \tag{22}$$

Suppose that for each m and $x \in \mathcal{X}$, the minimum

$$\min_{\xi \in \mathcal{X}_m} f\left(x_1, \cdots, x_{m-1}, \xi, x_{m+1}, \cdots, x_M\right)$$

is uniquely attained. Then every limit point of the sequence $\{x^k\}_{k \in \mathbb{N}}$ is a stationary point.

The proof can be found in Bertsekas (1999), p. 268-269. The next proposition basically establishes convergence of the proposed ℓ_p-norm MKL training algorithm.

Theorem 4 *Let V be the hinge loss and be $p > 1$. Let the kernel matrices $K_1, \ldots, K_M$ be positive definite. Then every limit point of Algorithm 1 is a globally optimal point of Optimization Problem* (14). *Moreover, suppose that the SVM computation is solved exactly in each iteration, then the same holds true for Algorithm 2.*

Proof If we ignore the numerical speed-ups, then the Algorithms 1 and 2 coincide for the hinge loss. Hence, it suffices to show the wrapper algorithm converges.

To this aim, we have to transform Optimization Problem (14) into a form such that the requirements for application of Prop. 3 are fulfilled. We start by expanding Optimization Problem (14) into

$$\begin{aligned}\min_{w,b,\xi,\theta} \quad & C\sum_{i=1}^{n}\xi_i+\frac{1}{2}\sum_{m=1}^{M}\frac{\|w_m\|_{\mathcal{H}_m}^2}{\theta_m},\\ \text{s.t.} \quad & \forall i:\ \sum_{m=1}^{M}\langle w_m,\psi_m(x_i)\rangle_{\mathcal{H}_m}+b\geq 1-\xi_i;\quad \xi\geq 0;\quad \|\theta\|_p^2\leq 1;\quad \theta\geq\mathbf{0},\end{aligned}$$

thereby extending the second block of variables, (w,b), into (w,b,ξ). Moreover, we note that after an application of the representer theorem[8] (Kimeldorf and Wahba, 1971) we may without loss of generality assume $\mathcal{H}_m=\mathbb{R}^n$.

In the problem's current form, the possibility of an optimal $\theta_m=0$ while $w_m\neq 0$ renders the objective function nondifferentiable. This hinders the application of Prop. 3. Fortunately, it follows from Prop. 2 (note that $K_m\succ 0$ implies $w\neq\mathbf{0}$) that this case is impossible for $p>1$. We therefore can substitute the constraint $\theta\geq\mathbf{0}$ by $\theta>\mathbf{0}$ for all m without changing the optimum. In order to maintain the closeness of the feasible set we subsequently apply a bijective coordinate transformation $\phi:\mathbb{R}_+^M\to\mathbb{R}^M$ with $\theta_m^{\text{new}}=\phi_m(\theta_m)=\log(\theta_m)$, resulting in the following equivalent problem,

$$\begin{aligned}\inf_{w,b,\xi,\theta} \quad & C\sum_{i=1}^{n}\xi_i+\frac{1}{2}\sum_{m=1}^{M}\exp(-\theta_m)\|w_m\|_{\mathbb{R}^n}^2,\\ \text{s.t.} \quad & \forall i:\ \sum_{m=1}^{M}\langle w_m,\psi_m(x_i)\rangle_{\mathbb{R}^n}+b\geq 1-\xi_i;\quad \xi\geq 0;\quad \|\exp(\theta)\|_p^2\leq 1,\end{aligned}$$

where we employ the notation $\exp(\theta)=(\exp(\theta_1),\cdots,\exp(\theta_M))^\top$.

Applying the Gauss-Seidel method in Equation (22) to the base problem Optimization Problem (14) and to the reparametrized problem yields the same sequence of solutions $\{(w,b,\theta)^k\}_{k\in\mathbb{N}_0}$. The above problem now allows to apply Prop. 3 for the two blocks of coordinates $\theta\in\mathcal{X}_1$ and $(w,b,\xi)\in\mathcal{X}_2$: the objective is continuously differentiable and the sets $\mathcal{X}_1$ and $\mathcal{X}_2$ are closed and convex. To see the latter, note that $\|\cdot\|_p^2\circ\exp$ is a convex function (cf., Section 3.2.4 in Boyd and Vandenberghe, 2004). Moreover, the minima in Equation (22) are uniquely attained: the (w,b)-step amounts to solving an SVM on a positive definite kernel mixture, and the analytical θ-step clearly yields unique solutions as well.

Hence, we conclude that every limit point of the sequence $\{(w,b,\theta)^k\}_{k\in\mathbb{N}}$ is a stationary point of Optimization Problem (14). For a convex problem, this is equivalent to such a limit point being globally optimal. ■

In practice, we are facing two problems. First, the standard Hilbert space setup necessarily implies that $\|w_m\|\geq 0$ for all m. However in practice this assumption may often be violated, either due to numerical imprecision or because of using an indefinite "kernel" function. However, for any $\|w_m\|\leq 0$ it also follows that $\theta_m^\star=0$ as long as at least one strictly positive $\|w_{m'}\|>0$ exists. This is because for any $\lambda<0$ we have $\lim_{h\to 0,h>0}\frac{\lambda}{h}=-\infty$. Thus, for any m with $\|w_m\|\leq 0$, we

8. Note that the coordinate transformation into $\mathbb{R}^n$ can be explicitly given in terms of the empirical kernel map (Schölkopf et al., 1999).

can immediately set the corresponding mixing coefficients $\theta_m^\star$ to zero. The remaining θ are then computed according to Equation (2), and convergence will be achieved as long as at least one strictly positive $\|w_{m'}\| > 0$ exists in each iteration.

Second, in practice, the SVM problem will only be solved with finite precision, which may lead to convergence problems. Moreover, we actually want to improve the α only a little bit before recomputing θ since computing a high precision solution can be wasteful, as indicated by the superior performance of the interleaved algorithms (cf. Sect. 6.5). This helps to avoid spending a lot of α-optimization (SVM training) on a suboptimal mixture θ. Fortunately, we can overcome the potential convergence problem by ensuring that the primal objective decreases within each α-step. This is enforced in practice, by computing the SVM by a higher precision if needed. However, in our computational experiments we find that this precaution is not even necessary: even without it, the algorithm converges in all cases that we tried (cf. Section 6).

Finally, we would like to point out that the proposed block coordinate descent approach lends itself more naturally to combination with primal SVM optimizers like Chapelle (2006), LibLinear (Fan et al., 2008) or Ocas (Franc and Sonnenburg, 2008). Especially for linear kernels this is extremely appealing.

4.4 Technical Considerations

In this section we report on implementation details and discuss kernel normalization.

4.4.1 IMPLEMENTATION DETAILS

We have implemented the analytic optimization algorithm described in the previous Section, as well as the cutting plane and Newton algorithms by Kloft et al. (2009a), within the SHOGUN toolbox (Sonnenburg et al., 2010) for regression, one-class classification, and two-class classification tasks. In addition one can choose the optimization scheme, that is, decide whether the interleaved optimization algorithm or the wrapper algorithm should be applied. In all approaches any of the SVMs contained in SHOGUN can be used. Our implementation can be downloaded from `http://www.shogun-toolbox.org`.

In the more conventional family of approaches, the *wrapper algorithms*, an optimization scheme on θ wraps around a single kernel SVM. Effectively this results in alternatingly solving for α and θ. For the outer optimization (i.e., that on θ) SHOGUN offers the three choices listed above. The semi-infinite program (SIP) uses a traditional SVM to generate new violated constraints and thus requires a single kernel SVM. A linear program (for $p = 1$) or a sequence of quadratically constrained linear programs (for $p > 1$) is solved via GLPK[9] or IBM ILOG CPLEX[10]. Alternatively, either an analytic or a Newton update (for ℓ_p norms with $p > 1$) step can be performed, obviating the need for an additional mathematical programming software.

The second, much faster approach performs interleaved optimization and thus requires modification of the core SVM optimization algorithm. It is currently integrated into the chunking-based SVRlight and SVMlight. To reduce the implementation effort, we implement a single function `perform_mkl_step(`$\sum\alpha$`,` obj_m`)`, that has the arguments $\sum\alpha := \sum_{i=1}^n \alpha_i$ and $\text{obj}_m = \frac{1}{2}\alpha^T K_m \alpha$, that is, the current linear α-term and the SVM objectives for each kernel. This function is either, in the

9. GLPK can be found at `http://www.gnu.org/software/glpk/`.
10. ILOG CPLEX can be found at `http://www.ibm.com/software/integration/optimization/cplex/`.

interleaved optimization case, called as a callback function (after each chunking step or a couple of SMO steps), or it is called by the wrapper algorithm (after each SVM optimization to full precision).

Recovering Regression and One-Class Classification. It should be noted that one-class classification is trivially implemented using $\sum_\alpha = 0$ while support vector regression (SVR) is typically performed by internally translating the SVR problem into a standard SVM classification problem with twice the number of examples once positively and once negatively labeled with corresponding α and α^*. Thus one needs direct access to α^* and computes $\sum_\alpha = -\sum_{i=1}^n (\alpha_i + \alpha_i^*)\varepsilon - \sum_{i=1}^n (\alpha_i - \alpha_i^*) y_i$ (cf. Sonnenburg et al., 2006a). Since this requires modification of the core SVM solver we implemented SVR only for interleaved optimization and SVMlight.

Efficiency Considerations and Kernel Caching. Note that the choice of the size of the kernel cache becomes crucial when applying MKL to large scale learning applications.[11] While for the wrapper algorithms only a *single* kernel SVM needs to be solved and thus a single large kernel cache should be used, the story is different for interleaved optimization. Since one must keep track of the several partial MKL objectives obj_m, requiring access to individual kernel rows, the same cache size should be used for all sub-kernels.

4.4.2 KERNEL NORMALIZATION

The normalization of kernels is as important for MKL as the normalization of features is for training regularized linear or single-kernel models. This is owed to the bias introduced by the regularization: optimal feature / kernel weights are requested to be small. This is easier to achieve for features (or entire feature spaces, as implied by kernels) that are scaled to be of large magnitude, while downscaling them would require a correspondingly upscaled weight for representing the same predictive model. Upscaling (downscaling) features is thus equivalent to modifying regularizers such that they penalize those features less (more). As is common practice, we here use isotropic regularizers, which penalize all dimensions uniformly. This implies that the kernels have to be normalized in a sensible way in order to represent an "uninformative prior" as to which kernels are useful.

There exist several approaches to kernel normalization, of which we use two in the computational experiments below. They are fundamentally different. The first one generalizes the common practice of standardizing features to entire kernels, thereby directly implementing the spirit of the discussion above. In contrast, the second normalization approach rescales the data points to unit norm in feature space. Nevertheless it can have a beneficial effect on the scaling of kernels, as we argue below.

Multiplicative Normalization. As done in Ong and Zien (2008), we multiplicatively normalize the kernels to have uniform variance of data points in feature space. Formally, we find a positive rescaling ρ_m of the kernel, such that the rescaled kernel $\tilde{k}_m(\cdot,\cdot) = \rho_m k_m(\cdot,\cdot)$ and the corresponding feature map $\tilde{\Phi}_m(\cdot) = \sqrt{\rho_m}\Phi_m(\cdot)$ satisfy

$$\frac{1}{n}\sum_{i=1}^{n} \left\| \tilde{\Phi}_m(x_i) - \tilde{\Phi}_m(x) \right\|^2 = 1$$

11. *Large scale* in the sense, that the data cannot be stored in memory or the computation reaches a maintainable limit. In the case of MKL this can be due both a large sample size or a high number of kernels.

for each $m = 1, \ldots, M$, where $\tilde{\Phi}_m(x) := \frac{1}{n}\sum_{i=1}^n \tilde{\Phi}_m(x_i)$ is the empirical mean of the data in feature space. The above equation can be equivalently be expressed in terms of kernel functions as

$$\frac{1}{n}\sum_{i=1}^n \tilde{k}_m(x_i,x_i) - \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n \tilde{k}_m(x_i,x_j) = 1,$$

so that the final normalization rule is

$$k(x,x) \longmapsto \frac{k(x,x)}{\frac{1}{n}\sum_{i=1}^n k(x_i,x_i) - \frac{1}{n^2}\sum_{i,j=1}^n, k(x_i,x_j)}.$$

Note that in case the kernel is centered (i.e., the empirical mean of the data points lies on the origin), the above rule simplifies to $k(x,x) \longmapsto k(x,x)/\frac{1}{n}\mathrm{tr}(K)$, where $\mathrm{tr}(K) := \sum_{i=1}^n k(x_i,x_i)$ is the trace of the kernel matrix K.

Spherical Normalization. Frequently, kernels are normalized according to

$$k(x,x) \longmapsto \frac{k(x,x)}{\sqrt{k(x,x)k(x,x)}}. \tag{23}$$

After this operation, $\|x\| = k(x,x) = 1$ holds for each data point x; this means that each data point is rescaled to lie on the unit sphere. Still, this also may have an effect on the scale of the features: a spherically normalized and centered kernel is also always multiplicatively normalized, because the multiplicative normalization rule becomes $k(x,x) \longmapsto k(x,x)/\frac{1}{n}\mathrm{tr}(K) = k(x,x)/1$.

Thus the spherical normalization may be seen as an approximate to the above multiplicative normalization and may be used as a substitute for it. Note, however, that it changes the data points themselves by eliminating length information; whether this is desired or not depends on the learning task at hand. Finally note that both normalizations achieve that the optimal value of C is not far from 1.

4.5 Limitations and Extensions of our Framework

In this section, we show the connection of ℓ_p-norm MKL to a formulation based on block norms, point out limitations and sketch extensions of our framework. To this aim let us recall the primal MKL problem (14) and consider the special case of ℓ_p-norm MKL given by

$$\inf_{w,b,\theta:\theta\geq\mathbf{0}} \quad C\sum_{i=1}^n V\left(\sum_{m=1}^M \langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b,\ y_i\right) + \frac{1}{2}\sum_{m=1}^M \frac{\|w_m\|^2_{\mathcal{H}_m}}{\theta_m}, \quad \text{s.t.} \quad \|\theta\|_p^2 \leq 1. \tag{24}$$

The subsequent proposition shows that (24) equivalently can be translated into the following mixed-norm formulation,

$$\inf_{w,b} \quad \tilde{C}\sum_{i=1}^n V\left(\sum_{m=1}^M \langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m} + b,\ y_i\right) + \frac{1}{2}\sum_{m=1}^M \|w_m\|^q_{\mathcal{H}_m}, \tag{25}$$

where $q = \frac{2p}{p+1}$, and $\tilde{C}$ is a constant. This has been studied by Bach et al. (2004) for $q = 1$ and by Szafranski et al. (2008) for hierarchical penalization.

Proposition 5 *Let be $p > 1$, be V a convex loss function, and define $q := \frac{2p}{p+1}$ (i.e., $p = \frac{q}{2-q}$). Optimization Problem* (24) *and* (25) *are equivalent, that is, for each C there exists a $\tilde{C} > 0$, such that for each optimal solution (w^*, b^*, θ^*) of OP* (24) *using C, we have that (w^*, b^*) is also optimal in OP* (25) *using $\tilde{C}$, and vice versa.*

Proof From Prop. 2 it follows that for any fixed w in (24) it holds for the w-optimal θ:

$$\exists \zeta: \quad \theta_m = \zeta \|w_m\|_{\mathcal{H}_m}^{\frac{2}{p+1}}, \quad \forall m = 1, \ldots, M.$$

Plugging the above equation into (24) yields

$$\inf_{w,b} \quad C \sum_{i=1}^{n} V\left(\sum_{m=1}^{M} \langle w_m, \psi_m(x_i) \rangle_{\mathcal{H}_m} + b,\ y_i \right) + \frac{1}{2\zeta} \sum_{m=1}^{M} \|w_m\|_{\mathcal{H}_m}^{\frac{2p}{p+1}}.$$

Defining $q := \frac{2p}{p+1}$ and $\tilde{C} := \zeta C$ results in (25). ■

Now, let us take a closer look on the parameter range of q. It is easy to see that when we vary p in the real interval $[1, \infty]$, then q is limited to range in $[1, 2]$. So in other words the methodology presented in this paper only covers the $1 \leq q \leq 2$ block norm case. However, from an algorithmic perspective our framework can be easily extended to the $q > 2$ case: although originally aiming at the more sophisticated case of hierarchical kernel learning, Aflalo et al. (2011) showed in particular that for $q \geq 2$, Equation (25) is equivalent to

$$\sup_{\theta: \theta \geq \mathbf{0}, \|\theta\|_r^2 \leq 1} \ \inf_{w,b} \quad \tilde{C} \sum_{i=1}^{n} V\left(\sum_{m=1}^{M} \langle w_m, \psi_m(x_i) \rangle_{\mathcal{H}_m} + b,\ y_i \right) + \frac{1}{2} \sum_{m=1}^{M} \theta_m \|w_m\|_{\mathcal{H}_m}^2,$$

where $r := \frac{q}{q-2}$. Note the difference to ℓ_p-norm MKL: the mixing coefficients θ appear in the nominator and by varying r in the interval $[1, \infty]$, the range of q in the interval $[2, \infty]$ can be obtained, which explains why this method is complementary to ours, where q ranges in $[1, 2]$.

It is straightforward to show that for every fixed (possibly suboptimal) pair (w, b) the optimal θ is given by

$$\theta_m = \frac{\|w_m\|_{\mathcal{H}_m}^{\frac{2}{r-1}}}{\left(\sum_{m'=1}^{M} \|w_{m'}\|_{\mathcal{H}_{m'}}^{\frac{2r}{r-1}} \right)^{1/r}}, \quad \forall m = 1, \ldots, M.$$

The proof is analogous to that of Prop. 2 and the above analytical update formula can be used to derive a block coordinate descent algorithm that is analogous to ours. In our framework, the mixings θ, however, appear in the denominator of the objective function of Optimization Problem (14). Therefore, the corresponding update formula in our framework is

$$\theta_m = \frac{\|w_m\|_{\mathcal{H}_m}^{\frac{-2}{r-1}}}{\left(\sum_{m'=1}^{M} \|w_{m'}\|_{\mathcal{H}_{m'}}^{\frac{-2r}{r-1}} \right)^{1/r}}, \quad \forall m = 1, \ldots, M. \tag{26}$$

This shows that we can simply optimize $2 < q \leq \infty$-block-norm MKL within our computational framework, using the update formula (26).

5. Theoretical Analysis

In this section we present a theoretical analysis of ℓ_p-norm MKL, based on Rademacher complexities.[12] We prove a theorem that converts any Rademacher-based generalization bound on ℓ_1-norm MKL into a generalization bound for ℓ_p-norm MKL (and even more generally: arbitrary-norm MKL). Remarkably this ℓ_1-to-ℓ_p conversion is obtained almost without any effort: by a simple 5-line proof. The proof idea is based on Kloft et al. (2010). We remark that an ℓ_p-norm MKL bound was already given in Cortes et al. (2010a), but their bound is only valid for the special cases where $p/(p-1)$ is an integer and is not tight for small p, as it diverges to infinity when $p > 1$ and p approaches one. By contrast, beside a negligible $\log(M)$-factor, our result matches the best known lower bounds, when p approaches one.

Let us start by defining the hypothesis set that we want to investigate. Following Cortes et al. (2010a), we consider the following hypothesis class for $p \in [1,\infty]$:

$$H_M^p := \left\{ h : \mathcal{X} \to \mathbb{R} \;\middle|\; h(x) = \sum_{m=1}^{M} \sqrt{\theta_m} \langle w_m, \psi_m(x) \rangle_{\mathcal{H}_m},\ \|w\|_{\mathcal{H}} \le 1,\ \|\theta\|_p \le 1 \right\}.$$

Solving our primal MKL problem (14) corresponds to empirical risk minimization in the above hypothesis class. We are thus interested in bounding the generalization error of the above class w.r.t. an i.i.d. sample $(x_1,y_1),\ldots,(x_n,y_n) \in \mathcal{X} \times \{-1,1\}$ from an arbitrary distribution P. In order to do so, we compute the *Rademacher complexity*,

$$\mathscr{R}(H_M^p) := \mathbb{E}\left[\sup_{h \in H_M^p} \frac{1}{n} \sum_{i=1}^{n} \sigma_i h(x_i) \right],$$

where $\sigma_1,\ldots,\sigma_n$ are independent Rademacher variables (i.e., they obtain the values -1 or +1 with the same probability 0.5) and the $\mathbb{E}$ is the expectation operator that removes the dependency on all random variables, that is, σ_i, x_i, and y_i ($i = 1,\ldots,n$). If the Rademacher complexity is known, there is a large body of results that can be used to bound the generalization error (e.g., Koltchinskii and Panchenko, 2002; Bartlett and Mendelson, 2002).

We now show a simple ℓ_q-to-ℓ_p conversion technique for the Rademacher complexity, which is the main result of this section:

Theorem 6 (ℓ_q-to-ℓ_p Conversion) *For any sample of size n and $1 \le q \le p \le \infty$ the Rademacher complexity of the hypothesis set H_M^p can be bounded in terms of H_M^q,*

$$\mathscr{R}(H_M^p) \le \sqrt{M^{\frac{1}{q}-\frac{1}{p}}}\, \mathscr{R}(H_M^q).$$

In particular, we have $\mathscr{R}(H_M^p) \le \sqrt{M^{1/p^*}}\, \mathscr{R}(H_M^1)$ (ℓ_1-to-ℓ_p Conversion), where $p^* := p/(p-1)$ is the conjugated exponent of p.

Proof By Hölder's inequality (e.g., Steele, 2004), denoting $\theta^p := (\theta_1^p,\cdots,\theta_M^p)^\top$, we have for all non-negative $\theta \in \mathbb{R}^M$,

$$\|\theta\|_q = (\mathbf{1}^\top \theta^q)^{1/q} \le \left(\|\mathbf{1}\|_{(p/q)^*} \|\theta^q\|_{p/q} \right)^{1/q} = M^{\frac{1}{q(p/q)^*}} \|\theta\|_p = M^{\frac{1}{q}-\frac{1}{p}} \|\theta\|_p \,. \tag{27}$$

12. An introduction to statistical learning theory, which may equip the reader with the needed notions used in this section, is given in Bousquet et al. (2004). See also, for example, Section 4 in Shawe-Taylor and Cristianini (2004).

Hence,

$$\begin{aligned}
\mathscr{R}(H_M^p) &\overset{\text{Def.}}{=} \mathbb{E}\left[\sup_{w:\|w\|_{\mathcal{H}}\le 1,\ \theta:\|\theta\|_p\le 1} \frac{1}{n}\sum_{i=1}^{n}\sigma_i \sum_{m=1}^{M}\sqrt{\theta_m}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m}\right] \\
&\overset{(27)}{\le} \mathbb{E}\left[\sup_{w:\|w\|_{\mathcal{H}}\le 1,\ \theta:\|\theta\|_q\le M^{\frac{1}{q}-\frac{1}{p}}} \frac{1}{n}\sum_{i=1}^{n}\sigma_i \sum_{m=1}^{M}\sqrt{\theta_m}\langle w_m, \psi_m(x_i)\rangle_{\mathcal{H}_m}\right] \\
&= \mathbb{E}\left[\sup_{w:\|w\|_{\mathcal{H}}\le 1,\ \theta:\|\theta\|_q\le 1} \frac{1}{n}\sum_{i=1}^{n}\sigma_i \sum_{m=1}^{M}\sqrt{\theta_m M^{\frac{1}{q}-\frac{1}{p}}}\langle w_m, \psi_m(x)\rangle_{\mathcal{H}_m}\right] \\
&\overset{\text{Def.}}{=} \sqrt{M^{\frac{1}{q}-\frac{1}{p}}}\,\mathscr{R}(H_M^q).
\end{aligned}$$

■

Remark 7 *More generally we have that for any norm $\|\cdot\|_\star$ on $\mathbb{R}^M$, because all norms on $\mathbb{R}^M$ are equivalent (e.g., Rudin, 1991), there exists a $c_\star \in \mathbb{R}$ such that*

$$\mathscr{R}(H_M^p) \le c_\star \mathscr{R}(H_M^\star).$$

This means the conversion technique extends to arbitrary norms: for any given norm $\|\cdot\|_\star$, we can convert any bound on $\mathscr{R}(H_M^p)$ into a bound on the Rademacher complexity $\mathscr{R}(H_M^\star)$ of hypothesis set induced by $\|\cdot\|_\star$.

A nice characteristic of the above result is that we can make use of any existing bound on the Rademacher complexity of H_M^1 in order to obtain a generalization bound for H_M^p. This fact is illustrated in the following. For example, it has recently been shown:

Theorem 8 (Cortes et al., 2010a) *Let $M > 1$ and assume that $k_m(x,x) \le R^2$ for all $x \in \mathcal{X}$ and $m = 1,\ldots,M$. Then, for any sample of size n, the Rademacher complexities of the hypothesis sets H_M^1 and H_M^p can be bounded as follows (where $c := 23/22$ and $\lceil\cdot\rceil$ rounds to the next largest integer):*

$$\mathscr{R}(H_M^1) \le \sqrt{\frac{ce\lceil\log M\rceil R^2}{n}}, \quad \mathscr{R}(H_M^p) \le \sqrt{\frac{cp^* M^{1/p^*} R^2}{n}},$$

for any $p > 1$ such that p^* is an even integer.

For $p = 1$ [$p > 1$] the above result directly leads to a $O(\sqrt{\log M})$ $\left[O(\sqrt{M^{1/p^*}})\right]$ bound on the generalization error and thus substantially improves on a series of loose results given within the past years (see Cortes et al., 2010a, and references therein). Unfortunately, since p^* is required to be an integer, the range of p is restricted to $p \in [1,2]$. As a remedy, in this paper we use the ℓ_q-to-ℓ_p Conversion technique to the above result[13] to obtain a bound for H_M^p that holds for all $p \in [1,\ldots,\infty]$: the following corollary is obtained from the previous theorem by using ℓ_q-to-ℓ_p-norm conversion for $q = 1$ and $q = \lceil p^*\rceil^*$, respectively, and then taking the minimum value of the so-obtained bounds.

13. The point here is that we could use any ℓ_1-bound, for example, the bounds of Kakade et al. (2009) and Kloft et al. (2010) have the same favorable $O(\log M)$ rate; in particular, whenever a new ℓ_1-bound is proven, we can plug it into our conversion technique to obtain a new bound.

Corollary 9 (of the previous two theorems) *Let $M > 1$ and assume that $k_m(x,x) \leq R^2$ for all $x \in \mathcal{X}$ and $m = 1,\ldots,M$. Then, for any sample of size n, the Rademacher complexity of the hypothesis set H_M^p can be bounded as follows:*

$$\forall p \in [1,...,\infty]: \quad \mathscr{R}(H_M^p) \leq \sqrt{\frac{cM^{1/p^*}R^2 \min\left(e\lceil \log M\rceil, \lceil p^*\rceil\right)}{n}},$$

where $p^* := p/(p-1)$ is the conjugated exponent of p and $c := 23/22$.

It is instructive to compare the above bound, which we obtained by our ℓ_q-to-ℓ_p conversion technique, with the one given in Cortes et al. (2010a): that is $\mathscr{R}(H_M^p) \leq \sqrt{\frac{cep^*M^{1/p^*}R^2}{n}}$ for any $p \in [1,...,\infty]$ such that p^* is an integer. First, we observe that for $p = 2$ the bounds' rates coincide. Second, we observe that for small p (close to one), the p^*-factor in the Cortes-bound leads to considerably high constants. When p approaches one, it even diverges to infinity. In contrast, our bound converges to $\mathscr{R}(H_M^p) \leq \sqrt{\frac{ce\lceil \log M\rceil R^2}{n}}$ when p approaches one, which is precisely the tight 1-norm bound of Thm. 8. Finally, it is also interesting to consider the case $p \geq 2$ (which is not covered by the Cortes et al., 2010a bound): if we let $p \to \infty$, we obtain $\mathscr{R}(H_M^p) \leq \sqrt{\frac{2cMR^2}{n}}$. This matches the well-known $O\left(\sqrt{M}\right)$ lower bounds based on the VC-dimension (e.g., Devroye et al., 1996, Section 14).

We now make use of the above analysis of the Rademacher complexity to bound the generalization error. There are many results in the literature that can be employed to this aim. Ours is based on Thm. 7 in Bartlett and Mendelson (2002):

Corollary 10 *Let $M > 1$ and $p \in]1,...,\infty]$. Assume that $k_m(x,x) \leq R^2$ for all $x \in \mathcal{X}$ and $m = 1,\ldots,M$. Assume the loss $V : \mathbb{R} \to [0,1]$ is Lipschitz with constant L and $V(t) \geq 1$ for all $t \leq 0$. Set $p^* := p/(p-1)$ and $c := 23/22$. Then, the following holds with probability larger than $1-\delta$ over samples of size n for all classifiers $h \in H_M^p$:*

$$R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{cM^{1/p^*}R^2 \min\left(e\lceil \log M\rceil, \lceil p^*\rceil\right)}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}},$$

where $R(h) = \mathrm{P}\left[yh(x) \leq 0\right]$ is the expected risk w.r.t. 0-1 loss and $\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^n V(y_i h(x_i))$ is the empirical risk w.r.t. loss V.

The above theorem is formulated for general Lipschitz loss functions. Since the margin loss $V(t) = \min\left(1, [1-t/\gamma]_+\right)$ is Lipschitz with constant $1/\gamma$ and upper bounding the 0-1 loss, it fulfills the preliminaries of the above corollary. Hence, we immediately obtain the following radius-margin bound (see also Koltchinskii and Panchenko, 2002):

Corollary 11 (ℓ_p-norm MKL Radius-Margin Bound) *Fix the margin $\gamma > 0$. Let $M > 1$ and $p \in]1,...,\infty]$. Assume that $k_m(x,x) \leq R^2$ for all $x \in \mathcal{X}$ and $m = 1,\ldots,M$. Set $p^* := p/(p-1)$ and $c := 23/22$. Then, the following holds with probability larger than $1-\delta$ over samples of size n for all classifiers $h \in H_M^p$:*

$$R(h) \leq \widehat{R}(h) + \frac{4R}{\gamma}\sqrt{\frac{cM^{1/p^*} \min\left(e\lceil \log M\rceil, \lceil p^*\rceil\right)}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}},$$

where $R(h) = \mathrm{P}\big[yh(x) \leq 0\big]$ *is the expected risk w.r.t. 0-1 loss and* $\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^{n} \min\big(1, [1 - y_i h(x_i)/\gamma]_+\big)$ *the empirical risk w.r.t. margin loss.*

Finally, we would like to point out that, for reasons stated in Remark 7, the ℓ_q-to-ℓ_p conversion technique can be extended to norms different than ℓ_p . This lets us extend the above bounds to, for example, block norms and sums of block norms as used in elastic-net regularization (see Kloft et al., 2010, for such bounds), but also non-isotropic norms such as weighted ℓ_p-norms.

5.1 Case-based Analysis of a Sparse and a Non-Sparse Scenario

From the results given in the last section it seems that it is beneficial to use a sparsity-inducing ℓ_1-norm penalty when learning with multiple kernels. This however somewhat contradicts our empirical evaluation, which indicated that the optimal norm parameter p depends on the true underlying sparsity of the problem. Indeed, as we show below, a refined theoretical analysis supports this intuitive claim. We show for an exemplary scenario that if the underlying truth is uniformly non-sparse, then a non-sparse ℓ_p-norm is more promising than a sparse one. On the other hand, we illustrate that in a sparse scenario, the sparsity-inducing ℓ_1-norm indeed can be beneficial.

We start by reparametrizing our hypothesis set based on block norms: by Prop. 5 it holds that

$$H_M^p = \left\{ h : \mathcal{X} \to \mathbb{R} \;\middle|\; h(x) = \sum_{m=1}^{M} \langle w_m, \psi_m(x) \rangle_{\mathcal{H}_m},\ \|w\|_{2,q} \leq 1,\ q := 2p/(p+1) \right\},$$

where $||w||_{2,q} := \left(\sum_{m=1}^{M} ||w_m||_{\mathcal{H}_m}^q\right)^{1/q}$ is the $\ell_{2,q}$-block norm. This means we can equivalently parametrize our hypothesis set in terms of block norms. Second, let us generalize the set by introducing an additional parameter C as follows

$${}^C H_M^p := \left\{ h : \mathcal{X} \to \mathbb{R} \;\middle|\; h(x) = \sum_{m=1}^{M} \langle w_m, \psi_m(x) \rangle_{\mathcal{H}_m},\ \|w\|_{2,q} \leq C,\ q := 2p/(p+1) \right\}.$$

Clearly, ${}^C H_M^p = H_M^p$ for $C = 1$, which explains why the parametrization via C is more general. It is straightforward to verify that $\mathscr{R}\left({}^C H_M^p\right) = C\mathscr{R}\left(H_M^p\right)$ for any C. Hence, under the preliminaries of Corollary 10, we have

$$\begin{aligned} p > 1: \qquad & R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{cM^{1/p^*}R^2C^2\min\left(e\lceil \log M \rceil, \lceil p^* \rceil\right)}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}, \\ p = 1: \qquad & R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{ce\lceil \log M \rceil R^2C^2}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}\,. \end{aligned} \tag{28}$$

We will exploit the above bound in the following two illustrate examples.

Example 1. Let the input space be $\mathcal{X} = \mathbb{R}^M$, and the feature map be $\psi_m(x) = x_m$ for all $m = 1, \ldots, M$ and $x = (x_1, ..., x_M) \in \mathcal{X}$ (in other words, ψ_m is a projection on the mth feature). Assume that the Bayes-optimal classifier is given by

$$w_{\text{Bayes}} = (1, \ldots, 1)^\top \in \mathbb{R}^M.$$

This means the best classifier possible is uniformly non-sparse (see Fig. 1, left). Clearly, it can be advantageous to work with a hypothesis set that is rich enough to contain the Bayes classifier,

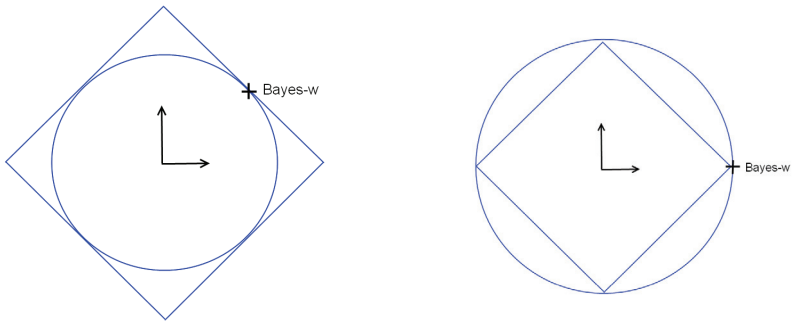


Figure 1: Illustration of the two analyzed cases for $M = 2$: a uniformly non-sparse (Example 1, left) and a sparse (Example 2, right) Scenario.

that is, $(1,\ldots,1)^\top \in {}^C H_M^p$. In our example, this is the case if and only if $\|(1,\ldots,1)^\top\|_{2p/(p+1)} \leq C$, which itself is equivalent to $M^{(p+1)/2p} \leq C$. The bound (28) attains its minimal value under the latter constraint for $M^{(p+1)/2p} = C$. Resubstitution into the bound yields

$$p > 1: \qquad R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{cM^2R^2\min\left(e\lceil\log M\rceil, \lceil p^*\rceil\right)}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

$$p = 1: \qquad R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{ceM^2\lceil\log M\rceil R^2C^2}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}\,.$$

Let us now compare the so obtained rate: for $p > 1$ we get $O(M^2)$ and for $p = 1$ we have $O(M^2\log(M))$. So the rates differ by a $\log(M)$ factor. This means that in this particular (non-sparse) example, neglecting the constants, the non-sparse $p > 1$-norm MKL variants yield a strictly better generalization bound than ℓ_1-norm MKL.

Example 2. In this second example we consider the same input space and kernels as before. But this time we assume a *sparse* Bayes-optimal classifier (see Fig. 1, right)

$$w_{\text{Bayes}} = (1,0,\ldots,0)^\top \in \mathbb{R}^M.$$

As in the previous example, in order w_{Bayes} to be in the hypothesis set, we have to require $\|(1,0,\ldots,0)^\top\|_{2p/(p+1)} \leq C$. But this time this simply solves to $C \geq 1$, which is independent of the norm parameter p. Thus, inserting $C = 1$ in the bound (28), we obtain

$$p > 1: \qquad R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{cM^2R^2\min\left(e\lceil\log M\rceil, \lceil p^*\rceil\right)}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

$$p = 1: \qquad R(h) \leq \widehat{R}(h) + 4L\sqrt{\frac{ceM^2\lceil\log M\rceil R^2}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}\,.$$

Clearly, in this particular sparse example, the $\ell_{p=1}$-bound is considerably smaller than the one of $\ell_{p>1}$-norm MKL—especially, if the number of kernels is high compared to the sample size. This is also intuitive: if the underlying truth is sparse, we expect a sparsity-inducing norm to match well the ground truth.

We conclude from the previous two examples that the optimal norm parameter p depends on the underlying ground truth: if it is sparse, then choosing a sparse regularization is beneficial; otherwise, a non-sparse norm p can perform well. This is somewhat contrary to anecdotal reports, which claim that sparsity-inducing norms are beneficial in high (kernel) dimensions. This is because those analyses implicitly assume the ground truth to be sparse. The present paper, however, clearly shows that we might encounter a non-sparse ground truth in practical applications (see experimental section).

6. Computational Experiments

In this section we study non-sparse MKL in terms of computational efficiency and predictive accuracy. We apply the method of Sonnenburg et al. (2006a) in the case of $p = 1$. We write ℓ_∞-norm MKL for a regular SVM with the unweighted-sum kernel $K = \sum_m K_m$.

We first study a toy problem in Section 6.1 where we have full control over the distribution of the relevant information in order to shed light on the appropriateness of sparse, non-sparse, and ℓ_∞-MKL. We report on real-world problems from bioinformatics, namely protein subcellular localization (Section 6.2), finding transcription start sites of RNA Polymerase II binding genes in genomic DNA sequences (Section 6.3), and reconstructing metabolic gene networks (Section 6.4). All data sets used in this section were made available online (see supplementary homepage of this paper: `http://doc.ml.tu-berlin.de/nonsparse_mkl/`).

6.1 Measuring the Impact of Data Sparsity—Toy Experiment

The goal of this section is to study the relationship of the level of sparsity of the true underlying function to be learned to the chosen norm p in the model. Intuitively, we might expect that the optimal choice of p directly corresponds to the true level of sparsity. Apart from verifying this conjecture, we are also interested in the effects of suboptimal choice of p. To this aim we constructed several artificial data sets in which we vary the degree of sparsity in the true kernel mixture coefficients. We go from having all weight focused on a single kernel (the highest level of sparsity) to uniform weights (the least sparse scenario possible) in several steps. We then study the statistical performance of ℓ_p-norm MKL for different values of p that cover the entire range $[1,\infty]$.

We generated a data set as follows (we made this so-called *mkl-toy* data set available at the mldata repository[14]). An n-element balanced sample $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ is generated from two $d = 50$-dimensional isotropic Gaussian distributions with equal covariance matrices $C = I_{d\times d}$ and equal, but opposite, means $\mu_1 = \frac{\rho}{\|\theta\|_2}\theta$ and $\mu_2 = -\mu_1$. Thereby θ is a binary vector, that is, $\forall i : \theta_i \in \{0,1\}$, encoding the true underlying data sparsity as follows. Zero components $\theta_i = 0$ clearly imply identical means of the two classes' distributions in the ith feature set; hence the latter does not carry any discriminating information. In summary, the fraction of zero components, $\nu(\theta) = 1 - \frac{1}{d}\sum_{i=1}^d \theta_i$, is a measure for the feature sparsity of the learning problem.

For $\nu \in \{0, 0.44, 0.64, 0.82, 0.92, 1\}$ we generate six data sets $\mathcal{D}_1, \ldots, \mathcal{D}_6$ fixing $\rho = 1.75$. Then, each feature is input to a linear kernel and the resulting kernel matrices are multiplicatively normalized as described in Section 4.4.2. Hence, $\nu(\theta)$ gives the fraction of noise kernels in the working kernel set. Then, classification models are computed by training ℓ_p-norm MKL for $p = 1, 4/3, 2, 4, \infty$ on each $\mathcal{D}_i$. Soft margin parameters C are tuned on independent $10,000$-elemental validation sets

14. The repository can be found at `http://mldata.org/repository/data/viewslug/mkl-toy/`.

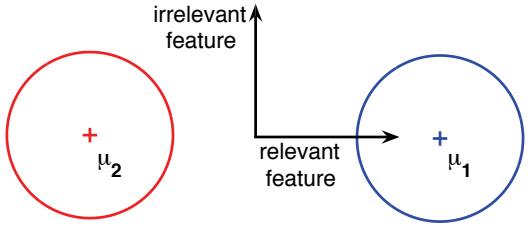


Figure 2: Illustration of the toy experiment for $\theta = (1,0)^\top$.

by grid search over $C \in 10^{[-4,3.5,\ldots,0]}$ (optimal Cs are attained in the interior of the grid). The relative duality gaps were optimized up to a precision of 10^{-3}. We report on test errors evaluated on 10,000-elemental independent test sets and pure mean ℓ_2 model errors of the computed kernel mixtures, that is $\mathrm{ME}(\widehat{\theta}) = \|\zeta(\widehat{\theta}) - \zeta(\theta)\|_2$, where $\zeta(x) = \frac{x}{\|x\|_2}$.

The results are shown in Fig. 3 for $n = 50$ and $n = 800$, where the figures on the left show the test errors and the ones on the right the model errors $\mathrm{ME}(\widehat{\theta})$. Regarding the latter, model errors reflect the corresponding test errors for $n = 50$. This observation can be explained by statistical learning theory. The minimizer of the empirical risk performs unstable for small sample sizes and the model selection results in a strongly regularized hypothesis, leading to the observed agreement between test error and model error.

Unsurprisingly, ℓ_1 performs best and reaches the Bayes error in the sparse scenario, where only a single kernel carries the whole discriminative information of the learning problem. However, in the other scenarios it mostly performs worse than the other MKL variants. This is remarkable because the underlying ground truth, that is, the vector θ, is sparse in all but the uniform scenario. In other words, selecting this data set may imply a bias towards ℓ_1-norm. In contrast, the vanilla SVM using an unweighted sum kernel performs best when all kernels are equally informative, however, its performance does not approach the Bayes error rate. This is because it corresponds to a $\ell_{2,2}$-block norm regularization (see Sect. 4.5) but for a truly uniform regularization a ℓ_∞-block norm penalty (as employed in Nath et al., 2009) would be needed. This indicates a limitation of our framework; it shall, however, be kept in mind that such a uniform scenario might quite artificial. The non-sparse ℓ_4- and ℓ_2-norm MKL variants perform best in the balanced scenarios, that is, when the noise level is ranging in the interval 64%-92%. Intuitively, the non-sparse ℓ_4-norm MKL is the most robust MKL variant, achieving a test error of less than 10% in all scenarios. Tuning the sparsity parameter p for each experiment, ℓ_p-norm MKL achieves the lowest test error across all scenarios.

When the sample size is increased to $n = 800$ training instances, test errors decrease significantly. Nevertheless, we still observe differences of up to 1% test error between the best (ℓ_∞-norm MKL) and worst (ℓ_1-norm MKL) prediction model in the two most non-sparse scenarios. Note that all ℓ_p-norm MKL variants perform well in the sparse scenarios. In contrast with the test errors, the mean model errors depicted in Figure 3 (bottom, right) are relatively high. Similarly to above reasoning, this discrepancy can be explained by the minimizer of the empirical risk becoming stable when increasing the sample size, which decreases the generalization error (see theoretical Analysis in Section 5, where it was shown that the speed of the minimizer becoming stable is at least of a rate of $O(1/\sqrt{n})$). Again, ℓ_p-norm MKL achieves the smallest test error for all scenarios for appropriately chosen p and for a fixed p across all experiments, the non-sparse ℓ_4-norm MKL performs the most robustly.

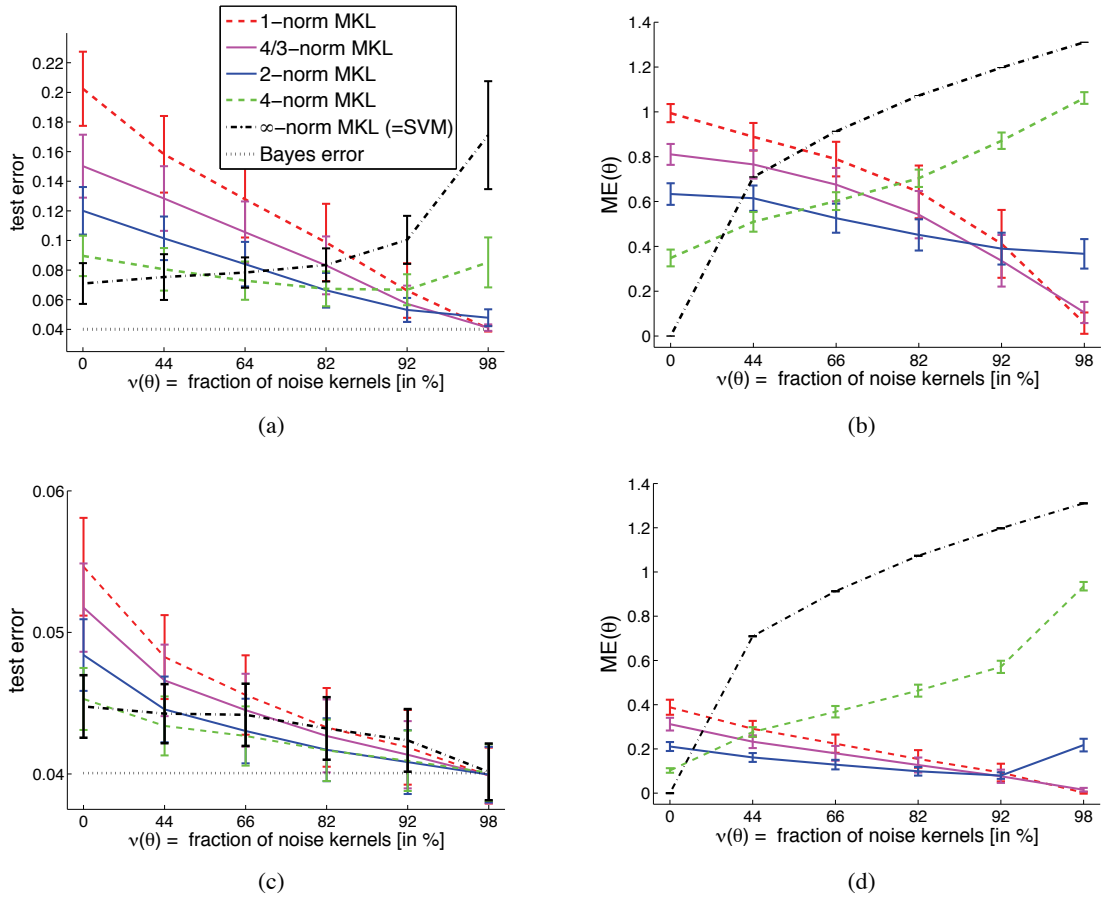


Figure 3: Results of the artificial experiment for sample sizes of $n = 50$ (top) and $n = 800$ (below) training instances in terms of test errors (left) and mean ℓ_2 model errors $\mathrm{ME}(\widehat{\theta})$ (right).

In summary, the choice of the norm parameter p is important for small sample sizes, whereas its impact decreases with an increase of the training data. As expected, sparse MKL performs best in sparse scenarios, while non-sparse MKL performs best in moderate or non-sparse scenarios, and for uniform scenarios the unweighted-sum kernel SVM performs best. For appropriately tuning the norm parameter, ℓ_p-norm MKL proves robust in all scenarios.

6.2 Protein Subcellular Localization—A Sparse Scenario

The prediction of the subcellular localization of proteins is one of the rare empirical success stories of ℓ_1-norm-regularized MKL (Ong and Zien, 2008; Zien and Ong, 2007): after defining 69 kernels that capture diverse aspects of protein sequences, ℓ_1-norm-MKL could raise the predictive accuracy significantly above that of the unweighted sum of kernels, and thereby also improve on established prediction systems for this problem. This has been demonstrated on 4 data sets, corresponding to 4 different sets of organisms (plants, non-plant eukaryotes, Gram-positive and Gram-negative

ℓ_p-norm	1	32/31	16/15	8/7	4/3	2	4	8	16	∞
plant	**8.18**	8.22	8.20	8.21	8.43	9.47	11.00	11.61	11.91	11.85
std. err.	±0.47	±0.45	±0.43	±0.42	±0.42	±0.43	±0.47	±0.49	±0.55	±0.60
nonpl	**8.97**	9.01	9.08	9.19	9.24	9.43	9.77	10.05	10.23	10.33
std. err.	±0.26	±0.25	±0.26	±0.27	±0.29	±0.32	±0.32	±0.32	±0.32	±0.31
psortNeg	9.99	9.91	**9.87**	10.01	10.13	11.01	12.20	12.73	13.04	13.33
std. err.	±0.35	±0.34	±0.34	±0.34	±0.33	±0.32	±0.32	±0.34	±0.33	±0.35
psortPos	13.07	**13.01**	13.41	13.17	13.25	14.68	15.55	16.43	17.36	17.63
std. err.	±0.66	±0.63	±0.67	±0.62	±0.61	±0.67	±0.72	±0.81	±0.83	±0.80

Table 1: Results for Protein Subcellular Localization. For each of the 4 data sets (rows) and each considered norm (columns), we present a measure of prediction error together with its standard error. As measure of prediction error we use 1 minus the average MCC, displayed as percentage.

bacteria) with differing sets of relevant localizations. In this section, we investigate the performance of non-sparse MKL on the same 4 data sets.

The experimental setup used here is related to that of Ong and Zien (2008), although it deviates from it in several details. The kernel matrices are multiplicatively normalized as described in Section 4.4.2. For each data set, we perform the following steps for each of the 30 predefined splits in training set and test set (downloaded from the same URL): We consider norms $p \in \{1, 32/31, 16/15, 8/7, 4/3, 2, 4, 8, \infty\}$ and regularization constants $C \in \{1/32, 1/8, 1/2, 1, 2, 4, 8, 32, 128\}$. For each parameter setting (p, C), we train ℓ_p-norm MKL using a 1-vs-rest strategy on the training set. The predictions on the test set are then evaluated w.r.t. average (over the classes) MCC (Matthews correlation coefficient). As we are only interested in the influence of the norm on the performance, we forbear proper cross-validation (the so-obtained systematical error affects all norms equally). Instead, for each of the 30 data splits and for each p, the value of C that yields the highest MCC is selected. Thus we obtain an optimized C and *MCC* value for each combination of data set, split, and norm p. For each norm, the final *MCC* value is obtained by averaging over the data sets and splits (i.e., C is selected to be optimal for each data set and split).

The results, shown in Table 1, indicate that indeed, with proper choice of a non-sparse regularizer, the accuracy of ℓ_1-norm can be recovered. On the other hand, non-sparse MKL can approximate the ℓ_1-norm arbitrarily close, and thereby approach the same results. However, even when 1-norm is clearly superior to ∞-norm, as for these 4 data sets, it is possible that intermediate norms perform even better. As the table shows, this is indeed the case for the PSORT data sets, albeit only slightly and not significantly so.

We briefly mention that the superior performance of $\ell_{p\approx 1}$-norm MKL in this setup is not surprising. There are four sets of 16 kernels each, in which each kernel picks up very similar information: they only differ in number and placing of gaps in all substrings of length 5 of a given part of the protein sequence. The situation is roughly analogous to considering (inhomogeneous) polynomial kernels of different degrees on the same data vectors. This means that they carry large parts of overlapping information. By construction, also some kernels (those with less gaps) in principle have access to more information (similar to higher degree polynomials including low degree polynomials). Further, Ong and Zien (2008) studied single kernel SVMs for each kernel individually and found that in most cases the 16 kernels from the same subset perform very similarly. This means

that each set of 16 kernels is highly redundant and the excluded parts of information are not very discriminative. This renders a non-sparse kernel mixture ineffective. We conclude that ℓ_1-norm must be the best prediction model.

6.3 Gene Start Recognition—A Weighted Non-Sparse Scenario

This experiment aims at detecting transcription start sites (TSS) of RNA Polymerase II binding genes in genomic DNA sequences. Accurate detection of the transcription start site is crucial to identify genes and their promoter regions and can be regarded as a first step in deciphering the key regulatory elements in the promoter region that determine transcription.

Transcription start site finders exploit the fact that the features of promoter regions and the transcription start sites are different from the features of other genomic DNA (Bajic et al., 2004). Many such detectors thereby rely on a combination of feature sets which makes the learning task appealing for MKL. For our experiments we use the data set from Sonnenburg et al. (2006b) which contains a curated set of 8,508 TSS annotated genes using dbTSS version 4 (Suzuki et al., 2002) and refseq genes. These are translated into positive training instances by extracting windows of size $[-1000,+1000]$ around the TSS. Similar to Bajic et al. (2004), 85,042 negative instances are generated from the interior of the gene using the same window size. Following Sonnenburg et al. (2006b), we employ five different kernels representing the TSS signal (weighted degree with shift), the promoter (spectrum), the 1st exon (spectrum), angles (linear), and energies (linear). Optimal kernel parameters are determined by model selection in Sonnenburg et al. (2006b). The kernel matrices are spherically normalized as described in section 4.4.2. We reserve 13,000 and 20,000 randomly drawn instances for validation and test sets, respectively, and use the remaining 60,000 as the training pool. Soft margin parameters C are tuned on the validation set by grid search over $C \in 2^{[-2,-1,\ldots,5]}$ (optimal Cs are attained in the interior of the grid). Figure 4 shows test errors for varying training set sizes drawn from the pool; training sets of the same size are disjoint. Error bars indicate standard errors of repetitions for small training set sizes.

Regardless of the sample size, ℓ_1-norm MKL is significantly outperformed by the sum-kernel. On the contrary, non-sparse MKL significantly achieves higher AUC values than the ℓ_∞-norm MKL for sample sizes up to 20k. The scenario is well suited for ℓ_2-norm MKL which performs best. Finally, for 60k training instances, all methods but ℓ_1-norm MKL yield the same performance. Again, the superior performance of non-sparse MKL is remarkable, and of significance for the application domain: the method using the unweighted sum of kernels (Sonnenburg et al., 2006b) has recently been confirmed to be leading in a comparison of 19 state-of-the-art promoter prediction programs (Abeel et al., 2009), and our experiments suggest that its accuracy can be further elevated by non-sparse MKL.

We give a brief explanation of the reason for optimality of a non-sparse ℓ_p-norm in the above experiments. It has been shown by Sonnenburg et al. (2006b) that there are three highly and two moderately informative kernels. We briefly recall those results by reporting on the AUC performances obtained from training a single-kernel SVM on each kernel individually: TSS signal 0.89, promoter 0.86, 1st exon 0.84, angles 0.55, and energies 0.74, for fixed sample size $n = 2000$. While non-sparse MKL distributes the weights over all kernels (see Fig. 4), sparse MKL focuses on the best kernel. However, the superior performance of non-sparse MKL means that dropping the remaining kernels is detrimental, indicating that they may carry additional discriminative information.

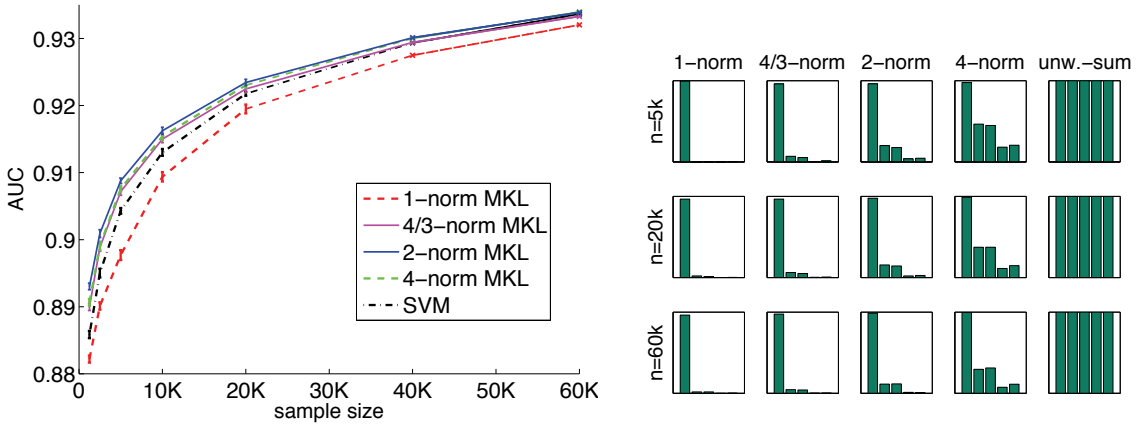


Figure 4: (left) Area under ROC curve (AUC) on test data for TSS recognition as a function of the training set size. Notice the tiny bars indicating standard errors w.r.t. repetitions on disjoint training sets. (right) Corresponding kernel mixtures. For $p = 1$ consistent sparse solutions are obtained while the optimal $p = 2$ distributes weights on the weighted degree and the 2 spectrum kernels in good agreement to Sonnenburg et al. (2006b).

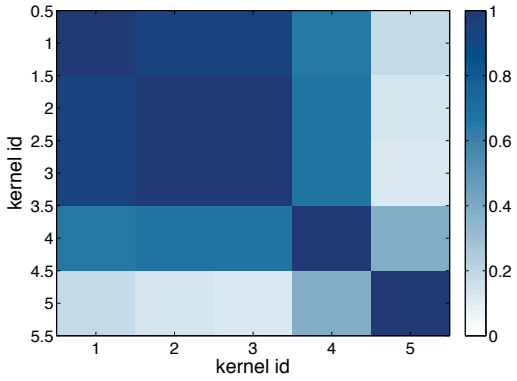


Figure 5: Pairwise alignments of the kernel matrices are shown for the gene start recognition experiment. From left to right, the ordering of the kernel matrices is TSS signal, promoter, 1st exon, angles, and energies. The first three kernels are highly correlated, as expected by their high AUC performances (AUC=0.84–0.89) and the angle kernel correlates decently (AUC=0.55). Surprisingly, the energy kernel correlates only few, despite a descent AUC of 0.74.

To investigate this hypothesis we computed the pairwise alignments of the kernel matrices, that is, $\mathcal{A}(i,j) = \frac{<K_i,K_j>_F}{\|K_i\|_F\|K_j\|_F}$, with respect to the Frobenius dot product (e.g., Golub and van Loan, 1996). The computed alignments are shown in Fig. 5. One can observe that the three relevant kernels are highly aligned as expected since they are correlated via the labels.

However, the energy kernel shows only a slight correlation with the remaining kernels, which is surprisingly little compared to its single kernel performance (AUC=0.74). We conclude that this kernel carries complementary and orthogonal information about the learning problem and should thus be included in the resulting kernel mixture. This is precisely what is done by non-sparse MKL, as can be seen in Fig. 4(right), and the reason for the empirical success of non-sparse MKL on this data set.

6.4 Reconstruction of Metabolic Gene Network—A Uniformly Non-Sparse Scenario

In this section, we apply non-sparse MKL to a problem originally studied by Yamanishi et al. (2005). Given 668 enzymes of the yeast *Saccharomyces cerevisiae* and 2782 functional relationships extracted from the KEGG database (Kanehisa et al., 2004), the task is to predict functional relationships for unknown enzymes. We employ the experimental setup of Bleakley et al. (2007) who phrase the task as graph-based edge prediction with local models by learning a model for each of the 668 enzymes. They provided kernel matrices capturing expression data (EXP), cellular localization (LOC), and the phylogenetic profile (PHY); additionally we use the integration of the former 3 kernels (INT) which matches our definition of an unweighted-sum kernel.

Following Bleakley et al. (2007), we employ a 5-fold cross validation; in each fold we train on average 534 enzyme-based models; however, in contrast to Bleakley et al. (2007) we omit enzymes reacting with only one or two others to guarantee well-defined problem settings. As Table 2 shows, this results in slightly better AUC values for single kernel SVMs where the results by Bleakley et al. (2007) are shown in brackets.

As already observed (Bleakley et al., 2007), the unweighted-sum kernel SVM performs best. Although its solution is well approximated by non-sparse MKL using large values of p, ℓ_p-norm MKL is not able to improve on this $p = \infty$ result. Increasing the number of kernels by including recombined and product kernels does improve the results obtained by MKL for small values of p, but the maximal AUC values are not statistically significantly different from those of ℓ_∞-norm MKL. We conjecture that the performance of the unweighted-sum kernel SVM can be explained by all three kernels performing well individually. Their correlation is only moderate, as shown in Fig. 6, suggesting that they contain complementary information. Hence, downweighting one of those three orthogonal kernels leads to a decrease in performance, as observed in our experiments. This explains why ℓ_∞-norm MKL is the best prediction model in this experiment.

6.5 Execution Time

In this section we demonstrate the efficiency of our implementations of non-sparse MKL. We experiment on the MNIST data set,[15] where the task is to separate odd vs. even digits. The digits in this $n = 60{,}000$-elemental data set are of size 28x28 leading to $d = 784$ dimensional examples. We compare our analytical solver for non-sparse MKL (Section 4.3.1–4.3.2) with the state-of-the art

15. This data set is available from `http://yann.lecun.com/exdb/mnist/`.

	AUC $\pm$ stderr	
EXP	71.69 ± 1.1	(69.3 ± 1.9)
LOC	58.35 ± 0.7	(56.0 ± 3.3)
PHY	73.35 ± 1.9	(67.8 ± 2.1)
INT (∞-norm MKL)	$\mathbf{82.94 \pm 1.1}$	$(\mathbf{82.1 \pm 2.2})$
1-norm MKL	75.08 ± 1.4	
$4/3$-norm MKL	78.14 ± 1.6	
2-norm MKL	80.12 ± 1.8	
4-norm MKL	81.58 ± 1.9	
8-norm MKL	81.99 ± 2.0	
10-norm MKL	$\mathbf{82.02 \pm 2.0}$	
Recombined and product kernels		
1-norm MKL	79.05 ± 0.5	
$4/3$-norm MKL	80.92 ± 0.6	
2-norm MKL	81.95 ± 0.6	
4-norm MKL	$\mathbf{83.13 \pm 0.6}$	

Table 2: Results for the reconstruction of a metabolic gene network. Results by Bleakley et al. (2007) for single kernel SVMs are shown in brackets.

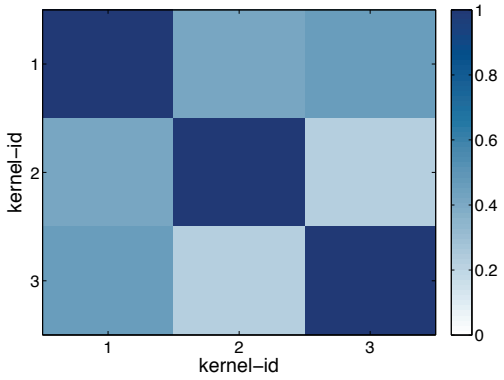


Figure 6: Pairwise alignments of the kernel matrices are shown for the metabolic gene network experiment. From left to right, the ordering of the kernel matrices is EXP, LOC, and PHY. One can see that all kernel matrices are equally correlated. Generally, the alignments are relatively low, suggesting that combining all kernels with equal weights is beneficial.

for ℓ_1-norm MKL, namely SimpleMKL[16] (Rakotomamonjy et al., 2008), HessianMKL[17] (Chapelle and Rakotomamonjy, 2008), SILP-based wrapper, and SILP-based chunking optimization (Sonnenburg et al., 2006a). We also experiment with the analytical method for $p = 1$, although convergence

16. We obtained an implementation from `http://asi.insa-rouen.fr/enseignants/~arakotom/code/`.
17. We obtained an implementation from `http://olivier.chapelle.cc/ams/hessmkl.tgz`.

is only guaranteed by our Theorem 4 for $p > 1$. We also compare to the semi-infinite program (SIP) approach to ℓ_p-norm MKL presented in Kloft et al. (2009a).[18] In addition, we solve standard SVMs[19] using the unweighted-sum kernel (ℓ_∞-norm MKL) as baseline.

We experiment with MKL using precomputed kernels (excluding the kernel computation time from the timings) and MKL based on on-the-fly computed kernel matrices measuring training time *including kernel computations*. Naturally, runtimes of on-the-fly methods should be expected to be higher than the ones of the precomputed counterparts. We optimize all methods up to a precision of 10^{-3} for the outer SVM-ε and 10^{-5} for the "inner" SIP precision, and computed relative duality gaps. To provide a fair stopping criterion to SimpleMKL and HessianMKL, we set their stopping criteria to the relative duality gap of their ℓ_1-norm SILP counterpart. SVM trade-off parameters are set to $C = 1$ for all methods.

6.5.1 SCALABILITY OF THE ALGORITHMS W.R.T. SAMPLE SIZE

Figure 7 (top) displays the results for varying sample sizes and 50 precomputed or on-the-fly computed Gaussian kernels with bandwidths $2\sigma^2 \in 1.2^{0,\ldots,49}$. Error bars indicate standard error over 5 repetitions. As expected, the SVM with the unweighted-sum kernel using precomputed kernel matrices is the fastest method. The classical MKL wrapper based methods, SimpleMKL and the SILP wrapper, are the slowest; they are even slower than methods that compute kernels on-the-fly. Note that the on-the-fly methods naturally have higher runtimes because they do not profit from precomputed kernel matrices.

Notably, when considering 50 kernel matrices of size 8,000 times 8,000 (memory requirements about 24GB for double precision numbers), SimpleMKL is the slowest method: it is more than 120 times slower than the ℓ_1-norm SILP solver from Sonnenburg et al. (2006a). This is because SimpleMKL suffers from having to train an SVM to full precision for each gradient evaluation. In contrast, kernel caching and interleaved optimization still allow to train our algorithm on kernel matrices of size 20000×20000, which would usually not completely fit into memory since they require about 149GB.

Non-sparse MKL scales similarly as ℓ_1-norm SILP for both optimization strategies, the analytic optimization and the sequence of SIPs. Naturally, the generalized SIPs are slightly slower than the SILP variant, since they solve an additional series of Taylor expansions within each θ-step. HessianMKL ranks in between on-the-fly and non-sparse interleaved methods.

6.5.2 SCALABILITY OF THE ALGORITHMS W.R.T. THE NUMBER OF KERNELS

Figure 7 (bottom) shows the results for varying the number of precomputed and on-the-fly computed RBF kernels for a fixed sample size of 1000. The bandwidths of the kernels are scaled such that for M kernels $2\sigma^2 \in 1.2^{0,\ldots,M-1}$. As expected, the SVM with the unweighted-sum kernel is hardly affected by this setup, taking an essentially constant training time. The ℓ_1-norm MKL by Sonnenburg et al. (2006a) handles the increasing number of kernels best and is the fastest MKL method. Non-sparse approaches to MKL show reasonable run-times, being just slightly slower. Thereby the analytical methods are somewhat faster than the SIP approaches. The sparse analytical method

18. The Newton method presented in the same paper performed similarly most of the time but sometimes had convergence problems, especially when $p \approx 1$ and thus was excluded from the presentation.
19. We use SVMlight as SVM-solver.

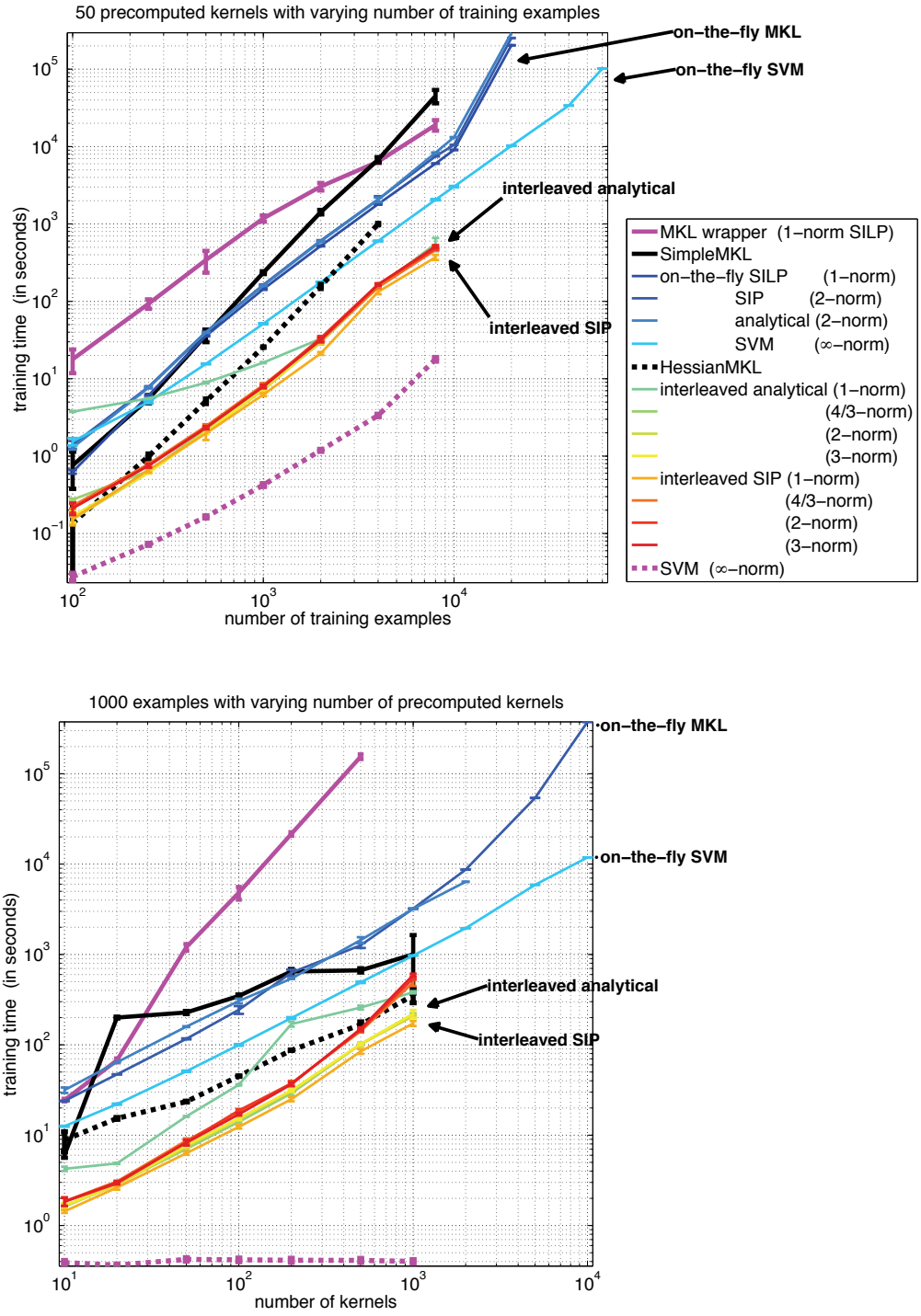


Figure 7: Results of the runtime experiment. Top: Training using fixed number of 50 kernels varying training set size. Bottom: For 1000 examples and varying numbers of kernels. Notice the tiny error bars and that these are log-log plots. The legend is sorted correspondingly to the curves from top to bottom.

performs worse than its non-sparse counterpart; this might be related to the fact that convergence of the analytical method is only guaranteed for $p > 1$. The wrapper methods again perform worst.

However, in contrast to the previous experiment, SimpleMKL becomes more efficient with increasing number of kernels. We conjecture that this is in part owed to the sparsity of the best solution, which accommodates the l_1-norm model of SimpleMKL. But the capacity of SimpleMKL remains limited due to memory restrictions of the hardware. For example, for storing 1,000 kernel matrices for 1,000 data points, about 7.4GB of memory are required. On the other hand, our interleaved optimizers which allow for effective caching can easily cope with 10,000 kernels of the same size (74GB). HessianMKL is considerably faster than SimpleMKL but slower than the non-sparse interleaved methods and the SILP. Similar to SimpleMKL, it becomes more efficient with increasing number of kernels but eventually runs out of memory.

Overall, our proposed interleaved analytic and cutting plane based optimization strategies achieve a speedup of up to one and two orders of magnitude over HessianMKL and SimpleMKL, respectively. Using efficient kernel caching, they allow for truely large-scale multiple kernel learning well beyond the limits imposed by having to precompute and store the complete kernel matrices. Finally, we note that performing MKL with 1,000 precomputed kernel matrices of size 1,000 times 1,000 requires less than 3 minutes for the SILP. This suggests that it focussing future research efforts on improving the accuracy of MKL models may pay off more than further accelerating the optimization algorithm.

7. Conclusion

In the past years, multiple kernel learning research has focused on *accelerating* algorithms for learning convex combinations of kernels. Unfortunately, empirical evidence often showed that sparse MKL-optimized kernel combinations rarely help in practice. By proposing *ℓ_p-norm multiple kernel learning*, conceiving an optimization scheme of unprecedented efficiency, and providing a really efficient implementation (`http://doc.ml.tu-berlin.de/nonsparse_mkl/`), this paper finally makes large-scale MKL practical and profitable.

These advances are founded on our novel general multiple kernel learning framework that subsumes many seemingly different approaches and provides a unifying view and new insights on MKL. In a theoretical analysis, we derived sharp generalization bounds showing that in a non-sparse scenario ℓ_p-norm MKL yields strictly better bounds than ℓ_1-norm MKL and vice versa. However, the difference between the ℓ_p and ℓ_1-norm bounds might not be sufficiently large to completely explain our empirical results. Using the local Rademacher complexity for ℓ_p-norm MKL, one may obtain even tighter bounds, for which the results in Section 5 may serve as a starting point.

In an extensive empirical evaluation, we showed that ℓ_p-norm MKL can significantly improve classification accuracies on diverse and relevant real-world data sets from bioinformatics. Using artificial data, we provided insights by connecting the ℓ_p-norm with the size of the true sparsity pattern. A related—and obtruding!—question is whether the optimality of the parameter p can retrospectively be explained or, more profitably, even be estimated in advance. Clearly, cross-validation based model selection over the choice of p will inevitably tell us which cases call for sparse or non-sparse models. The analyses of our real-world applications suggests that both the correlation amongst the kernels with each other and their correlation with the target (i.e., the amount of discriminative information that they carry) play a role in the distinction of sparse from non-sparse scenarios.

We not only provide a thorough theoretical and empirical analysis, but also contribute an efficient and freely available implementation useful for large-scale real-world applications.

Finally, we would like to note that it may be worthwhile to rethink the current strong preference for sparse models in the scientific community. For example, already weak connectivity in a causal graphical model may be sufficient for all variables to be required for optimal predictions, and even the prevalence of sparsity in causal flows is being questioned (e.g., for the social sciences Gelman, 2010 argues that "There are (almost) no true zeros"). A main reason for favoring sparsity may be the presumed interpretability of sparse models. However, in general sparse MKL solutions are sensitive to kernel normalization, and in particular in the presence of strongly correlated kernels the selection of kernels may be somewhat arbitrary. This puts the interpretation of sparsity patterns in doubt, and it may be more honest to focus on predictive accuracy. In this respect we demonstrate that non-sparse models may improve quite impressively over sparse ones.

Acknowledgments

The authors wish to thank Vojtech Franc, Peter Gehler, Pavel Laskov for stimulating discussions; and Yoann Fabre, Chris Hinrichs, and Klaus-Robert Müller for helpful comments on the manuscript. We thank Motoaki Kawanabe for a valuable suggestion that improved the design of the toy experiment and Gilles Blanchard for a comment that lead to a tighter generalization bound. We acknowledge Peter L. Bartlett and Ulrich Rückert for contributions to parts of an earlier version of the theoretical analysis that appeared at ECML 2010. We thank the anonymous reviewers for comments and suggestions that helped to improve the manuscript. This work was supported in part by the German Bundesministerium für Bildung und Forschung (BMBF) under the project REMIND (FKZ 01-IS07007A), and by the FP7-ICT program of the European Community, under the PASCAL2 Network of Excellence, ICT-216886. Sören Sonnenburg acknowledges financial support by the German Research Foundation (DFG) under the grant MU 987/6-1 and RA 1894/1-1, and Marius Kloft acknowledges a scholarship by the German Academic Exchange Service (DAAD).

Appendix A. Switching Between Tikhonov and Ivanov Regularization

In this appendix, we show a useful result that justifies switching from Tikhonov to Ivanov regularization and vice versa, if the bound on the regularizing constraint is tight. It is the key ingredient of the proof of Theorem 1. We state the result for arbitrary convex functions, so that it can be applied beyond the multiple kernel learning framework of this paper.

Proposition 12 *Let $D \subset \mathbb{R}^d$ be a convex set, let $f, g : D \to \mathbb{R}$ be convex functions. Consider the convex optimization tasks*

$$\min_{x \in D} \quad f(x) + \sigma g(x), \tag{29}$$

$$\min_{x \in D : g(x) \leq \tau} \quad f(x). \tag{30}$$

Assume that the minima exist and that a constraint qualification holds in (30)*, which gives rise to strong duality, for example, that Slater's condition is satisfied. Furthermore assume that the*

constraint is active at the optimal point, that is,

$$\inf_{x\in D} f(x) \;<\; \inf_{x\in D: g(x)\leq \tau} f(x). \tag{31}$$

Then we have that for each $\sigma > 0$ *there exists* $\tau > 0$ *—and vice versa—such that OP* (29) *is equivalent to OP* (30)*, that is, each optimal solution of one is an optimal solution of the other, and vice versa.*

Proof
(a). Let be $\sigma > 0$ and x^* be the optimal of (29). We have to show that there exists a $\tau > 0$ such that x^* is optimal in (30). We set $\tau = g(x^*)$. Suppose x^* is not optimal in (30), that is, it exists $\tilde{x} \in D : g(\tilde{x}) \leq \tau$ such that $f(\tilde{x}) < f(x^*)$. Then we have

$$f(\tilde{x}) + \sigma g(\tilde{x}) < f(x^*) + \sigma\tau,$$

which by $\tau = g(x^*)$ translates to

$$f(\tilde{x}) + \sigma g(\tilde{x}) < f(x^*) + \sigma g(x^*).$$

This contradics the optimality of x^* in (29), and hence shows that x^* is optimal in (30), which was to be shown.
(b). Vice versa, let $\tau > 0$ be x^* optimal in (30). The Lagrangian of (30) is given by

$$\mathcal{L}(\sigma) = f(x) + \sigma\,(g(x) - \tau)\,, \quad \sigma \geq 0.$$

By strong duality x^* is optimal in the saddle point problem

$$\sigma^* := \operatorname*{argmax}_{\sigma\geq 0} \min_{x\in D} \quad f(x) + \sigma\,(g(x) - \tau)\,,$$

and by the strong max-min property (cf. Boyd and Vandenberghe, 2004, p. 238) we may exchange the order of maximization and minimization. Hence x^* is optimal in

$$\min_{x\in D} \quad f(x) + \sigma^*\,(g(x) - \tau)\,. \tag{32}$$

Removing the constant term $-\sigma^*\tau$, and setting $\sigma = \sigma^*$, we have that x^* is optimal in (29), which was to be shown. Moreover by (31) we have that

$$x^* \neq \operatorname*{argmin}_{x\in D} f(x),$$

and hence we see from Equation (32) that $\sigma^* > 0$, which completes the proof of the proposition. ■

References

T. Abeel, Y. Van de Peer, and Y. Saeys. Towards a gold standard for promoter prediction evaluation. *Bioinformatics*, 2009.

J. Aflalo, A. Ben-Tal, C. Bhattacharyya, J. Saketha Nath, and S. Raman. Variable sparsity kernel learning—algorithms and applications. *Journal of Machine Learning Research*, 2011. To appear.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 105–112, 2009.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proc. 21st ICML*. ACM, 2004.

V. B. Bajic, S. L. Tan, Y. Suzuki, and S. Sugano. Promoter prediction analysis on the whole human genome. *Nature Biotechnology*, 22(11):1467–1473, 2004.

P.L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, November 2002.

D.P. Bertsekas. *Nonlinear Programming, Second Edition*. Athena Scientific, Belmont, MA, 1999.

K. Bleakley, G. Biau, and J.-P. Vert. Supervised reconstruction of biological networks with local models. *Bioinformatics*, 23:i57–i65, 2007.

O. Bousquet and D.J.L. Herrmann. On the complexity of learning the kernel matrix. In *Advances in Neural Information Processing Systems*, 2002.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors, *Advanced Lectures on Machine Learning*, volume 3176 of *Lecture Notes in Computer Science*, pages 169–207. Springer Berlin / Heidelberg, 2004.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambrigde University Press, Cambridge, UK, 2004.

O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 2006.

O. Chapelle and A. Rakotomamonjy. Second order optimization of kernel parameters. In *Proc. of the NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, 2008.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1):131–159, 2002.

C. Cortes, A. Gretton, G. Lanckriet, M. Mohri, and A. Rostamizadeh. Proceedings of the NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels, 2008. URL `http://www.cs.nyu.edu/learning_kernels`.

C. Cortes, M. Mohri, and A. Rostamizadeh. L2 regularization for learning kernels. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2009a.

C. Cortes, M. Mohri, and A. Rostamizadeh. Learning non-linear combinations of kernels. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 396–404, 2009b.

C. Cortes, M. Mohri, and A. Rostamizadeh. Generalization bounds for learning kernels. In *Proceedings, 27th ICML*, 2010a.

C. Cortes, M. Mohri, and A. Rostamizadeh. Two-stage learning kernel algorithms. In *Proceedings of the 27th Conference on Machine Learning (ICML 2010)*, 2010b.

N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On kernel-target alignment. In *Advances in Neural Information Processing Systems*, 2002.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Number 31 in Applications of Mathematics. Springer, New York, 1996.

R. 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.

R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using the second order information for training support vector machines. *Journal of Machine Learning Research*, 6:1889–1918, 2005.

V. Franc and S. Sonnenburg. OCAS optimized cutting plane algorithm for support vector machines. In *Proceedings of the 25nd International Machine Learning Conference*. ACM Press, 2008.

P. V. Gehler and S. Nowozin. Infinite kernel learning. In *Proceedings of the NIPS 2008 Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, 2008.

A. Gelman. Causality and statistical learning. *American Journal of Sociology*, 0, 2010.

G.H. Golub and C.F. van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, London, 3rd edition, 1996.

M. Gönen and E. Alpaydin. Localized multiple kernel learning. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 352–359, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: http://doi.acm.org/10.1145/1390156.1390201.

V.K. Ivanov, V.V. Vasin, and V.P. Tanana. *Theory of Linear Ill-Posed Problems and its application*. VSP, Zeist, 2002.

S. Ji, L. Sun, R. Jin, and J. Ye. Multi-label multiple kernel learning. In *Advances in Neural Information Processing Systems*, 2009.

T. Joachims. Making large–scale SVM learning practical. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

S. M. Kakade, S. Shalev-Shwartz, and A. Tewari. Applications of strong convexity–strong smoothness duality to learning with matrices. *CoRR*, abs/0910.0610, 2009.

M. Kanehisa, S. Goto, S. Kawashima, Y. Okuno, and M. Hattori. The KEGG resource for deciphering the genome. *Nucleic Acids Res*, 32:D277–D280, 2004.

G. Kimeldorf and G. Wahba. Some results on tchebycheffian spline functions. *J. Math. Anal. Applic.*, 33:82–95, 1971.

M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg. Non-sparse multiple kernel learning. In *Proc. of the NIPS Workshop on Kernel Learning: Automatic Selection of Kernels*, dec 2008.

M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K.-R. Müller, and A. Zien. Efficient and accurate lp-norm multiple kernel learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 997–1005. MIT Press, 2009a.

M. Kloft, S. Nakajima, and U. Brefeld. Feature selection for density level-sets. In W. L. Buntine, M. Grobelnik, D. Mladenic, and J. Shawe-Taylor, editors, *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, pages 692–704, 2009b.

M. Kloft, U. Rückert, and P. L. Bartlett. A unifying view of multiple kernel learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, 2010. To appear. ArXiv preprint: `http://arxiv.org/abs/1005.0437`.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30:1–50, 2002.

G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. *JMLR*, 5:27–72, 2004.

D.C. Liu and J. Nocedal. On the limited memory method for large scale optimization. *Mathematical Programming B*, 45(3):503–528, 1989.

P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, volume 2, no. 1, April 1936.

M. Markou and S. Singh. Novelty detection: a review – part 1: statistical approaches. *Signal Processing*, 83:2481–2497, 2003a.

M. Markou and S. Singh. Novelty detection: a review – part 2: neural network based approaches. *Signal Processing*, 83:2499–2521, 2003b.

C. A. Micchelli and M. Pontil. Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Neural Networks*, 12(2):181–201, May 2001.

S. Nash and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, New York, NY, 1996.

J. S. Nath, G. Dinesh, S. Ramanand, C. Bhattacharyya, A. Ben-Tal, and K. R. Ramakrishnan. On the algorithmics and applications of a mixed-norm based kernel learning formulation. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 844–852, 2009.

A. Nemirovski. Prox-method with rate of convergence o(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15:229–251, 2004.

C. S. Ong and A. Zien. An Automated Combination of Kernels for Predicting Protein Subcellular Localization. In *Proc. of the 8th Workshop on Algorithms in Bioinformatics*, 2008.

C. S. Ong, A. J. Smola, and R. C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 6:1043–1071, 2005.

S. Özögür-Akyüz and G.W. Weber. Learning with infinitely many kernels via semi-infinite programming. In *Proceedings of Euro Mini Conference on Continuous Optimization and Knowledge Based Technologies*, 2008.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. More efficiency in multiple kernel learning. In *ICML*, pages 775–782, 2007.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

R. M. Rifkin and R. A. Lippert. Value regularization and Fenchel duality. *J. Mach. Learn. Res.*, 8: 441–479, 2007.

V. Roth and B. Fischer. Improved functional prediction of proteins by learning kernel combinations in multilabel settings. *BMC Bioinformatics*, 8(Suppl 2):S12, 2007. ISSN 1471-2105.

V. Roth and B. Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML 2008)*, volume 307, pages 848–855. ACM, 2008.

E. Rubinstein. Support vector machines via advanced optimization techniques. Master's thesis, Faculty of Electrical Engineering, Technion, 2005, Nov 2005.

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

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

B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

B. Schölkopf, S. Mika, C.J.C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A.J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5): 1000–1017, September 1999.

B. Schölkopf, J. Platt, J. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

S. Sonnenburg, G. Rätsch, and C. Schäfer. Learning interpretable SVMs for biological sequence classification. In *RECOMB 2005, LNBI 3500*, pages 389–407. Springer-Verlag Berlin Heidelberg, 2005.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, July 2006a.

S. Sonnenburg, A. Zien, and G. Rätsch. Arts: Accurate recognition of transcription starts in human. *Bioinformatics*, 22(14):e472–e480, 2006b.

S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN Machine Learning Toolbox. *Journal of Machine Learning Research*, 2010.

J. M. Steele. *The Cauchy-Schwarz Master Class: An Introduction to the Art of Mathematical Inequalities*. Cambridge University Press, New York, NY, USA, 2004. ISBN 052154677X.

M. Stone. Cross-validatory choice and assessment of statistical predictors (with discussion). *Journal of the Royal Statistical Society*, B36:111–147, 1974.

Y. Suzuki, R. Yamashita, K. Nakai, and S. Sugano. dbTSS: Database of human transcriptional start sites and full-length cDNAs. *Nucleic Acids Research*, 30(1):328–331, 2002.

M. Szafranski, Y. Grandvalet, and A. Rakotomamonjy. Composite kernel learning. In *Proceedings of the International Conference on Machine Learning*, 2008.

M. Szafranski, Y. Grandvalet, and A. Rakotomamonjy. Composite kernel learning. *Mach. Learn.*, 79(1-2):73–103, 2010. ISSN 0885-6125. doi: http://dx.doi.org/10.1007/s10994-009-5150-6.

D.M.J. Tax and R.P.W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20 (11–13):1191–1199, 1999.

A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed problems*. W. H. Winston, Washington, 1977.

M. Varma and B. R. Babu. More generality in efficient multiple kernel learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 1065–1072, New York, NY, USA, 2009. ACM.

M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *IEEE 11th International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

Z. Xu, R. Jin, I. King, and M. 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, 2009.

Z. Xu, R. Jin, H. Yang, I. King, and M. Lyu. Simple and efficient multiple kernel learning by group lasso. In *Proceedings of the 27th Conference on Machine Learning (ICML 2010)*, 2010.

Y. Yamanishi, , J.-P. Vert, and M. Kanehisa. Supervised enzyme network inference from the integration of genomic data and chemical information. *Bioinformatics*, 21:i468–i477, 2005.

Y. Ying, C. Campbell, T. Damoulas, and M. Girolami. Class prediction from disparate biological data sources using an iterative multi-kernel algorithm. In Visakan Kadirkamanathan, Guido Sanguinetti, Mark Girolami, Mahesan Niranjan, and Josselin Noirel, editors, *Pattern Recognition in Bioinformatics*, volume 5780 of *Lecture Notes in Computer Science*, pages 427–438. Springer Berlin / Heidelberg, 2009.

S. Yu, T. Falck, A. Daemen, L.-C. Tranchevent, J. Suykens, B. De Moor, and Y. Moreau. L2-norm multiple kernel learning and its application to biomedical data fusion. *BMC Bioinformatics*, 11 (1):309, 2010. ISSN 1471-2105.

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.

A. Zien and C. S. Ong. Multiclass multiple kernel learning. In *Proceedings of the 24th international conference on Machine learning (ICML)*, pages 1191–1198. ACM, 2007.

Journal of Machine Learning Research 12 (2011) 999-1024 Submitted 5/10; Revised 11/10; Published 3/11

Unsupervised Similarity-Based Risk Stratification for Cardiovascular Events Using Long-Term Time-Series Data

Zeeshan Syed ZHS@EECS.UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2121, USA

John Guttag GUTTAG@CSAIL.MIT.EDU
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139-4307, USA

Editor: Carla Brodley

Abstract

In medicine, one often bases decisions upon a comparative analysis of patient data. In this paper, we build upon this observation and describe similarity-based algorithms to risk stratify patients for major adverse cardiac events. We evolve the traditional approach of comparing patient data in two ways. First, we propose similarity-based algorithms that compare patients in terms of their long-term physiological monitoring data. Symbolic mismatch identifies functional units in long-term signals and measures changes in the morphology and frequency of these units across patients. Second, we describe similarity-based algorithms that are unsupervised and do not require comparisons to patients with known outcomes for risk stratification. This is achieved by using an anomaly detection framework to identify patients who are unlike other patients in a population and may potentially be at an elevated risk. We demonstrate the potential utility of our approach by showing how symbolic mismatch-based algorithms can be used to classify patients as being at high or low risk of major adverse cardiac events by comparing their long-term electrocardiograms to that of a large population. We describe how symbolic mismatch can be used in three different existing methods: one-class support vector machines, nearest neighbor analysis, and hierarchical clustering. When evaluated on a population of 686 patients with available long-term electrocardiographic data, symbolic mismatch-based comparative approaches were able to identify patients at roughly a two-fold increased risk of major adverse cardiac events in the 90 days following acute coronary syndrome. These results were consistent even after adjusting for other clinical risk variables.

Keywords: risk stratification, cardiovascular disease, time-series comparison, symbolic analysis, anomaly detection

1. Introduction

In medicine, as in many other disciplines, decisions are often based upon a comparative analysis. Patients are given treatments that worked in the past on apparently similar conditions. When given simple data (e.g., demographics, comorbidities, and laboratory values) such comparisons are relatively straightforward. For more complex data, such as continuous long-term signals recorded during physiological monitoring, they are harder. Comparing such time-series is made challenging by three factors: the need to capture the many different changes that occur over long periods, for

©2011 Zeeshan Syed and John Guttag.

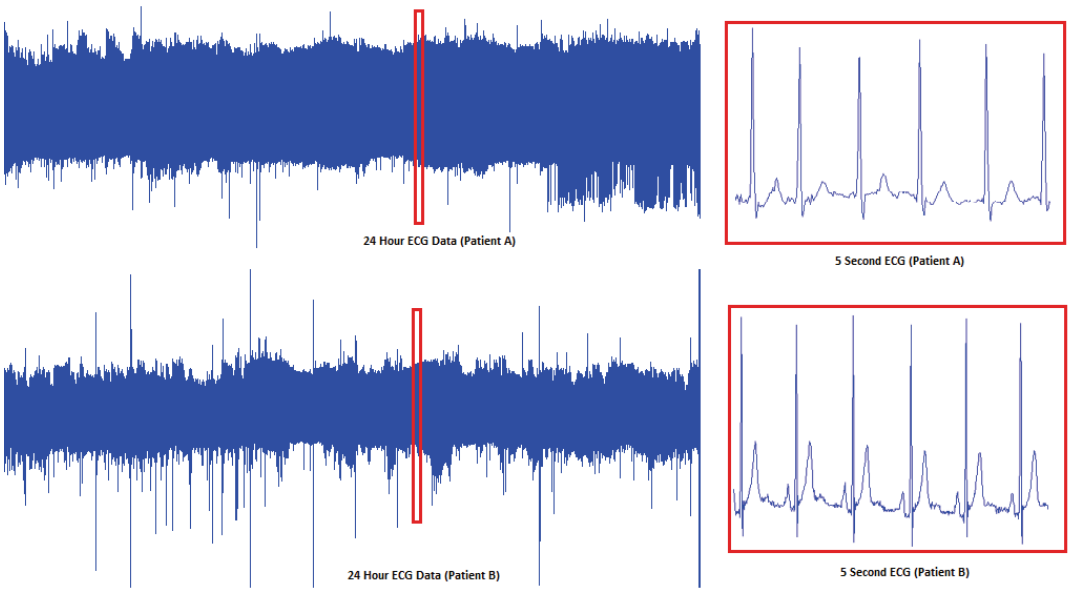


Figure 1: 24 hour ECG signals from two patients. Each time-series is over ten million samples long and contains patient-specific differences in the shape, frequency and time scale of activity over the recording duration. These differences need to be captured while comparing these data.

example, in the shape, frequency, or time scale of activity; the need to efficiently compare very long signals across a large number of patients; and the need to deal with patient-specific differences (Figure 1).

Despite these challenges, comparative analyses of long-term physiological time-series can potentially offer clinically useful prognostic information. While there is an extensive body of research focussed on comparing relatively short time-series, including measures such as dynamic time warping (Keogh and Pazzani, 2001; Keogh and Ratanamahatana, 2005), longest common subsequence (Vlachos et al., 2002), edit distance with real penalty (Cheng and Ng, 2004), sequence weighted alignment (Morse and Patel, 2007), spatial assembling distance (Chen et al., 2007), this work does not directly focus on comparing very long time-series (e.g., millions of samples long). In this paper, we investigate the hypothesis that comparative analyses of long-term physiological activity can aid risk stratification and present symbolic mismatch as a way to quantify differences between the physiological signals of patients. Symbolic mismatch compares long-term time-series by mapping the original signals into a symbolic domain and quantifying differences between the morphology and frequency of prototypical functional units. The use of symbolization is an abstraction process that greatly reduces the size of the data to be compared. For example, comparing the long-term electrocardiographic (ECG) activity between two patients may involve comparing over a hundred thousand beats (with each beats having roughly 500 samples per beat). Using symbolization to reduce this data to a small number of representative units can greatly decrease the size of this problem. This reduction allows for symbolic mismatch to be useful in analyzing very long time-series.

We describe how this measure can be modified in a reasonably simple manner for use with kernel classifiers.

We also present different ways in which symbolic mismatch can be used to identify high risk patients in a population. The methods we propose are motivated by the observation that high risk patients typically constitute a small minority in a population. For example, cardiac mortality over a 90 day period following acute coronary syndrome (ACS) was reported to be 1.79% for the SYMPHONY trial involving 14,970 patients (Newby et al., 2003) and 1.71% for the DISPERSE2 trial with 990 patients (Cannon et al., 2007). The rate of myocardial infarction (MI) over the same period for the two trials was 5.11% for the SYMPHONY trial and 3.54% for the DISPERSE2 trial. Our hypothesis is that these patients can be discovered as anomalies in the population, that is, their physiological activity over long periods of time is dissimilar to the majority of the patients in the population. In contrast to algorithms that require labeled training data, we propose identifying these patients using unsupervised approaches based on three methods previously reported in the literature: one-class support vector machines (SVMs) (Scholkopf et al., 2001), nearest neighbor analysis (Cover and Hart, 1967) and hierarchical clustering (Ward Jr, 1963).

In the remainder of this paper, we describe our work in the context of risk stratification for cardiovascular disease. Cardiovascular disease is the leading cause of death globally and causes roughly 17 million deaths each year (World Health Organization, 2009). By 2030, this number is expected to grow to nearly 24 million deaths annually (World Health Organization, 2009). Most of these cases are expected to be the result of coronary attacks. Despite improvements in survival rates, in the United States, 1 in 4 men and 1 in 3 women still die within a year of a recognized first heart attack (Mackay et al., 2004). This risk of death can be substantially lowered with an appropriate choice of treatment (e.g., drugs to lower cholesterol and blood pressure, aspirin; operations such as coronary artery bypass graft and balloon angioplasty; and medical devices such as pacemakers and implantable cardioverter defibrillators) (World Health Organization, 2009). However, matching patients with treatments that are appropriate for their risk has proven to be challenging (Bailey et al., 2001; Lopera and Curtis, 2009). Existing techniques based upon conventional medical knowledge continue to be inadequate for risk stratification. This leads us to explore methods with few *a priori* assumptions. We focus, in particular, on identifying patients at risk of major adverse cardiac events (death, myocardial infarction and severe recurrent ischemia) following coronary attacks. This work uses long-term ECG signals recorded during patient admission for acute coronary syndrome. These signals are routinely collected, potentially allowing for the work presented here to be deployed easily without imposing additional needs on patients, caregivers, or the healthcare infrastructure. We demonstrate that it is possible to do long-term ECG-based risk stratification without defining a set of features, performing cross-patient symbol or feature alignment, or having any labeled data.

The main contributions of our work are: (1) we describe a novel approach for cardiovascular risk stratification that is complementary to existing clinical approaches, (2) we explore the idea of similarity-based clinical risk stratification where patients are categorized in terms of their similarities rather than specific features based on prior knowledge, (3) we develop the hypothesis that patients at future risk of adverse outcomes can be detected using an unsupervised approach as outliers in a population, (4) we present symbolic mismatch, as a way to efficiently compare very long time-series without first reducing them to a set of features or requiring symbol registration across patients, and (5) we present a rigorous evaluation of unsupervised similarity-based risk stratifying using long-term data in a population of 700 patients with detailed admissions and follow-up data.

While this manuscript focuses on cardiovascular disease, we believe that the ideas presented here can potentially be extended to other data sets in a relatively straightforward manner.

2. Background

We start by briefly reviewing the clinical background for our research. We focus, in particular, on aspects of cardiac function related to electrophysiology. This is followed by a discussion of acute coronary syndromes (ACS) and a summary of recently proposed long-term ECG metrics for risk stratification following ACS. Finally, we present a short overview of survival analysis methods used to evaluate metrics for risk stratification.

2.1 Electrocardiogram

An electrocardiogram (ECG) is a continuous recording of the electrical activity of the heart muscle or myocardium (Lilly, 2007). A cardiac muscle cell at rest maintains a negative voltage with respect to the outside of the cell. During depolarization, this voltage increases and may even become positive. Consequently, when depolarization is propagating through a cell, there exists a potential difference on the membrane between the part of the cell that has been depolarized and the part of the cell at resting potential. After the cell is completely depolarized, its membrane is uniformly charged again, but at a more positive voltage than initially. The reverse situation takes place during repolarization, which returns the cell to baseline.

These changes in potential, summed over many cells, can be measured by electrodes placed on the surface of the body. For any pair of electrodes, a voltage is recorded whenever the direction of depolarization (or repolarization) is aligned with the line connecting the two electrodes. The sign of the voltage indicates the direction of depolarization, and the axis of the electrode pair is termed the lead. Multiple electrodes along different axes can be used so that the average direction of depolarization, as a three-dimensional vector, can be reconstructed from the ECG tracings. However, such multi-lead data is not always available, especially for portable ECG monitors that maximize battery life by reducing the number of electrodes used. Much of our work is therefore designed for the single ECG lead case. As we show subsequently, there is sufficient information even within a single lead of ECG to risk stratify patients.

ECG data is routinely recorded for hospitalized patients, since it is useful for determination of heart rate and pulse, and for the detection of progressive cardiac disease and complicating arrhythmias (Goldstein, 1997). ECG has the advantage of being easy to acquire; the electrical activity of the heart can be measured on the surface of the body in an inexpensive and non-invasive manner. In an in-patient setting, the ECG is typically captured by bedside monitors. In an out-patient setting, a Holter monitor (a portable ECG device worn by patients) can record data continuously over multiple days.

The ECG is a quasi-periodic signal (i.e., corresponding to the quasi-periodic nature of cardiac activity). Three major segments can be identified in a normal ECG. The *P wave* is associated with depolarization of cardiac cells in the upper two chambers of the heart (i.e., the atria). The *QRS complex* (comprising the Q, R and S waves) is associated with depolarization of cardiac cells in the lower two chambers of the heart (i.e., the ventricles). The *T wave* is associated with repolarization of the cardiac cells in the ventricles. The QRS complex is larger than the P wave because the ventricles are much larger than the atria. The QRS complex also coincides with the repolarization of the atria, which is therefore usually not seen on the ECG. The T wave has a larger width and

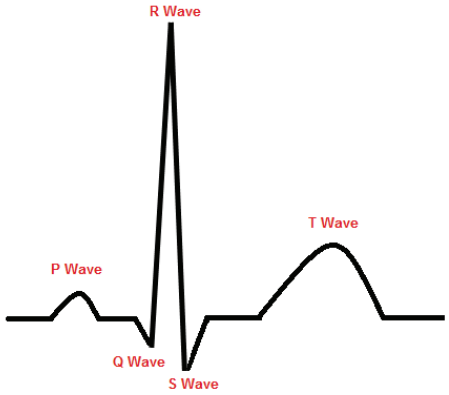


Figure 2: Schematic representation of the normal ECG for a single heart beat.

smaller amplitude than the QRS complex because repolarization takes longer than depolarization (Lilly, 2007). Figure 2 presents a schematic representation of the normal ECG.

2.2 Acute Coronary Syndromes

In our research, we mainly focus on identifying high risk patients following acute coronary syndrome (ACS), an umbrella term covering clinical symptoms compatible with reduced blood supply to the heart (i.e., myocardial ischemia). Heart attacks and unstable angina are included in this group. An ACS is an event in which the blood supply to the myocardium is blocked or severely reduced. The most common symptom of ACS is unusual and unprovoked chest pain, but this may often be absent (most notably in patients with diabetes who experience "silent" heart attacks). Other symptoms may include shortness of breath, profuse sweating, and nausea.

An ACS is usually caused by the rupture of an atherosclerotic plaque producing a clot within a coronary artery. This restricts blood flow to the heart, causing ischemia and potentially cell death in the myocardium. Various sub-classifications of ACS are distinguished by the presence of myocardial necrosis (cell death) and by ECG diagnosis (Lilly, 2007).

Unstable angina refers to an ACS event in which necrosis does not occur, while *myocardial infarction* (MI) refers to one in which it does. Usually, we distinguish between ACS where the ECG shows an ST segment elevation (indicative of complete occlusion of an artery with myocardial necrosis) and ACS where the ECG does not show an ST segment elevation (indicative of partal occlusion of an artery with or without myocardial necrosis). Patients with ST segment elevation are given the diagnosis of ST-elevation MI (STEMI) while patients without ST segment elevation are given the diagnosis of non-ST-elevation ACS (NSTEACS). NSTEACS can correspond to either unstable angina or non-ST-elevation MI (NSTEMI). Patients with STEMI are typically at a higher risk relative to patients with NSTEACS. The differentiation between whether the NSTEACS corresponds to unstable angina or NSTEMI is done on the basis of whether necrosis occurs. This involves blood tests to measure the levels of two serum biomarkers, cardiac-specific troponin and creatine kinase MB, which are chemicals released into the bloodstream when myocardial cells die.

2.3 Post-ACS Risk Stratification and Long-Term ECG Techniques

Since patients who experience ACS remain at an elevated risk of death even after receiving treatment for the initial index event (Newby et al., 2003), post-ACS risk stratification is an important clinical step in determining which patients should be monitored and treated more (or less) aggressively subsequently. Roughly speaking, the goal of risk stratification is to identify groups of patients within the post-ACS population with different rates of adverse outcomes despite receiving similar care. This information can provide an important basis to deliver customized care and to perform clinical cost-benefit analyses.

A number of different methods have been proposed to risk stratify patients following ACS with an excellent review provided by Breall and Simons (2010) and Alpert (2010). We focus here on recent techniques for risk stratification based on information in long-term ECG. A variety of methods have been proposed that assess risk based on automated analysis of long-term ECG data collected in the hours or days following admission. Such data is routinely collected during a patient's stay and therefore these risk assessments can be obtained at almost no additional cost. We discuss three ECG-based methods that have been proposed in the literature: *heart rate variability* (HRV) (Malik et al., 1996; Kleiger et al., 2005), *heart rate turbulence* (HRT) (Barthel et al., 2003), and *deceleration capacity* (DC) (Bauer et al., 2006). Each of these measures has been shown to correlate with an increased risk of adverse events in the period following an ACS. One additional long-term ECG-based risk stratification technique, T-wave alternans (TWA) (Smith et al., 1988; Rosenbaum et al., 1994), has also shown promise. However, evaluating TWA typically requires the use of specialized recording equipment, patient maneuvers or chemical agents to elevate heart rate, and input by a human expert for signal selection. As a result we do not consider TWA in our present study, while focusing on automated methods that can be applied to data collected routinely from post-ACS patients.

The class of ECG-based risk stratification techniques that has been discussed most extensively in the literature is HRV (Malik et al., 1996; Kleiger et al., 2005). The theory underlying HRV-based techniques is that in healthy people, the body should continuously compensate for changes in oxygen demand by changing the heart rate. The heart rate should also change as a result of physiological phenomena such as respiratory sinus arrhythmia (Lilly, 2007). A heart rate that changes little suggests that the heart or its control systems are not actively responding to stimuli.

HRT (Barthel et al., 2003) is related to HRV in that it studies the autonomic tone of patients (i.e., control of the heart rate by the nervous system). HRT studies the return to equilibrium of the heart rate after premature beats. Typically, following a premature beat there is a brief speed-up in heart rate followed by a slow decrease back to the baseline rate. This corresponds to the "turbulence" in the heart rate and is present in patients with a healthy autonomic nervous system. HRT is essentially a reflex phenomenon. When a premature beat interrupts the normal cardiac cycle, the ventricles have not had time to fill to their normal levels, resulting in a weaker pulse. This triggers pressure reflex mechanisms that compensate by increasing the heart rate. This compensatory increase in heart rate causes blood pressure to overcompensate and actives the pressure reflex in reverse. Patients that have diminished HRT responses after premature beats are believed to be at high risk due to abnormal nervous control of the cardiovascular system.

DC (Bauer et al., 2006) is an extension of work on HRV and HRT, and also studies information in the heart rate signal. The theory underlying DC is that decreased decelerations in the heart rate suggest an increased unresponsiveness of the heart to cardioprotective signals from the vagal system

for the heart to slow down. This is often the first line of defense against major adverse events such as fatal arrhythmias.

2.4 Survival Analysis

Metrics for risk stratification are typically evaluated using survival analysis techniques. These methods study the rates of adverse outcomes in patients assigned to different groups (e.g., high and low risk groups in the context of risk stratification post-ACS). In general, such analyses can be carried out in terms of sensitivity and specificity. However, data from clinical studies often consists of a mix of patients who either remain event free throughout the duration of the study, experience events at fairly different times within the study, or leave the study before it is complete (a phenomenon termed censoring). Survival analysis focuses on using information in all these cases, that is, by factoring in both the timing of events in patients who experience adverse outcomes, and by using data from patients who leave the study early for the period during which they participated.

Survival data is commonly visualized as a Kaplan-Meier survival curve (Efron, 1988), which reflects the event rate over time in patients belonging to different groups. We present examples of Kaplan-Meier survival curves subsequently in this manuscript. Visually observed differences between Kaplan-Meier survival curves (i.e., differences in the rates of events over time in patients belonging to two or more groups) can be quantified through a variety of methods. The Cox proportional hazards test is most widely used (Cox, 1972) and corresponds to a regression-based approach to determine how the relative risk between populations varies over time in response to explanatory covariates. The outcomes of this analysis are typically a hazard ratio estimating the multiplicative increase in the rate of events over time between populations, and a measure of the statistical strength of this estimate (a 95% confidence interval for the hazard ratio or a p value). Findings are usually considered to be significant if the p value is less than 0.05 (corresponding to a 5% chance of rejecting the null hypothesis, that is, a Type I error).

3. Symbolic Mismatch

In this section, we describe the process through which symbolic mismatch is computed between a pair of long-term ECG time-series. The use of symbolic mismatch for risk stratification is presented in Section 4.

3.1 Symbolization

As a first step, the ECG signal z_m for each patient $m = 1, ..., n$ is symbolized using the technique proposed by Syed et al. (2007). To segment the ECG signal into beats, we use two open-source QRS detection algorithms (Hamilton and Tompkins, 1986; Zong et al., 2003). QRS complexes are marked at locations where both algorithms agree. A variant of dynamic time-warping (DTW) (Myers and Rabiner, 1981) is then used to quantitatively measure differences in morphology between beats. Using this information, beats with distinct morphologies are partitioned into groups, with each group being assigned a unique label or symbol. This is done by means of a Max-Min iterative clustering algorithm that starts by choosing the first observation as the first centroid, c_1, and initializes the set S of centroids to $\{c_1\}$. During the i-th iteration, c_i is chosen such that it maximizes the minimum difference between c_i and observations in S:

$$c_i = \arg\max_{x \notin S} \min_{y \in S} C(x,y)$$

where $C(x,y)$ is the DTW difference between x and y. The set S is incremented at the end of each iteration such that $S = S \cup c_i$.

The number of clusters discovered by Max-Min clustering is chosen by iterating until the maximized minimum difference falls below a threshold θ (chosen empirically as the 'knee' of the maximized minimum difference curve). At this point, the set S comprises the centroids for the clustering process, and the final assignment of beats to clusters proceeds by matching each beat to its nearest centroid. Each set of beats assigned to a centroid constitutes a unique cluster. The final number of clusters, γ, obtained using this process depends on the separability of the underlying data.

Intuitively, the Max-Min clustering algorithms attempts to partition the data into groups such that the similarity of points within the same groups is minimized, while the similarity of points within different groups is maximized. Theoretical analyses of Max-Min cut-based methods show that this approach leads to more balanced separations of the data than other approaches (Ding et al., 2001).

The overall effect of DTW-based partitioning of beats is to transform the original raw ECG signal into a sequence of symbols, that is, into a sequence of labels corresponding to the different beat morphology classes that occur in the signal. Our approach differs from the methods typically used to annotate ECG signals in two ways. First, we avoid using specialized knowledge to partition beats into known clinical classes. There is a set of generally accepted labels that cardiologists use to differentiate distinct kinds of heart beats. For example, the Association for the Advancement of Medical Instrumentation (AAMI) standards define five basic beat classes (AAMI, 1998, 1987; de Chazal et al., 2004), while the Physionet MIT-BIH Arrhythmia database sub-divides these classes to produce 18 different heart beat labels (Physionet, 2005). However, in some cases, even finer distinctions than provided by these labels can be clinically relevant (Syed et al., 2007). Our use of beat clustering rather than beat classification allows us to identify a set of characteristic morphology classes dynamically from the data that capture these finer-grained distinctions. Second, our approach does not involve extracting features (e.g., the length of the beat or the amplitude of the P wave) from individual beats. Instead, our clustering algorithm compares the entire raw morphology of pairs of beats. This approach is potentially advantageous, because it does not assume *a priori* knowledge about what aspects of a beat are most relevant. It can also be easily extended to other time-series data (e.g., blood pressure waveforms and respiratory activity).

3.2 Measuring Mismatch in Symbolic Representations

Denoting the set of symbol centroids for patient p as S_p and the set of frequencies with which these symbols occur in the electrocardiogram as F_p (for patient q an analogous representation is adopted), we define the symbolic mismatch between the long-term ECG time-series for patients p and q as:

$$\psi_{p,q} = \sum_{p_i \in S_p} \sum_{q_j \in S_q} C(p_i, q_j) F_p[p_i] F_q[q_j] \tag{1}$$

where $C(p_i, q_j)$ corresponds to the DTW cost of aligning the centroids of symbol classes p_i and q_j.

Intuitively, the symbolic mismatch between patients p and q corresponds to an estimate of the expected difference in morphology between any two randomly chosen beats from these patients.

The symbolic mismatch computation achieves this by weighting the difference between the centroids for every pair of symbols for the patients by the frequencies with which these symbols occur.

An important feature of symbolic mismatch is that it is explicitly designed to avoid the need to set up a correspondence between the symbols of patients p and q. In contrast to cluster matching techniques (Chang et al., 1997; Cohen and Richman, 2002) to compare data for two patients by first making an assignment from symbols in one patient to the other, symbolic mismatch does not require any cross-patient registration of symbols. Instead, it performs weighted morphologic comparisons between all symbol centroids for patients p and q. As a result, the symbolization process does not need to be restricted to well-defined labels and is able to use a richer set of patient-specific symbols that capture fine-grained activity over long periods of time.

3.3 Spectrum Clipping

The formulation for symbolic mismatch in Equation 1 gives rise to a symmetric dissimilarity matrix. For methods that are unable to work directly from dissimilarities, this can be transformed into a similarity matrix using a generalized radial basis function (Vanschoenwinkel and Manderick, 2005). For both the dissimilarity and similarity case, however, symbolic mismatch can produce a matrix that is indefinite. This can be problematic when using symbolic mismatch with kernel-based algorithms because the optimization problems become non-convex and the underlying theory is invalidated (Chen et al., 2009b). In particular, kernel-based classification methods require Mercer's condition (Scholkopf and Smola, 2002) to be satisfied by a positive semi-definite kernel matrix. This creates the need to transform the symbolic mismatch matrix before it can be used as a kernel in these methods.

We use spectrum clipping to generalize the use of symbolic mismatch for classification. This approach has been shown both theoretically and empirically to offer advantages over other strategies (e.g., spectrum flipping, spectrum shifting, spectrum squaring, and the use of indefinite kernels) (Chen et al., 2009a). The symmetric mismatch matrix Ψ has an eigenvalue decomposition:

$$\Psi = U^T \Lambda U$$

where U is an orthogonal matrix and Λ is a diagonal matrix of real eigenvalues:

$$\Lambda = diag(\lambda_1, ..., \lambda_n).$$

Spectrum clipping makes Ψ positive semi-definite by clipping all negative eigenvalues to zero. The modified positive semi-definite symbolic mismatch matrix is then given by:

$$\Psi_{clip} = U^T \Lambda_{clip} U$$

where:

$$\Lambda_{clip} = diag(max(\lambda_1, 0), ..., max(\lambda_n, 0)).$$

Using Ψ_{clip} as a kernel matrix is then equivalent to using $(\Lambda_{clip})^{1/2} u_i$ as the representation of the i-th training sample.

We note that while we introduce spectrum clipping mainly for the purpose of broadening the applicability of symbolic mismatch to kernel-based methods, this approach offers additional advantages. Some researchers, for example, assume that negative eigenvalues of the similarity matrix are

caused by noise and view spectrum clipping as a denoising step (Wu et al., 2005). The results of our experiments, presented later in this paper, support the view of spectrum clipping being useful in a broader context.

4. Risk Stratification Using Symbolic Mismatch

We now present three different approaches using symbolic mismatch to identify high risk patients in a population. We choose these algorithms consistent with Eskin et al. (2002), as methods that can operate on high dimensional data, and that each detect points lying in sparse regions of the feature space in different ways. The first of these approaches uses a one-class SVM and a symbolic mismatch similarity matrix obtained using a generalized radial basis transformation. The other two approaches, nearest neighbor analysis and hierarchical clustering, use the symbolic mismatch dissimilarity matrix. In each case, the symbolic mismatch matrix is processed using spectrum clipping. All three of our approaches focus on finding patients with long-term ECG time-series that are anomalies in the population.

4.1 Classification Approach

Our first approach is based on SVMs. SVMs have been applied to anomaly detection in the one-class setting. Scholkopf et al. (2001) propose a method of adapting the SVM methodology to the one-class classification problem. This is done by mapping the data into the feature space corresponding to the kernel and separating instances from the origin with the maximum margin. To separate data from the origin, the following quadratic program is solved:

$$\min_{w,\xi,p} \frac{1}{2}\|w\|^2 + \frac{1}{vn}\sum_i \xi_i - p$$

subject to:

$$(w \cdot \Phi(z_i)) \geq p - \xi_i \;\; i = 1,...,n \;\; \xi_i \geq 0$$

where v reflects the tradeoff between incorporating outliers and minimizing the size of the support region.

For a new instance, the label is determined by evaluating which side of the hyperplane the instance falls on in the feature space. The resulting predicted label in terms of the Lagrange multipliers α_i and the spectrum clipped symbolic mismatch similarity matrix Ψ_{clip} is then:

$$\hat{y}_j = sgn(\sum_i \alpha_i \Psi_{clip}(i,j) - p).$$

We apply this approach to train a one-class SVM on all patients. Patients who lie outside the enclosing boundary are then labeled as anomalies. The parameter v can be varied to control the size of this group.

4.2 Nearest Neighbor Approach

Our second approach is based on the concept of nearest neighbor analysis. The assumption underlying this approach is that normal data instances occur in dense neighborhoods, while anomalies occur far from their closest neighbors.

We use an approach similar to that in Eskin et al. (2002). In this case, the anomaly score of each patient's long-term time-series is computed as the sum of its distances from time-series for its k-nearest neighbors, as measured by symbolic mismatch. Patients with anomaly scores exceeding a threshold θ are labeled as anomalies.

4.3 Clustering Approach

Our third approach is based on agglomerative hierarchical clustering. We begin by putting each patient in a separate cluster, and then proceed in each iteration to merge the two clusters that are most similar to each other. The distance between two clusters is defined as the average of the pairwise symbolic mismatch of the patients in each cluster. The clustering process terminates when it enters a region of diminishing returns (i.e., at a 'knee' of the curve corresponding to the distance of clusters merged together at each iteration). At this point, all patients outside the largest cluster are labeled as anomalies.

5. Evaluation Methodology

We evaluated our work on patients enrolled in the DISPERSE2 trial (Cannon et al., 2007). Patients in the study were admitted to a hospital with non-ST-elevation acute coronary syndrome (non-ST-elevation myocardial infarction or unstable angina). Three lead continuous ECG monitoring (LifeCard CF / Pathfinder, DelMar Reynolds / Spacelabs, Issaqua WA) was performed for a median duration of 4 days at a sampling rate of 128 Hz. The endpoints of cardiovascular death, myocardial infarction and severe recurrent ischemia were adjudicated by a blinded panel of clinical experts for a median follow-up period of 60 days. The maximum follow-up was 90 days. Data from 686 patients was available after removal of noise-corrupted signals. During the follow-up period there were 14 cardiovascular deaths, 28 myocardial infarctions, and 13 cases of severe recurrent ischemia. We define a major adverse cardiac event to be any of these three adverse events. The clinical characteristics of this patient population are presented in Table 1.

We studied the ability of each approach (i.e., classification, nearest neighbor analysis and clustering) to identify a high risk group of patients. Consistent with earlier studies to evaluate new methods for risk stratification in the setting of acute coronary syndrome (Shlipak et al., 2008), we classified patients in the highest quartile as the high risk group. For the classification approach, this corresponded to choosing v such that the group of patients lying outside the enclosing boundary constituted roughly 25% of the population. For the nearest neighbor approach we investigated all odd values of k from 3 to 9, and patients with anomaly scores in the top 25% of the population were classified as being at high risk. For the clustering approach, the varying sizes of the clusters merged together at each step made it difficult to select a high risk quartile. Instead, patients lying outside the largest cluster were categorized as being at risk. In the tests reported later in this paper, this group contained roughly 23% the patients in the population. We used the LIBSVM implementation for our one-class SVM. Both the nearest neighbor and clustering approaches were carried out using MATLAB implementations.

We employed Kaplan-Meier survival analysis (Efron, 1988) to compare the rates for major adverse cardiac events between patients declared to be at high and low risk for all three approaches. Hazard ratios (HR) and 95% confidence interval (CI) were estimated using a Cox proportional hazards regression model (Cox, 1972). The predictions of each approach were studied in univariate models, and also in multivariate models that additionally included other clinical risk variables

Age (years)	62 (53 to 70)
Age≥65 years	41%
Female Gender	36%
Current Smoker	57%
Hypertension	69%
Diabetes Mellitus	23%
Hyperlipidemia	64%
History of COPD	9%
History of CHD	37%
Previous MI	25%
Previous angina	58%
ST depression>0.5mm	66%
Index diagnosis of MI	49%

Table 1: Clinical characteristics of patient population used for study.

(age≥65 years, gender, smoking history, hypertension, diabetes mellitus, hyperlipidemia, history of chronic obstructive pulmonary disorder (COPD), history of coronary heart disease (CHD), previous MI, previous angina, ST depression on admission, index diagnosis of MI) as well as ECG risk metrics proposed in the past such as heart rate variability (HRV) (Malik et al., 1996), heart rate turbulence (HRT) (Barthel et al., 2003), and deceleration capacity (DC) (Bauer et al., 2006).

For HRV, we used metrics proposed by the Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology: the standard deviation of normal-to-normal intervals (SDNN), standard deviation of sequential five minute normal-to-normal means (SDANN), mean of the standard deviation of sequential five minute normal-to-normal intervals (ASDNN), root mean square successive differences (rMSSD), heart rate variability index (HRVI), percent of normal-to-normal interval increments greater than 50 ms (pNN50) and the ratio of low frequency power to the high frequency power (LF/HF). While we computed all HRV measures, we only report results for the best performing one, that is, LF/HF. HRV-LF/HF was dichotomized at 0.95 using the results reported earlier in the literature (Malik et al., 1996).

We measured HRT and DC for each patient using the libRASCH software shared for research use by the inventors of the method (Technische Universitat Munchen, Munich, Germany). HRT was trichotomized based on the turbulence onset (TO) and turbulence slope (TS) as follows: HRT0 (TO<0 and TS>2.5ms), HRT1 (either TO>0 or TS<2.5ms), and HRT2 (TO>0 and TS<2.5ms) (Barthel et al., 2003). DC was trichotomized as follows: category 0 (>4.5ms), category 1 (2.5 ms-4.5 ms), and category 2 (<2.5ms) (Bauer et al., 2006). Both HRT and DC were treated as continuous variables in our study.

We did not compare our work to T-wave alternans (TWA) (Rosenbaum et al., 1994) as TWA is typically measured using specialized hardware and maneuvers to increase the heart rate. While we experimented with a TWA algorithm that has recently been proposed to measure TWA on ECG data collected routinely during admission (Nearing and Verrier, 2002), this algorithm did not produce good results. We believe these results reflect an inability to measure TWA without specialized hardware and manoeuvres, as opposed to the lack of predictive discrimination for the method. We therefore excluded TWA from our analysis, as an ECG approach that is more appropriate for ECG signals collected under specialized conditions.

Method	HR	P Value	95% CI
One-Class SVM	1.38	0.033	1.04-1.89
3-Nearest Neighbor	1.91	0.031	1.06-3.44
5-Nearest Neighbor	2.10	0.013	1.17-3.76
7-Nearest Neighbor	2.28	0.005	1.28-4.07
9-Nearest Neighbor	2.07	0.015	1.15-3.71
Hierarchical Clustering	2.04	0.017	1.13-3.68

Table 2: Univariate association of high risk predictions from different approaches using symbolic mismatch with major adverse cardiac events over a 90 day period following acute coronary syndrome.

6. Results

We divide our results into two broad groups: univariate results (symbolic mismatch-based approaches in univariate models), and multivariate results (symbolic mismatch-based approaches in multivariate models). We also report on the effect of spectrum clipping on performance and provide some brief results regarding the runtime performance of our approach.

6.1 Univariate Results

Results of univariate analysis for all three unsupervised symbolic mismatch-based approaches are presented in Table 2. The predictions from all methods showed a statistically significant (i.e., $p < 0.05$) association with major adverse cardiac events following acute coronary syndrome. The results in Table 2 can be interpreted as roughly a doubled rate of adverse outcomes per unit time in patients identified as being at high risk by the nearest neighbor and clustering approaches. For the classification approach, patients identified as being at high risk had a nearly 40% increased risk of adverse outcomes. Kaplan-Meier survival curves for all three methods are shown in Figures 3 to 5. For the nearest neighbor approach, we present only the results for the best performing k (i.e., $k = 7$).

For comparison, we also include the univariate association of the other clinical and ECG risk variables in our study (Tables 3 and 4). Both the nearest neighbor and clustering approaches had a higher hazard ratio in this patient population than any of the clinical and ECG risk variables studied. Of the clinical risk variables, only age was found to be associated on univariate analysis with major cardiac events after acute coronary syndrome. Diabetes (p=0.072) was marginally outside the 5% level of significance. Of the ECG risk variables, both HRT and DC showed a univariate association with major adverse cardiac events during follow-up.

These results suggest that unsupervised risk stratification using symbolic mismatch can successfully identify patients at an elevated risk of major adverse cardiac events following ACS. In particular, our data shows that patients categorized as high risk by our methods continue to experience an increased risk of adverse outcomes throughout the entire 90 day period post-ACS (Figures 3-5). Our findings are also encouraging in that the relative increase in patient risk found using our methods compares quite favorably with other metrics based on specialized knowledge that are used in existing cardiac risk scoring tools. While the size of our patient population leads us to avoid statements about the nearest neighbor and clustering approaches being better than the other variables in our study (i.e., on the basis of having a higher observed hazard ratio than these other variables), we

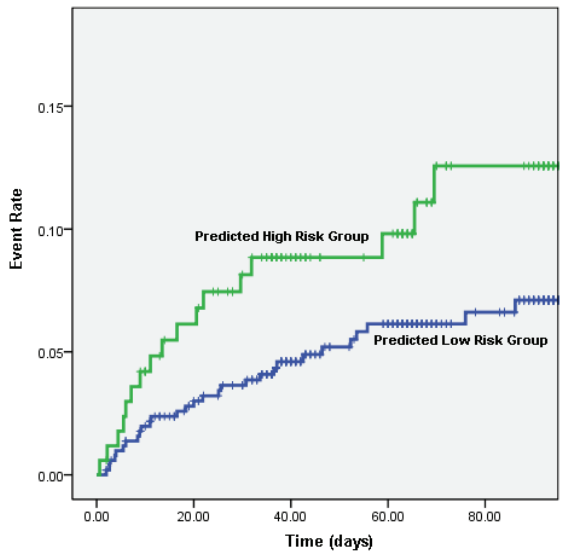


Figure 3: Kaplan-Meier major cardiac event curve for the one-class SVM approach. Ticks represent patient censoring (i.e., patients leaving the study before completion). The top (green) line corresponds to patients with anomaly scores in the top quartile of the population.

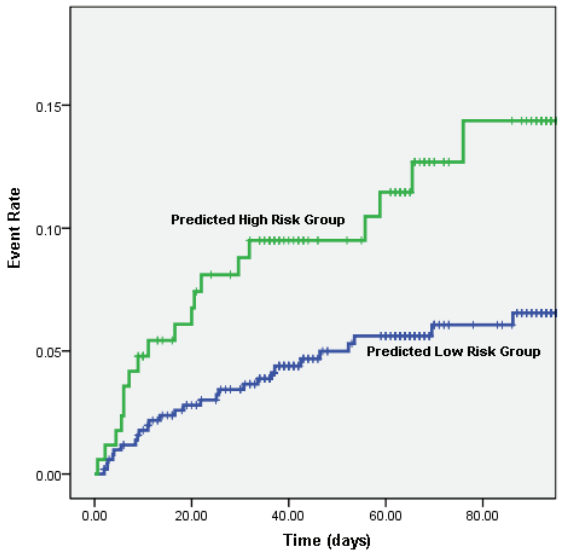


Figure 4: Kaplan-Meier major cardiac event curve for the 7-nearest neighbor approach. Ticks represent patient censoring (patients leaving the study before completion). The top (green) line corresponds to patients with anomaly scores in the top quartile of the population.

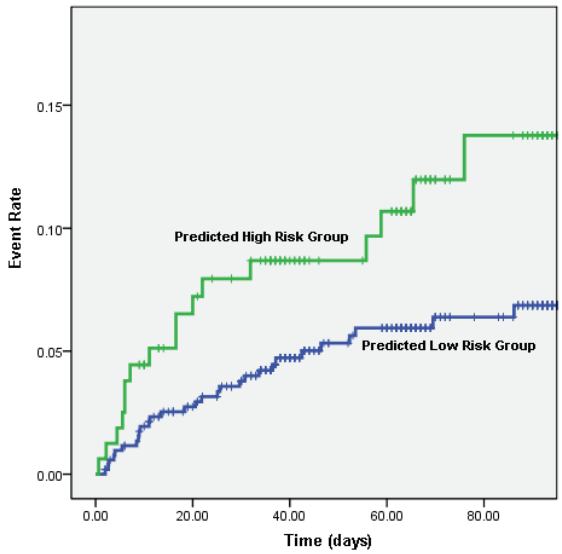


Figure 5: Kaplan-Meier major cardiac event curve for the hierarchical clustering approach. Ticks represent patient censoring (patients leaving the study before completion). The top (green) line corresponds to patients with anomaly scores in roughly the top quartile of the population.

Clinical Variable	HR	P Value	95% CI
Age≥65 years	1.82	0.041	1.02-3.24
Female Gender	0.69	0.261	0.37-1.31
Current Smoker	1.05	0.866	0.59-1.87
Hypertension	1.44	0.257	0.77-2.68
Diabetes Mellitus	1.95	0.072	0.94-4.04
Hyperlipidemia	1.00	0.994	0.55-1.82
History of COPD	1.05	0.933	0.37-2.92
History of CHD	1.10	0.994	0.37-2.92
Previous MI	1.17	0.630	0.62-2.22
Previous angina	0.94	0.842	0.53-1.68
ST depression>0.5mm	1.13	0.675	0.64-2.01
Index diagnosis of MI	1.42	0.134	0.90-2.26

Table 3: Univariate association of existing clinical risk variables with major adverse cardiac events over a 90 day period following acute coronary syndrome.

believe that our data provides strong support for the ability of unsupervised risk stratification to add information beyond these existing metrics.

ECG Variable	HR	P Value	95% CI
HRV	1.56	0.128	0.88-2.77
HRT	1.64	0.013	1.11-2.42
DC	1.77	0.002	1.23-2.54

Table 4: Univariate association of existing ECG risk variables with major adverse cardiac events over a 90 day period following acute coronary syndrome.

	Age	Fem	Smo	Hpt	Dia	Lip	COPD	CHD	MI	Ang	ST	Ind
One-Class SVM	-0.07	0.02	-0.03	-0.08	-0.06	0.03	-0.03	0.03	-0.07	0.01	0.06	-0.02
3-Nearest Neighbor	0.11	0.00	-0.02	0.05	0.03	-0.05	0.04	-0.09	0.08	0.04	0.01	0.02
5-Nearest Neighbor	0.12	0.01	-0.03	0.05	0.05	-0.04	0.04	-0.10	0.09	0.05	0.02	0.02
7-Nearest Neighbor	0.11	0.00	-0.03	0.05	0.06	-0.04	0.04	-0.10	0.09	0.06	0.02	0.02
9-Nearest Neighbor	0.11	0.00	-0.02	0.05	0.06	-0.04	0.05	-0.10	0.09	0.07	0.01	0.02
Hierarchical Clustering	0.16	0.03	-0.04	0.05	0.08	-0.05	0.05	-0.08	0.04	0.00	0.03	0.04

Table 5: Correlation of high risk predictions with clinical risk variables (Age=age$\geq$65, Fem=female gender, Smo=current smoker, Hpt=hypertension, Dia=diabetes mellitus, Lip=hyperlipidemia, COPD=history of COPD, CHD=history of CHD, MI=previous MI, Ang=previous angina, ST=ST depression$>$0.5mm, Ind=Index diagnosis of MI).

	HRV	HRT	DC
One-Class SVM	-0.14	-0.01	-0.09
3-Nearest Neighbor	0.16	0.00	0.02
5-Nearest Neighbor	0.16	0.01	0.03
7-Nearest Neighbor	0.15	0.01	0.03
9-Nearest Neighbor	0.17	0.01	0.04
Hierarchical Clustering	0.20	0.06	0.08

Table 6: Correlation of high risk predictions with ECG risk variables.

6.2 Multivariate Results

We measured the correlation between the predictions of the unsupervised symbolic mismatch-based approaches and both the clinical and ECG risk variables. These results are shown in Tables 5 and 6. All of the unsupervised symbolic mismatch-based approaches had low correlation with both the clinical and ECG variables ($R \leq 0.2$).

Our results on multivariate analysis reflect this low correlation between the symbolic mismatch-based approaches and existing clinical and ECG risk variables. On multivariate analysis, both the nearest neighbor approach and the clustering approach were independent predictors of adverse outcomes (Table 7). In our study, the nearest neighbor approach (for $k > 3$) had the highest hazard ratio on both univariate and multivariate analysis. Both the nearest neighbor and clustering approaches predicted patients with an approximately two-fold increased risk of adverse outcomes. This increased risk did not change much even after adjusting for other clinical and ECG risk variables. While the classification approach did not perform as well, we note that this result may potentially be improved with availability of a larger data set to learn an enclosing boundary and by only using data from patients known to remain event-free on follow-up.

Method	Adjusted HR	P Value	95% CI
One-Class SVM	1.32	0.074	0.97-1.79
3-Nearest Neighbor	1.88	0.042	1.02-3.46
5-Nearest Neighbor	2.07	0.018	1.13-3.79
7-Nearest Neighbor	2.25	0.008	1.23-4.11
9-Nearest Neighbor	2.04	0.021	1.11-3.73
Hierarchical Clustering	1.86	0.042	1.02-3.46

Table 7: Multivariate association of high risk predictions from different approaches using symbolic mismatch with major adverse cardiac events over a 90 day period following acute coronary syndrome. Multivariate results adjusted for age$\geq$65 years, gender, smoking history, hypertension, diabetes mellitus, hyperlipidemia, history of COPD, history of CHD, previous MI, previous angina, ST depression on admission, index diagnosis of MI, HRV-LF/HF, HRT and DC.

Method	HR	P Value	95% CI
One-Class SVM	1.36	0.038	1.01-1.79
3-Nearest Neighbor	1.74	0.069	0.96-3.16
5-Nearest Neighbor	1.57	0.142	0.86-2.88
7-Nearest Neighbor	1.73	0.071	0.95-3.14
9-Nearest Neighbor	1.89	0.034	1.05-3.41
Hierarchical Clustering	1.19	0.563	0.67-2.12

Table 8: Univariate association of high risk predictions without the use of spectrum clipping. None of the approaches showed a statistically significant association with the study endpoint in any of the multivariate models including other clinical risk variables when spectrum clipping was not used.

Method	AUROC (Model 1)	AUROC (Model 2)
One-Class SVM	0.683	0.705
3-Nearest Neighbor	0.683	0.713
5-Nearest Neighbor	0.683	0.721
7-Nearest Neighbor	0.683	0.725
9-Nearest Neighbor	0.683	0.719
Hierarchical Clustering	0.683	0.711

Table 9: Improvement in discrimination when information obtained through unsupervised risk stratification is added to multivariate models containing age$\geq$65 years, gender, smoking history, hypertension, diabetes mellitus, hyperlipidemia, history of COPD, history of CHD, previous MI, previous angina, ST depression on admission, index diagnosis of MI, HRV-LF/HF, HRT and DC (Model A: existing risk variables, Model B: existing risk variables combined with unsupervised risk stratification).

Consistent with the univariate case above, we consider these findings to be encouraging. Our data suggests that the information available through unsupervised risk stratification based on sym-

bolic mismatch is generally independent of the information available through other specialized metrics. Moreover, our approach can potentially be used in a synergistic manner with these other metrics to improve risk stratification. For example, our study found that nearest neighbor-based risk stratification using symbolic mismatch can identify individuals who are at a two- to three-fold increased risk of adverse outcomes, even after adjusting for an extensive set of existing risk variables. This provides strong support for the potential ability of our research to complement present approaches to prognosticate cardiac patients. We hypothesize that these results are largely due to our focus on capturing information that is quite distinct from existing metrics. In particular, both our approach of risk stratifying patients within an unsupervised anomaly detection framework, and our focus on exploiting large volumes of long-term ECG data that is not well-suited for human analysis, predispose to capturing information that is clinically useful but not reflected in current metrics.

To quantify this effect better, we also studied how the area under the receiver operating characteristic curve (AUROC) changed for multivariate models constructed with and without the use of information obtained through unsupervised risk stratification. Table 9 presents the results obtained for this experiment. For each of the unsupervised risk stratification approaches, the addition of the information produced by these methods increased the ability of models developed using existing risk variables to discriminate between high and low risk patients. Consistent with the earlier results, this improvement was greatest for the 7-nearest neighbor approach. The results here provide further support for the information provided by our methods being potentially complementary to that captured by current risk variables.

6.3 Effect of Spectrum Clipping

We also investigated the effect of spectrum clipping on the performance of our different risk stratification approaches. Table 8 presents the associations when spectrum clipping was not used. For all three methods, performance was improved when spectrum clipping was used. We note that while our motivation for using spectrum clipping was largely to broaden the applicability of symbolic mismatch to kernel-based methods, the ability of spectrum clipping to reduce noise provided a positive effect for all methods.

6.4 Runtime Performance

Figure 6 presents a histogram of the number of heart beats in each patient's long-term ECG signal over the first 24 hours following admission. The median number of beats per patient was 99,581, with an interquartile range of 89,051 to 110,337. The minimum number of beats was 45,330 while the maximum was 161,696.

Figure 7 presents similar information for the number of symbols obtained per patient through the clustering process described in Section 3.1. The median number of symbols per patient was 66, with an interquartile range of 37 to 114. The minimum number of symbols was 1 while the maximum was 284.

On a dual-core Intel Pentium 4 3.06 GHz platform with 4GB RAM running MATLAB R2007a with Ubuntu 9.10 the symbolization of each patient's data (24 hours of ECG sampled at 128 Hz) took around 10 minutes. Roughly speaking, the use of symbolization compressed around 100,000 beats per patient to below 70 symbols (i.e., a reduction by a factor of just under 1,500). The corresponding improvement in the runtime of comparing long-term ECG signals was quadratic in this reduction, since (roughly speaking) instead of 100,000-by-100,000 comparisons of heart beats in the original

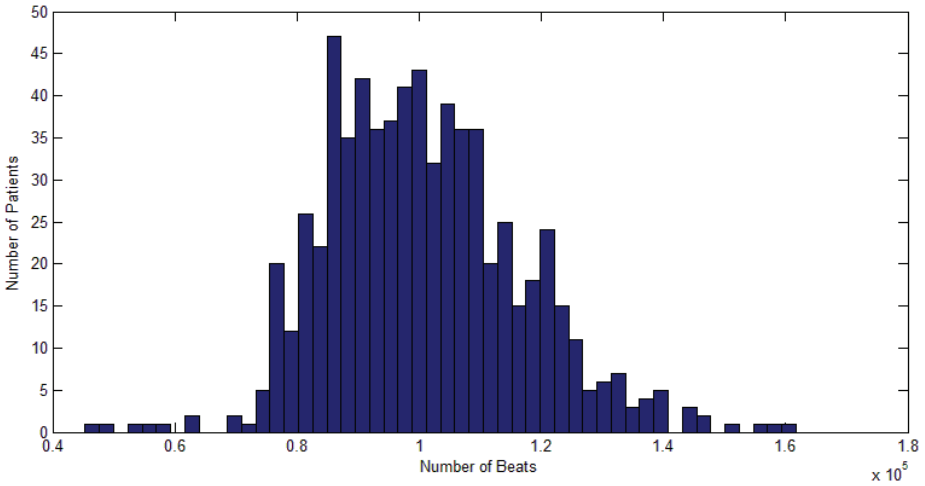


Figure 6: Histogram of the heart beats per patient over 24 hours (x-axis scale: $\times 10^5$).

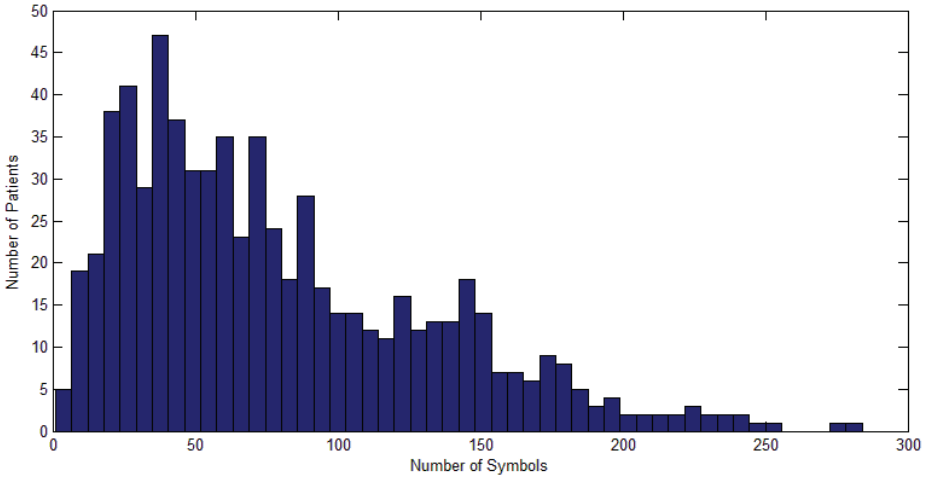


Figure 7: Histogram of the number of symbols per patient.

signals, only 70-by-70 symbol centroid and symbol frequency comparisons were necessary. The overall runtime complexity of our analysis was therefore $O(n^2\theta^2l^2) + O(n\theta ml^2)$, where the left term corresponds to the runtime of finding anomalies using symbolized data and the right term corresponds to the runtime of creating symbolic representations of the original ECG signals. We denote the number of patients by n, the maximum number of symbols for any patient by θ, the maximum number of heart beats for any patient by m, and the maximum length of any heart beat (in samples) by l. The left term of the runtime above is quadratic in the number of patients (since all pairs of patients are compared to find anomalies), the number of symbols (since all pairs of symbols are compared for any pair of patients), and the length of symbol centroids (since DTW compares all the samples for each pair of symbol centroids). The right term of the runtime above is linear in the number of patients (since each patient's data is symbolized once), with θm corresponding to the

time taken to make a pass through all m beats in each of the θ iterations of Max-Min clustering, and l^2 being the cost of using DTW to compare heart beats. While the computational cost of clustering is significant, it leads to a θ^2 factor in the runtime associated with finding anomalies using symbolic mismatch rather than an m^2 factor (where m is much larger than θ as shown by our results). The use of symbolization therefore represents one of the key sources of speedup in our study, reducing the runtime from $O(n^2m^2l^2)$ without symbolization to $O(n^2\theta^2l^2) + O(n\theta ml^2)$ with symbolization. We note that while other sources of runtime improvement are also possible (e.g., by addressing the quadratic runtime of DTW or by avoiding comparisons between all pairs of patients in the population for anomaly detection), the values of n and l are both much smaller, and therefore represent smaller gains, than the m^2 factor reduced by symbolization.

7. Related Work

Previous work on comparing time-series can be divided into two broad classes: methods to compare signals based on their raw samples, and methods to compare signals by extracting features from the data.

Most previous work on comparing signals in terms of their raw samples, including metrics such as dynamic time warping (Keogh and Pazzani, 2001; Keogh and Ratanamahatana, 2005), longest common subsequence (Vlachos et al., 2002), edit distance with real penalty (Cheng and Ng, 2004), sequence weighted alignment (Morse and Patel, 2007), spatial assembling distance (Chen et al., 2007), focuses on relatively short time-series. This is due to the runtime of these methods (quadratic for many methods) and the need to reason in terms of the frequency and dynamics of higher-level signal constructs (as opposed to individual samples) when studying systems over long periods. These existing methods do not, therefore, directly focus on comparing very long signals (e.g., tens of millions of samples).

In contrast to this, the vast majority of prior research on comparing long-term time-series focuses on extracting specific features from long-term signals and quantifying the differences between these features. For example, in the context of cardiovascular disease, long-term ECG is often reduced to features (e.g., mean heart rate or heart rate variability) and compared in terms of these features. These approaches, unlike our symbolic mismatch based approaches, draw upon significant *a priori* knowledge. Our belief was that for applications like risk stratifying patients for major cardiac events, focusing on a set of specialized features leads to important information being potentially missed. In our work, we focus instead on developing an approach that avoids use of significant *a priori* knowledge by comparing the raw morphology of long-term time-series. We propose doing this in a computationally efficient and systematic way through symbolization. While this use of symbolization represents a lossy compression of the original signal, the underlying process of quantifying differences between long-term time-series remains grounded in the comparison of raw morphology.

The process of symbolization maps the problem of comparing long-term time-series into the domain of sequence comparison. There is an extensive body of prior work focusing on the comparison of sequential or string data. Algorithms based on measuring the edit distance between strings are widely used in disciplines such as computational biology (Jones and Pevzner, 2004; Durbin et al., 1998), but are typically restricted to comparisons of short sequences because of their computational complexity. More closely related to our research is previous work on the use of profile hidden Markov models (Krogh, 1994; Jaakkola et al., 1999) to optimize recognition of binary labeled se-

quences. This work focuses on learning the parameters of a hidden Markov model that can represent approximations of sequences and can be used to score other sequences. Generally, this approach requires large amounts of data or sophisticated priors to train the hidden Markov models. Computing forward and backward probabilities from the Baum-Welch algorithm is also very computationally intensive. Subsequent research in this area focuses on mismatch tree-based kernels (Leslie et al., 2003), which use the mismatch tree data structure (Eskin and Pevzner, 2002) to quantify the difference between two sequences based on the approximate occurrence of fixed length subsequences within them. Similar to this approach is work on using a "bag of motifs" representation (Ben-Hur and Brutlag, 2006), which provides a more flexible representation than fixed length subsequences but usually requires prior knowledge of motifs in the data (Ben-Hur and Brutlag, 2006).

In contrast to these efforts, we use a simple, computationally efficient approach to compare symbolic sequences without prior knowledge. Most importantly, our approach helps address the scenario where symbolizing long-term time-series in a patient-specific manner leads to the symbolic sequences for patients being drawn from different alphabets (Syed et al., 2010). In this case, hidden Markov models, mismatch trees or a "bag of motifs" approach trained on one patient cannot be easily used to score the sequences for other patients. Instead, any comparative approach must maintain a hard or soft registration of symbols across individuals. Symbolic mismatch addresses this scenario and complements existing work on sequence comparison through a simple, computationally efficient measure that quantifies differences across patients while retaining information on how the symbols for these patients differ.

Finally, we also distinguish our work from earlier efforts to risk stratify patients using long-term data. In particular, we supplement our symbolic mismatch kernel with the idea of detecting high risk patients as those individuals in the population with unusual long-term time-series. For example, in the context of cardiovascular disease, techniques such as heart rate variability (Malik et al., 1996), heart rate turbulence (Barthel et al., 2003), and t-wave alternans (Smith et al., 1988; Rosenbaum et al., 1994) have all been shown to be useful in risk stratifying patients at risk for future cardiovascular events following acute coronary syndromes. The focus of these methods is to calculate a particular pre-defined feature from the raw ECG signal, and to use it to rank patients along a risk continuum. Our approach, focusing on detecting patients with high symbolic mismatch relative to other patients in the population, is orthogonal to the use of specialized high risk features along two important dimensions. First, it does not require the presence of significant prior knowledge. For the cardiovascular care, we only assume that ECG signals from patients who are at high risk differ from those of the rest of the population. There are no specific assumptions about the nature of these differences. Second, the ability to partition patients into groups with similar ECG characteristics and potentially common risk profiles potentially allows for a more fine-grained understanding of a how a patient's future health may evolve over time. Matching patients to past cases with similar ECG signals could lead to more accurate assignments of risk scores for particular events such as death and recurring heart attacks.

8. Discussion

In this paper, we proposed using symbolic mismatch to quantify differences in long-term physiological time-series. Our approach uses a symbolic transformation to measure changes in the morphology and frequency of prototypical functional units observed over long periods in two signals.

In addition to proposing symbolic mismatch, which avoids feature extraction and deals with inter-patient differences in a parameter-less way, we also explored the hypothesis that high risk patients in a population can be identified as individuals with anomalous long-term signals. We developed multiple comparative approaches to detect such patients, and evaluated these methods in a real-world application of risk stratification for major adverse cardiac events following acute coronary syndrome. Our results suggest that symbolic mismatch-based comparative approaches may have clinical utility in identifying high risk patients, and can provide information that is complementary to existing clinical risk variables.

In particular, we note that the hazard ratios we report are typically considered clinically meaningful. Risk stratification following ACS is an extremely challenging goal. In a different study of 118 variables in 15,000 post-ACS patients with 90 day follow-up similar to our population, Newby et al. (2003) did not find any variables with a hazard ratio greater than 2.00. We observed a similar result in our patient population, where all of the existing clinical and ECG risk variables had a hazard ratio less than 2.00. In contrast to this, our nearest neighbor-based approach achieved a hazard ratio of 2.28, even after being adjusted for existing risk measures. We believe that these results provide strong support for the potential role of our research in improving the management of patients post-ACS.

We envision our techniques being primarily useful in their ability to enrich models for cardiovascular risk stratification. In other words, we expect the risk scores generated by our methods to serve as features that can be combined with other features based on specialized knowledge while assessing the overall health of patients. While we dichotomized the results of all of our methods for evaluation consistent with the way most new cardiovascular risk metrics are validated, we believe that the best use of this information is in its original continuous form to provide a finer grained distinction between high and low risk patients. We further believe that the eventual use case of our tools will be to assess individual patients that present at different times as anomalies relative to a continuously increasing data set of patients seen previously. Aspects of our research, such as symbolic mismatch, may also have a role in a supervised setting, as we discuss later in this section.

In the context of cardiovascular disease, techniques such as heart rate variability, heart rate turbulence, T wave alternans, and morphologic variability have all been shown to be useful in risk stratifying patients at risk for future cardiovascular events following acute coronary syndromes. The focus of these methods is to calculate a specific pre-defined feature from the raw ECG signal, and to use it to rank patients along a risk continuum. Our approach, focusing on detecting patients with high symbolic mismatch relative to other patients in the population, is orthogonal (and perhaps complementary) to the use of specialized high risk features. First, it does not require the presence of significant prior knowledge. For the cardiovascular care, we only assume that ECG signals from patients who are at high risk differ from those of the rest of the population. There are no specific assumptions about the nature of these differences. Second, the ability to partition patients into groups with similar ECG characteristics and potentially common risk profiles potentially allows for a more fine-grained understanding of a how a patient's future health may evolve over time. Matching patients to past cases with similar ECG signals could lead to more accurate assignments of risk scores for particular events such as death and recurring heart attacks.

We conclude with some limitations of our work. While our decision to compare the morphology and frequency of prototypical functional units leads to a measure that is computationally efficient on large volumes of data, this process does not capture information related to the dynamics of these prototypical units or in specific sequences of symbols. We also observe that all three of the com-

parative approaches investigated in our study focus only on identifying patients who are anomalies. Although we believe that symbolic mismatch may have further use in supervised learning, the size of our patient population and the small number of adverse cardiac outcomes over the 90 day follow-up meant that dividing the patients into separate training and testing groups would make it challenging to learn models that generalized well. This hypothesis, that is, of symbolic mismatch being useful in the setting of supervised learning, therefore needs to be evaluated more fully on larger patient populations. Finally, we note that we did not have echocardiographic data for patients in the DISPERSE2 trial. As a result, we did not include a comparison in our study to metrics such as left ventricular ejection fraction (LVEF). We believe that our research warrants further investigation on larger data sets, with a more comprehensive set of existing clinical metrics, and longer follow-ups in the future.

Acknowledgments

We thank GE Healthcare and Georg Schmidt for sharing the libRASCH toolbox to measure HRT and DC values. This research was supported by the National Science Foundation CAREER award 1054419.

References

AAMI. Recommended practice for testing and reporting performance results of ventricular arrhythmia detection algorithms. 1987.

AAMI. Testing and reporting performance results of cardiac rhythm and ST segment measurement algorithms. 1998.

J.S. Alpert. Risk stratification for cardiac events after acute ST elevation myocardial infarction. *UpToDate*, 2010.

J.J. Bailey, A.S. Berson, H. Handelsman, and M. Hodges. Utility of current risk stratification tests for predicting major arrhythmic events after myocardial infarction. *Journal of the American College of Cardiology*, 38(7):1902–1911, 2001.

P. Barthel, R. Schneider, A. Bauer, K. Ulm, C. Schmitt, A. Schomig, and G. Schmidt. Risk stratification after acute myocardial infarction by heart rate turbulence. *Circulation*, 108(10):1221–1226, 2003.

A. Bauer, J.W. Kantelhardt, P. Barthel, R. Schneider, T. Makikallio, K. Ulm, K. Hnatkova, A. Schomig, H. Huikuri, A. Bunde, et al. Deceleration capacity of heart rate as a predictor of mortality after myocardial infarction: cohort study. *The Lancet*, 367(9523):1674–1681, 2006.

A. Ben-Hur and D. Brutlag. Sequence motifs: highly predictive features of protein function. *Feature Extraction*, pages 625–645, 2006.

J.A Breall and M. Simons. Risk stratification after unstable angina or non-ST elevation myocardial infarction. *UpToDate*, 2010.

C.P. Cannon, S. Husted, R.A. Harrington, B.M. Scirica, H. Emanuelsson, G. Peters, and R.F. Storey. Safety, Tolerability, and Initial Efficacy of AZD6140, the First Reversible Oral Adenosine Diphosphate Receptor Antagonist, Compared With Clopidogrel, in Patients With Non–ST-Segment Elevation Acute Coronary Syndrome Primary Results of the DISPERSE-2 Trial. *Journal of the American College of Cardiology*, 50(19):1844–1851, 2007.

S.H. Chang, F.H. Cheng, W. Hsu, and G. Wu. Fast algorithm for point pattern matching: invariant to translations, rotations and scale changes. *Pattern Recognition*, 30(2):311–320, 1997.

Y. Chen, MA Nascimento, B.C. Ooi, and AKH Tung. Spade: On shape-based pattern detection in streaming time series. In *IEEE International Conference on Data Engineering*, pages 786–795, 2007.

Y. Chen, E.K. Garcia, M.R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *Journal of Machine Learning Research*, 10:747–776, 2009a.

Y. Chen, MR. Gupta, and B. Recht. Learning kernels from indefinite similarities. In *International Conference on Machine Learning*, pages 145–152, New York, NY, USA, 2009b. ACM. ISBN 978-1-60558-516-1.

L. Cheng and R.T. Ng. On The marriage of LP-norms and edit distance. In *Very Large Data Bases*, pages 792–803, 2004.

W.W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 475–480. ACM New York, NY, USA, 2002.

T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

D.R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.

P. de Chazal, M. O'Dwyer, and R.B. Reilly. Automatic classification of heartbeats using ECG morphology and heartbeat interval features. In *IEEE Transactions on Biomedical Engineering*, pages 1196–1206, 2004.

C.H. Ding, X. He, H. Zha, M. Gu, and H. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *IEEE International Conference on Data Mining*, pages 107–114, 2001.

R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

B. Efron. Logistic regression, survival analysis, and the Kaplan-Meier curve. *Journal of the American Statistical Association*, 83(402):414–425, 1988.

E. Eskin and P.A. Pevzner. Finding composite regulatory patterns in DNA sequences. *Bioinformatics*, 18:354–363, 2002.

E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection. *Applications of Data Mining in Computer Security*, pages 77–200, 2002.

B. Goldstein. Intensive care unit ECG monitoring. *Cardiac Electrophysiology Review*, 1(3):308–310, 1997. ISSN 1385-2264.

P.S. Hamilton and W.J. Tompkins. Quantitative investigation of QRS detection rules using the MIT/BIH arrhythmia database. *IEEE Transactions on Biomedical Engineering*, 33(12):1157–1165, 1986.

T. Jaakkola, M. Diekhans, and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. In *Proceedings of the International Conference on Intelligent Systems for Molecular Biology*, pages 149–158, 1999.

N.C. Jones and P. Pevzner. *An Introduction to Bioinformatics Algorithms*. the MIT Press, 2004.

E. Keogh and C.A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, 2005.

E.J. Keogh and M.J. Pazzani. Derivative dynamic time warping. In *SIAM international Conference on Data Mining*. Citeseer, 2001.

R.E. Kleiger, P.K. Stein, and J.T. Bigger Jr. Heart rate variability: measurement and clinical utility. *Annals of Noninvasive Electrocardiology*, 10(1):88–101, 2005. ISSN 1542-474X.

A. Krogh. Hidden Markov models for labeled sequences. In *International Conference on Pattern Recognition*, pages 140–144. IEEE Computer Society Press, 1994.

C. Leslie, E. Eskin, J. Weston, and W.S. Noble. Mismatch string kernels for SVM protein classification. *Advances in Neural Information Processing Systems*, pages 1441–1448, 2003.

L.S. Lilly. Pathophysiology of Heart Failure. 2007.

G. Lopera and A.B. Curtis. Risk stratification for sudden cardiac death: current approaches and predictive value. *Current Cardiology Reviews*, 5:56–64, 2009.

J. Mackay, G.A. Mensah, S. Mendis, and K. Greenlund. *The Atlas of Heart Disease and Stroke*. World Health Organization, 2004.

M. Malik, J.T. Bigger, A.J. Camm, R.E. Kleiger, A. Malliani, A.J. Moss, and P.J. Schwartz. Heart rate variability: Standards of measurement, physiological interpretation, and clinical use. *European Heart Journal*, 17(3):354–381, 1996.

M.D. Morse and J.M. Patel. An efficient and accurate method for evaluating time series similarity. In *ACM SIGMOD International Conference on Management of Data*, pages 569–580. ACM, 2007.

CS Myers and LR Rabiner. A comparative study of several dynamic time-warping algorithms for connected word recognition. *Bell Systems Technical Journal*, 60(7):1389–1409, 1981.

B.D. Nearing and R.L. Verrier. Modified moving average analysis of T-wave alternans to predict ventricular fibrillation with high accuracy. *Journal of Applied Physiology*, 92(2):541–549, 2002.

L.K. Newby, M.V. Bhapkar, H.D. White, E.J. Topol, F.C. Dougherty, R.A. Harrington, M.C. Smith, L.F. Asarch, R.M. Califf, et al. Predictors of 90-day outcome in patients stabilized after acute coronary syndromes. *European Heart Journal*, 24(2):172–181, 2003.

Physionet. MIT-BIH Arrhythmia Database. 2005.

D.S. Rosenbaum, L.E. Jackson, J.M. Smith, H. Garan, J.N. Ruskin, and R.J. Cohen. Electrical alternans and vulnerability to ventricular arrhythmias. *New England Journal of Medicine*, 330 (4):235–241, 1994.

B. Scholkopf and A.J. Smola. *Learning with Kernels*. MIT Press, 2002.

B. Scholkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

M.G. Shlipak, J.H. Ix, K. Bibbins-Domingo, F. Lin, and M.A. Whooley. Biomarkers to predict recurrent cardiovascular disease: the Heart and Soul Study. *The American Journal of Medicine*, 121(1):50–57, 2008.

J.M. Smith, E.A. Clancy, C.R. Valeri, J.N. Ruskin, and R.J. Cohen. Electrical alternans and cardiac electrical instability. *Circulation*, 77(1):110–121, 1988.

Z. Syed, J. Guttag, and C. Stultz. Clustering and symbolic analysis of cardiovascular signals: Discovery and visualization of medically relevant patterns in long-term data using limited prior knowledge. *EURASIP Journal on Advances in Signal Processing*, 2007:14, 2007.

Z. Syed, C. Stultz, M. Kellis, P. Indyk, and J. Guttag. Motif discovery in physiological datasets: a methodology for inferring predictive elements. *ACM Transactions on Knowledge Discovery from Data*, 4(1):1–23, 2010.

B. Vanschoenwinkel and B. Manderick. Appropriate kernel functions for support vector machine learning with sequences of symbolic data. *Lecture Notes in Computer Science*, 3635/2005:256–280, 2005.

M. Vlachos, D. Gunopoulos, and G. Kollios. Discovering similar multidimensional trajectories. In *IEEE International Conference on Data Engineering*, pages 673–684. Published by the IEEE Computer Society, 2002.

J.H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

World Health Organization. Cardiovascular diseases (CVDs) fact sheet, 2009.

G. Wu, E.Y. Chang, and Z. Zhang. An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines. Technical report, University of California, Santa Barbara, 2005.

W. Zong, G.B. Moody, and D. Jiang. A robust open-source algorithm to detect onset and duration of QRS complexes. *Computers in Cardiology*, pages 737–740, 2003.

Journal of Machine Learning Research 12 (2011) 1025-1068 Submitted 7/10; Revised 1/11; Published 3/11

Two Distributed-State Models For Generating High-Dimensional Time Series*

Graham W. Taylor GWTAYLOR@CS.NYU.EDU
Courant Institute of Mathematical Sciences
New York University
New York, NY 10003, USA

Geoffrey E. Hinton HINTON@CS.TORONTO.EDU
Department of Computer Science
University of Toronto
Toronto, ON M5S 3G1, Canada

Sam T. Roweis ROWEIS@CS.NYU.EDU
Courant Institute of Mathematical Sciences
New York University
New York, NY 10003, USA

Editor: Yoshua Bengio

Abstract

In this paper we develop a class of nonlinear generative models for high-dimensional time series. We first propose a model based on the restricted Boltzmann machine (RBM) that uses an undirected model with binary latent variables and real-valued "visible" variables. The latent and visible variables at each time step receive directed connections from the visible variables at the last few time-steps. This "conditional" RBM (CRBM) makes on-line inference efficient and allows us to use a simple approximate learning procedure. We demonstrate the power of our approach by synthesizing various sequences from a model trained on motion capture data and by performing on-line filling in of data lost during capture.

We extend the CRBM in a way that preserves its most important computational properties and introduces multiplicative three-way interactions that allow the effective interaction weight between two variables to be modulated by the dynamic state of a third variable. We introduce a factoring of the implied three-way weight tensor to permit a more compact parameterization. The resulting model can capture diverse styles of motion with a single set of parameters, and the three-way interactions greatly improve its ability to blend motion styles or to transition smoothly among them.

Videos and source code can be found at `http://www.cs.nyu.edu/~gwtaylor/publications/jmlr2011`.

Keywords: unsupervised learning, restricted Boltzmann machines, time series, generative models, motion capture

1. Introduction

The simplest time series models, and the earliest studied, contain no hidden variables. Two members of this class of "fully-observed" models are the vector autoregressive model and the N^{th} order

*. This article is dedicated to the memory of the third author who unexpectedly passed away on January 12, 2010.

©2011 Graham W. Taylor, Geoffrey E. Hinton and Sam T. Roweis.

Markov model. Though elegant in their construction, these models are limited by their lack of memory. To capture long-range structure they must maintain explicit links to observations in the distant past, which results in a blow-up in the number of parameters. The strong regularities present in many time series suggest that a more efficient parameterization is possible.

More powerful models, such as the popular hidden Markov model (HMM), introduce a hidden (or latent) state variable that controls the dependence of the current observation on the history of observations. HMMs, however, cannot efficiently model data that is a result of multiple underlying influences since they rely on a single, discrete K-state multinomial to represent the entire history of observations. To model N bits of information about the past, they require 2^N hidden states.

In this paper, we propose an alternative class of time series models that have three key properties which distinguish them from the prior art. The first property is distributed (i.e., componential) hidden state. Mixture models such as HMMs generate each observation from a single category. Distributed state models (e.g., products) generate each object from a set of features that each contain some aspect of that object's description. linear dynamical systems (LDS) have a continuous, and therefore componential hidden state, but in order to make inference in these models tractable, the relationship between latent and visible variables is constrained to be linear. We show that by carefully choosing the right form of nonlinear observation model it is possible to attain tractable, exact inference, yet retain a rich representational capacity that is linear in the number of components.

Directed acyclic graphical models (or Bayes nets) are a dominant paradigm in models of static data. Their temporal counterparts, dynamic Bayes nets (Ghahramani, 1998), generalize many existing models such as the HMM and its various extensions. In all but the simplest directed models, inference is made difficult due to a phenomenon known as "explaining away" where observing a child node renders its parents dependent (Pearl, 1988). To perform inference in these networks, typically one resorts to approximate techniques such as variational inference (Neal and Hinton, 1998) or Monte Carlo methods which have a significant number of disadvantages (Ghahramani, 1998; Murphy, 2002).

An alternative to directed models is to abandon the causal relationship between variables, and instead focus on *undirected* models. One such model, the restricted Boltzmann machine (RBM) (Smolensky, 1986), has garnered recent interest due to its desirable property of permitting efficient exact inference. Unfortunately this comes at a cost: Exact maximum likelihood learning is no longer possible due to the existence of an intractable normalizing constant called the partition function. However, the RBM has an efficient, approximate learning algorithm, contrastive divergence (CD) (Hinton, 2002), that has been shown to scale well to large problems. RBMs have been used in a variety of applications (Welling et al., 2005; Gehler et al., 2006; Hinton and Salakhutdinov, 2006; Larochelle et al., 2007; Salakhutdinov et al., 2007) and over the last few years their properties have become better understood (Bengio and Delalleau, 2008; Salakhutdinov and Murray, 2008; Sutskever and Hinton, 2008). The CD learning procedure has also been improved (Carreira-Perpinan and Hinton, 2005; Tieleman, 2008; Tieleman and Hinton, 2009). With a few exceptions (Hinton and Brown, 2000; Sutskever and Hinton, 2007) the literature on RBMs is confined to modeling static data. In this paper, we leverage the desirable properties of an undirected architecture, the RBM, and extend it to model time series. This brings us to the second key property of the models we propose: their observation or emission distribution is an undirected, bipartite graph. This makes inference in our models simple and efficient.

The final key property of our proposed models is that they can form the building blocks of deep networks by incrementally learning one layer of feature extractors at a time. One motivation for

promoting deep architectures is biological plausibility. Experimental evidence supports the belief that the brain uses multiple layers of feature-detecting neurons to process rich sensory input such as speech or visual signals (Hinton, 2007). There is also a practical argument for deep learning. In capturing more abstract, high-level structure from the data, the higher layers provide a more statistically salient representation for tasks such as discrimination. These ideas are not new, but until recently, the problem of how to efficiently train deep networks remained open. The backpropagation algorithm requires a large amount of labeled data and has difficulties with poor local minima and vanishing gradients. A resurgence in the study of deep architectures has been sparked by the discovery that deep networks can be initialized by greedy unsupervised learning of each layer (Hinton et al., 2006). RBMs were originally proposed for this task, but autoencoders (Bengio et al., 2007) and sparse encoder-decoder networks (Ranzato et al., 2006) have also been shown to work. After a pre-training stage, the entire network can be fine-tuned with either a generative or discriminative objective.

2. Modeling Human Motion

Motion capture (mocap) is the process of recording the movement of a subject as a time series of 3D cartesian coordinates corresponding to real or virtual points on the body. Most modern systems use a series of synchronized high-speed cameras to capture the location of strategically-placed physical markers attached to the subject (so called "marker-based" systems) or use image features to infer points of interest (so-called "markerless" systems). Marker-based systems are much more common but necessitate the use of a laboratory setting. Markerless systems permit motion capture in more natural environments (e.g., outdoors) but in general require more time to post-process the data. Recent advances in motion capture technology have fueled interest in the analysis and synthesis of motion data for computer animation and tracking.

Given its high-dimensional nature, nonlinearities, and long-range dependencies, mocap data is ideal for both studying the limitations of time series models and demonstrating their effectiveness. Several large motion capture data repositories are available, and people are very good at detecting anomalies in data that is generated from a model, so it is easy to judge the relative generative ability of two models. While focusing on a particular domain has greatly facilitated model development and comparison, there is nothing motion-specific to any of the models discussed herein. Therefore, there is no reason to believe that they cannot be applied to other high-dimensional, highly-structured time series data. In the following discussion, we briefly review related work in mocap-driven motion synthesis.

2.1 Motion Synthesis for Computer Animation

A dominant approach in computer animation is "keyframing" whereby an animator employs software to manually configure the "key" body poses over time, and these frames are interpolated to form smooth trajectories. This process, however, is time and labor intensive. It is therefore common to use mocap data to supplement or replace keyframing. A variety of methods have been developed to exploit the plethora of high-quality motion sequences available for animation. These approaches can be loosely divided into a handful of categories which we describe below.

2.1.1 CONCATENATION METHODS

Perhaps the simplest way to generate new motion sequences based on data is to sensibly concatenate short examples from a motion database to meet sparse user-specified constraints (Tanco and Hilton, 2000; Arikan and Forsyth, 2002; Kovar et al., 2002; Lee et al., 2002; Arikan et al., 2003). Pullen and Bregler (2002) propose a hybrid approach where low-frequency components are retained from user input and high-frequency components, called "texture", are added from the database. The obvious benefit of concatenation approaches is the high-quality motion that is produced. However, the "synthesized" motions are restricted to content already in the database and therefore many resources must be devoted to capture all desired content.

2.1.2 BLENDING AND INTERPOLATION METHODS

Many methods produce new motions by interpolating or blending existing content from a database (Rose et al., 1998; Park et al., 2002; Kovar and Gleicher, 2004; Mukai and Kuriyama, 2005). Unfortunately, these methods typically require extensive pre-processing which generally involves some type of time-warping to align the original sequences. Furthermore, the resulting motions often grossly violate dynamics, resulting in artifacts such as "footskate" and thereby requiring extensive clean-up using inverse kinematics.

2.1.3 TRANSFORMING EXISTING MOTION

Another method is to transform motion in the training data to new sequences by learning to adjust its style or other characteristics (Urtasun et al., 2004; Hsu et al., 2005; Torresani et al., 2007). Such approaches have produced impressive results given user-supplied motion content but we seek more powerful methods that can synthesize both style and content.

2.1.4 PHYSICS-BASED METHODS

Models based on the physics of masses and springs have produced some impressive results by using sophisticated "energy-based" learning methods (LeCun et al., 1998) to estimate physical parameters from motion capture data (Liu et al., 2005). However, if we want to generate realistic human motion, we need to model all the complexities of the real dynamics which is extremely difficult to do analytically. In this paper we focus on model driven analysis and synthesis but avoid the complexities involved in imposing physics-based constraints, relying instead on a "pure" learning approach in which all the knowledge in the model comes from the data.

2.1.5 GENERATIVE MODELS

Data from modern motion capture systems is high-dimensional and contains complex nonlinear relationships among the components of each observation, which is typically a series of joint angles with respect to some skeletal structure. This is a challenge for existing approaches to sequence modeling. However, there are examples of successes in the literature. Brand and Hertzmann (2000) model style and content of human motion with hidden Markov models (HMMs) whose emission distributions depend on stylistic parameters learned directly from the data. Their approach permits sampling of novel sequences from the model and applying new styles to existing content. HMMs, however, cannot efficiently model mocap data due to their simple, discrete state. Linear dynamical systems, on the other hand, have a more powerful hidden state but they cannot model the complex

nonlinear dynamics created by the nonlinear properties of muscles, contact forces of the foot on the ground and myriad other factors. This problem has been addressed by applying piecewise-linear models to synthesize motion (Pavlovic et al., 2001; Li et al., 2002; Bissacco, 2005). In general, exact inference and learning is intractable in such models and approximations are costly and difficult to evaluate.

2.1.6 GAUSSIAN PROCESS MODELS

Models based on Gaussian processes (GPs) have received a great deal of recent attention, especially in the tracking literature. The Gaussian process dynamical model (Wang et al., 2008) extends the Gaussian process latent variable model (GP-LVM) (Lawrence, 2004) with a GP-based dynamical model over the latent representations. This model has been shown to discover interesting structure in motion data and permit synthesis of simple actions. However, the main concern with GP-based approaches is their computational expense (cubic in the number of training examples for learning, quadratic in the number of training examples for prediction or generation). This problem may be alleviated by sparse methods but this remains to be seen. Another downside of the GPDM is that a single model cannot synthesize multiple types of motion, a limitation of the simple manifold structure and unimodal dynamics learned by these models. Recently proposed models such as the multifactor GP (Wang et al., 2007) and hierarchical GP-LVMs (Lawrence and Moore, 2007) address this limitation.

3. Conditional Restricted Boltzmann Machines

We have emphasized that models with distributed hidden state are necessary for efficiently modeling complex time series. But using distributed representations for hidden state in directed models of time series (Bayes nets) makes inference difficult in all but the simplest models (HMMs and linear dynamical systems). If, however, we use a restricted Boltzmann machine (RBM) to model the probability distribution of the observation vector at each time frame, the posterior over latent variables factorizes completely, making inference easy. In this section, we first review the RBM and then propose a simple extension to capture temporal dependencies yet maintain its most important computational properties: simple, exact inference and efficient approximate learning using the contrastive divergence algorithm.

3.1 Restricted Boltzmann Machines

The restricted Boltzmann machine (Smolensky, 1986) is a Boltzmann machine with a special structure (Figure 1c). It has a layer of visible units fully connected to a layer of hidden units but no connections within a layer. This bi-partite structure ensures that the hidden units are conditionally independent given a setting of the visible units and vice-versa. Simplicity and exactness of inference are the main advantages to using an RBM compared to a fully connected Boltzmann machine.

To make the distinction between visible and hidden units clear, we use v_i to denote the state of visible unit i and h_j to denote the state of hidden unit j. We also distinguish biases on the visible units, a_i from biases on the hidden units, b_j. The RBM assigns a probability to any joint setting of the visible units, $\mathbf{v}$ and hidden units, $\mathbf{h}$:

$$p(\mathbf{v},\mathbf{h}) = \frac{\exp\left(-E(\mathbf{v},\mathbf{h})\right)}{Z} \tag{1}$$

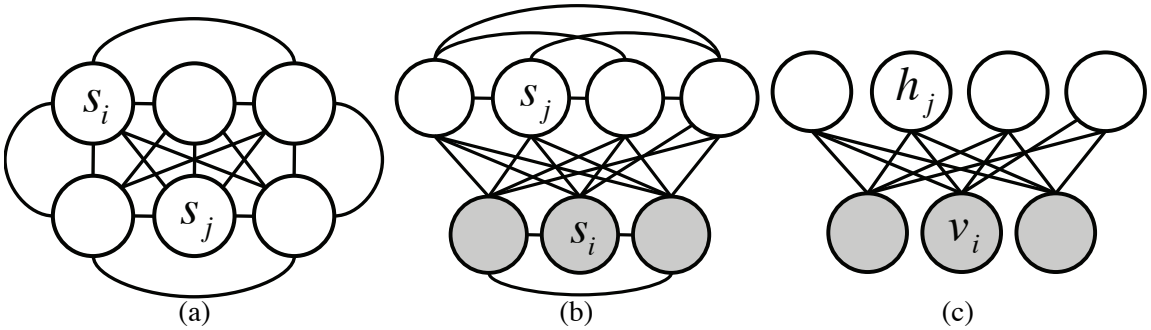


Figure 1: a) A Boltzmann machine. b) A Boltzmann machine partitioned into visible (shaded) and hidden units. c) A restricted Boltzmann machine.

where $E(\mathbf{v},\mathbf{h})$ is an energy function. When both the visible and the hidden units are binary with states 1 and 0, the energy function is

$$E(\mathbf{v},\mathbf{h}) = -\sum_{ij} W_{ij} v_i h_j - \sum_i a_i v_i - \sum_j b_j h_j$$

where Z is a normalization constant called the partition function, whose name comes from statistical physics. The partition function is intractable to compute exactly as it involves a sum over the (exponential) number of possible joint configurations:

$$Z = \sum_{v',h'} E(\mathbf{v}',\mathbf{h}').$$

Marginalizing over the hidden units in Equation 1 and maximizing the likelihood leads to a very simple maximum likelihood weight update rule:

$$\Delta W_{ij} \propto \langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{model}}. \tag{2}$$

where $\langle\cdot\rangle_{\text{data}}$ is an expectation with respect to the data distribution and $\langle\cdot\rangle_{\text{model}}$ is an expectation with respect to the model's equilibrium distribution. Because of the conditional independence properties of the RBM, we can easily obtain an unbiased sample of $\langle v_i h_j \rangle_{\text{data}}$ by clamping the visible units to a vector in the training data set, and sampling the hidden units in parallel according to

$$p(h_j = 1|\mathbf{v}) = \frac{1}{1+\exp(-b_j - \sum_i W_{ij} v_i)}. \tag{3}$$

This is repeated for each vector in a representative "mini-batch" from the training set to obtain an empirical estimate of $\langle v_i h_j \rangle_{\text{data}}$ To compute $\langle v_i h_j \rangle_{\text{model}}$ requires us to obtain unbiased samples from the joint distribution $p(\mathbf{v},\mathbf{h})$. However, there is no known algorithm to draw samples from this distribution in a practical amount of time. We can perform alternating Gibbs sampling by iterating between sampling from $p(\mathbf{h}|\mathbf{v})$ using Equation 3 and sampling from $p(\mathbf{v}|\mathbf{h})$ using

$$p(v_i = 1|\mathbf{h}) = \frac{1}{1+\exp(-a_i - \sum_j W_{ij} h_j)}. \tag{4}$$

However, Gibbs sampling in high-dimensional spaces typically takes too long to converge. Empirical evidence suggests that rather than running the Gibbs sampler to convergence, learning works well if we replace Equation 2 with

$$\Delta W_{ij} \propto \langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{recon}}, \tag{5}$$

where the second expectation is with respect to the distribution of "reconstructed" data. The reconstruction is obtained by starting with a data vector on the visible units and alternating between sampling all of the hidden units using Equation 3 and all of the visible units using Equation 4 K times. The learning rules for the biases are just simplified versions of Equation 5:

$$\begin{aligned} \Delta a_i &\propto \langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{recon}}, \\ \Delta b_j &\propto \langle h_j \rangle_{\text{data}} - \langle h_j \rangle_{\text{recon}}. \end{aligned} \tag{6}$$

The above procedure is not maximum likelihood learning but it corresponds to approximately following the gradient of another function called the contrastive divergence (Hinton, 2002). We use the notation CD-K to denote contrastive divergence using K full steps of alternating Gibbs sampling after first inferring the states of the hidden units for a datavector from the training set. Typically K is set to 1, but recent results show that gradually increasing K with learning can significantly improve performance at an additional computational cost that is roughly linear in K (Carreira-Perpinan and Hinton, 2005).

3.2 RBMs with Real-Valued Observations

Typically, RBMs use stochastic binary units for both the visible data and hidden variables, but for many applications the observed data is non-binary. For some domains (e.g., modeling handwritten digits) we can normalize the data and use the real-valued probabilities of the binary visible units in place of their activations. When we use mean-field logistic units to model data that is very non-binary (e.g., modeling patches of natural images), it is difficult to obtain sharp predictions for intermediate values and so it is more desirable to use units that match the distribution of the data.

Fortunately, the stochastic binary units of RBMs can be generalized to any distribution that falls in the exponential family (Welling et al., 2005). This includes multinomial units, Poisson units and linear, real-valued units that have Gaussian noise (Freund and Haussler, 1992). To model real-valued data (e.g., mocap), we use a modified RBM with binary logistic hidden units and real-valued Gaussian visible units. The joint probability of $\mathbf{v}$ and $\mathbf{h}$ follows the form of Equation 1 where the energy function is now

$$E(\mathbf{v},\mathbf{h}) = \sum_i \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{ij} W_{ij} \frac{v_i}{\sigma_i} h_j - \sum_j b_j h_j.$$

where a_i is the bias of visible unit i, b_j is the bias of hidden unit j and σ_i is the standard deviation of the Gaussian noise for visible unit i. The symmetric weight, W_{ij}, connects visible unit i to hidden unit j.

Any setting of the hidden units makes a linear contribution to the mean of each visible unit:

$$p(v_i|\mathbf{h}) = \mathcal{N}\left(a_i + \sigma_i \sum_j W_{ij} h_j, \sigma_i^2\right). \tag{7}$$

Inference simply uses a scaled form of Equation 3:

$$p(h_j = 1|\mathbf{v}) = \frac{1}{1+\exp(-b_j - \sum_i W_{ij}\frac{v_i}{\sigma_i})}.$$

Given the hidden units, the distribution of each visible unit is defined by a parabolic log likelihood function that makes extreme values very improbable. For any setting of the parameters, the gradient of the quadratic term with respect to a visible unit will always overwhelm the gradient due to the weighted input from the binary hidden units provided the value v_i of a visible unit is far enough from its bias, a_i. Conveniently, the contrastive divergence learning rules remain the same as in an RBM with binary visible units.

Finally, a brief note about σ_i: it is possible to learn, but this is difficult using CD-1 (Hinton, 2010). In practice, we simply rescale our data to have zero mean and unit variance and fix σ_i to be 1. Provided no noise is added to the mean reconstructions given by Equation 7, this makes the learning work well even though we would expect a good model to predict the data with much higher precision. For the remainder of the paper, we will assume $\sigma_i = 1$, but that no noise is added to the reconstructions used for learning.

3.3 The Conditional RBM

The RBM models static frames of data, but does not incorporate any temporal information. We can model temporal dependencies by treating the visible variables in the previous time slice(s) as additional fixed inputs. We add two types of directed connections: autoregressive connections from the past N configurations (time steps) of the visible units to the current visible configuration, and connections from the past M configurations of the visible units to the current hidden configuration. The addition of these directed connections turns the RBM into a conditional RBM (Figure 2). The autoregressive weights can model linear, temporally local structure very well, leaving the hidden units to model nonlinear, higher-level structure.

N and M are tunable parameters and need not be the same for both types of directed connections. To simplify discussion, we will assume $N = M$ and refer to N as the order of the model. Typically, in our experiments, we use a small number such as $N = 3$. In modeling motion capture with higher frame rates, we have found that a good rule of thumb is to set $N = F/10$ where F is the frame rate of the data (in frames per second).

To simplify the presentation, we will assume the data at $t-1,\ldots,t-N$ is concatenated into a "history" vector which we call $\mathbf{v}_{<t}$. So if $\mathbf{v}_t$ is of dimension D, then $\mathbf{v}_{<t}$ is of dimension $N \cdot D$. We will use k to index the individual, scalar components of $\mathbf{v}_{<t}$. The autoregressive parameters are summarized by an $N \cdot D \times D$ weight matrix called A and the directed "past to hidden" parameters are summarized by an $N \cdot D \times H$ matrix B where H is the number of binary hidden units. This does not change the computation, but allows us to simplify the presentation of the following equations as we can avoid explicitly summing over past frames.

3.3.1 INFERENCE AND LEARNING

Fortunately, inference in the CRBM is no more difficult than in the standard RBM. The states of the hidden units are determined by both the input they receive from the current observation and the input they receive from the recent past. Given $\mathbf{v}_t$ and $\mathbf{v}_{<t}$, the hidden units at time t are conditionally

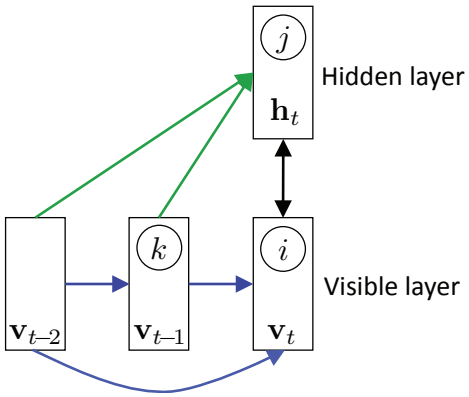


Figure 2: Architecture of the CRBM. In this figure we show $N = 2$ but in our experiments, we typically use a slightly higher order. There are no connections between the hidden units at different time steps (see Section 3.4.3).

independent. The effect of the past on each hidden unit can be viewed as a dynamic bias:

$$\hat{b}_{j,t} = b_j + \sum_k B_{kj} v_{k,<t}$$

which includes the static bias, b_j, and the contribution from the past. This slightly modifies the factorial distribution over hidden units: b_j in Equation 3 is replaced with $\hat{b}_{j,t}$ to obtain

$$p(h_{j,t} = 1 | \mathbf{v}_t, \mathbf{v}_{<t}) = \frac{1}{1 + \exp(-\hat{b}_{j,t} - \sum_i W_{ij} v_{i,t})}. \tag{8}$$

Note that we are now conditioning on $\mathbf{v}_{<t}$. Figure 3 shows an example of frame-by-frame inference in a trained CRBM.

The past has a similar effect on the visible units. The reconstruction distribution becomes

$$p(v_{i,t} | \mathbf{h_t}, \mathbf{v}_{<t}) = \mathcal{N}\left(\hat{a}_{i,t} + \sum_j W_{ij} h_{j,t}, 1\right) \tag{9}$$

where $\hat{a}_{i,t}$ is also a dynamically changing bias that is an affine function of the past:

$$\hat{a}_{i,t} = a_i + \sum_k A_{ki} v_{k,<t}.$$

We can still use contrastive divergence for training the CRBM. The updates for the symmetric weights, W, as well as the static biases, $\mathbf{a}$ and $\mathbf{b}$, have the same form as Equation 5 and Equation 6 but have a different effect because the states of the hidden units are now influenced by the previous visible units. The updates for the directed weights are also based on simple pairwise products. The

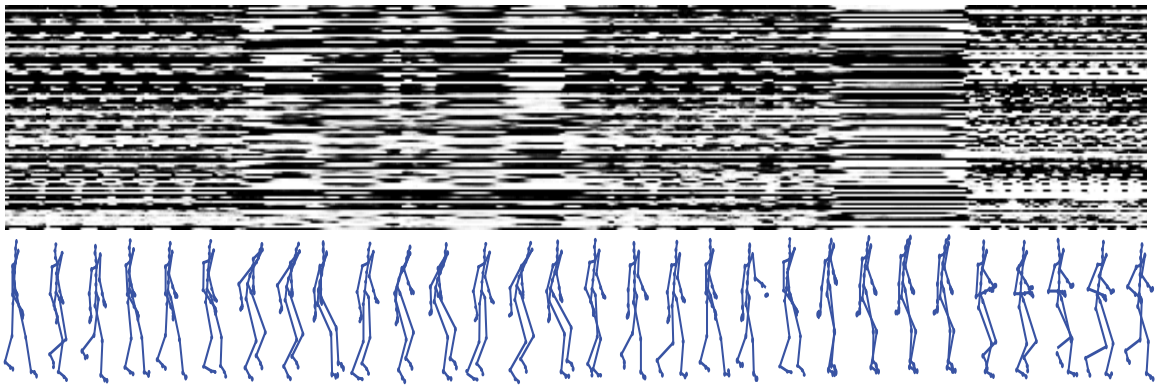


Figure 3: In a trained model, probabilities of each feature being "on", conditional on the data at the visible units. Shown is a 100-hidden unit model and a sequence which contains (in order) walking, sitting/standing (three times), walking, crouching, and running. Rows represent features, columns represent sequential frames.

gradients are now summed over all time steps:

$$\Delta W_{ij} \propto \sum_t \left(\langle v_{i,t} h_{j,t}\rangle_{\text{data}} - \langle v_{i,t} h_{j,t}\rangle_{\text{recon}}\right), \tag{10}$$

$$\Delta A_{ki} \propto \sum_t \left(\langle v_{i,t} v_{k,<t}\rangle_{\text{data}} - \langle v_{i,t} v_{k,<t}\rangle_{\text{recon}}\right), \tag{11}$$

$$\Delta B_{kj} \propto \sum_t \left(\langle h_{j,t} v_{k,<t}\rangle_{\text{data}} - \langle h_{j,t} v_{k,<t}\rangle_{\text{recon}}\right), \tag{12}$$

$$\Delta a_i \propto \sum_t \left(\langle v_{i,t}\rangle_{\text{data}} - \langle v_{i,t}\rangle_{\text{recon}}\right), \tag{13}$$

$$\Delta b_j \propto \sum_t \left(\langle h_{j,t}\rangle_{\text{data}} - \langle h_{j,t}\rangle_{\text{recon}}\right) \tag{14}$$

where $\langle\cdot\rangle_{\text{data}}$ is an expectation with respect to the data distribution, and $\langle\cdot\rangle_{\text{recon}}$ is the K-step reconstruction distribution as obtained by alternating Gibbs sampling, starting with the visible units clamped to the training data.

While learning a CRBM, we do not need to proceed sequentially through the training data sequences. The updates are only conditional on the past N time steps, not the entire sequence. As long as we isolate "chunks" of $N+1$ frames (the size depending on the order of the directed connections), these small windows can be mixed and formed into mini-batches. To speed up the learning, we assemble these chunks of frames into "balanced" mini-batches of size 100.

We randomly assign chunks to different mini-batches so that the chunks in each mini-batch are as uncorrelated as possible. To save computer memory, time frames are not actually replicated in mini-batches; we simply use indexing to simulate the "chunking" of frames.

3.3.2 SCORING OBSERVATIONS

The CRBM defines a joint probability distribution over a data vector, $\mathbf{v}_t$, and a vector of hidden states, $\mathbf{h}_t$, conditional on the recent past, $\mathbf{v}_{<t}$:

$$p(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t}) = \frac{\exp\left(-E\left(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t}\right)\right)}{Z(\mathbf{v}_{<t})} \tag{15}$$

where the partition function, Z, is constant with respect to $\mathbf{v}_t$ and $\mathbf{h}_t$ but depends on $\mathbf{v}_{<t}$. As in the RBM, it is intractable to compute exactly because it involves an integration over all possible settings of the visible and hidden units:

$$Z(\mathbf{v}_{<t}) = \sum_{\mathbf{h}'_t} \int_{\mathbf{v}'_t} \exp\left(\left(-E(\mathbf{v}'_t, \mathbf{h}'_t | \mathbf{v}_{<t}\right)\right) d\mathbf{v}'_t.$$

The energy function is given by

$$E(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t}) = \frac{1}{2} \sum_i (v_{i,t} - \hat{a}_{i,t})^2 - \sum_{ij} W_{ij} v_{i,t} h_{j,t} - \sum_j \hat{b}_{j,t} h_{j,t} \tag{16}$$

where we have assumed $\sigma_i = 1$. The probability of observing $\mathbf{v}_t$ can be expressed by marginalizing out the binary hidden units:

$$p(\mathbf{v}_t | \mathbf{v}_{<t}) = \sum_{\mathbf{h}_t} p(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t}) = \frac{\sum_{\mathbf{h}_t} \exp\left(-E(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t})\right)}{Z(\mathbf{v}_{<t})}. \tag{17}$$

Under the CRBM, the probability of observing a *sequence*, $\mathbf{v}_{(N+1):T}$, given $\mathbf{v}_{1:N}$, is just the product of all the local conditional probabilities:

$$p(\mathbf{v}_{(N+1):T} | \mathbf{v}_{1:N}) = \prod_{t=N+1}^{T} p(\mathbf{v}_t | \mathbf{v}_{<t}). \tag{18}$$

We do not attempt to model the first N frames of each sequence, though a separate set of biases could be learned for this purpose.

Although the partition function makes Equation 17 and Equation 18 intractable to compute exactly, we can exploit the fact that the hidden units are binary and integrate them out to arrive at the "free energy":

$$F(\mathbf{v}_t | \mathbf{v}_{<t}) = \frac{1}{2} \sum_i (v_{i,t} - \hat{a}_{i,t})^2 - \sum_j \log\left(1 + \exp(\sum_i W_{ij} v_{i,t} + \hat{b}_{j,t})\right), \tag{19}$$

which is a function of the model parameters and recent past. It is the negative log probability of an observation plus $\log Z$ (see Equation 15). Given a history, the free energy allows us to score a single temporal frame of observations under a fixed setting of the parameters, but unlike a probability it does not let us compare between models.[1] It can still be useful, however, in making deterministic forward predictions (as described in the following section). Freund and Haussler (1992) give details on deriving the free energy for an RBM.

1. Different models will have different partition functions.

3.3.3 GENERATION

Generation from a learned CRBM can be done on-line. The visible states at the last few time steps determine the effective biases of the visible and hidden units at the current time step. We always keep the previous visible states fixed and perform alternating Gibbs sampling to obtain a joint sample from the CRBM. This picks new hidden and visible states that are compatible with each other and with the recent (visible) history. To start alternating Gibbs sampling, we need to initialize with either $\mathbf{v}_t$ or $\mathbf{h}_t$. For time-series data that is smooth (e.g., mocap), a good choice is to initially set $\mathbf{v}_t = \mathbf{v}_{t-1}$. In practice, we alternate 30 to 100 times, though the quality of generated data does not seem to be sensitive to this parameter.

Generation does not require us to retain the training data set, but it does require initialization with N observations. Typically we use randomly drawn consecutive frames from the training data as an initial configuration.

A trained CRBM has the ability to fill in missing data (complete or partial observations), regardless of where the dropouts occur in a sequence. To be strictly correct, we would need to use smoothing (i.e., conditioning on future as well as past observations) in order to take into account the effect of a filled-in value on the probability of future observed values. As in the learning procedure, we ignore smoothing and this approximation allows us to fill in missing data on-line. Filling in missing data with the CRBM is very similar to generation. We simply clamp the known data to the visible units, initialize the missing data to something reasonable (for example, the value at the previous frame), and alternate between stochastically updating the hidden and visible units, *with the known visible states held fixed*.

The noise in sampling may be an asset when using the CRBM to generate sequences, but when using the CRBM to fill in missing data, or in a predictive setting it may be undesirable. Rather than obtaining a sample, we may want the model's "best guess". Given the model parameters, and past history, we can follow the negative gradient of the free energy (Equation 19) with respect to either a complete or partial setting of the visible variables, $\mathbf{v}_t$:

$$\frac{\partial F(\mathbf{v}_t|\mathbf{v}_{<t})}{\partial v_{k,t}} = v_{k,t} - \left(\hat{a}_{i,t} + \sum_j W_{ij} f\left(-\sum_i W_{ij} v_{i,t} - \hat{b}_{j,t}\right)\right)$$

where $f(\cdot)$ is the logistic function. The gradient at a unit has an intuitive form: it is the difference between its current value and the value that would be obtained by mean-field reconstruction. We use conjugate-gradient optimization, but any general purpose gradient-based optimizer is suitable.

3.4 Higher Level Models: The Conditional Deep Belief Network

Once we have trained the model, we can add layers in the same way as a deep belief network (DBN) (Hinton et al., 2006). The previous layer CRBM is kept, and the sequence of hidden state vectors, while driven by the data, is treated as a new kind of "fully observed" data. The next level CRBM has the same architecture as the first (though we can alter the number of its units) and is trained in the exact same way. Upper levels of the network can then model higher-order structure.

Figure 4a shows a CRBM whose symmetric, undirected weights have been represented explicitly by two sets of directed weights: top-down "generative" weights W_0, and bottom-up "recognition" weights, W_0^T. This representation is purely illustrative: it does not at all change the model. The use of the zero subscripts and superscripts simply indicates that the CRBM is first in a series of layers which we will introduce shortly.

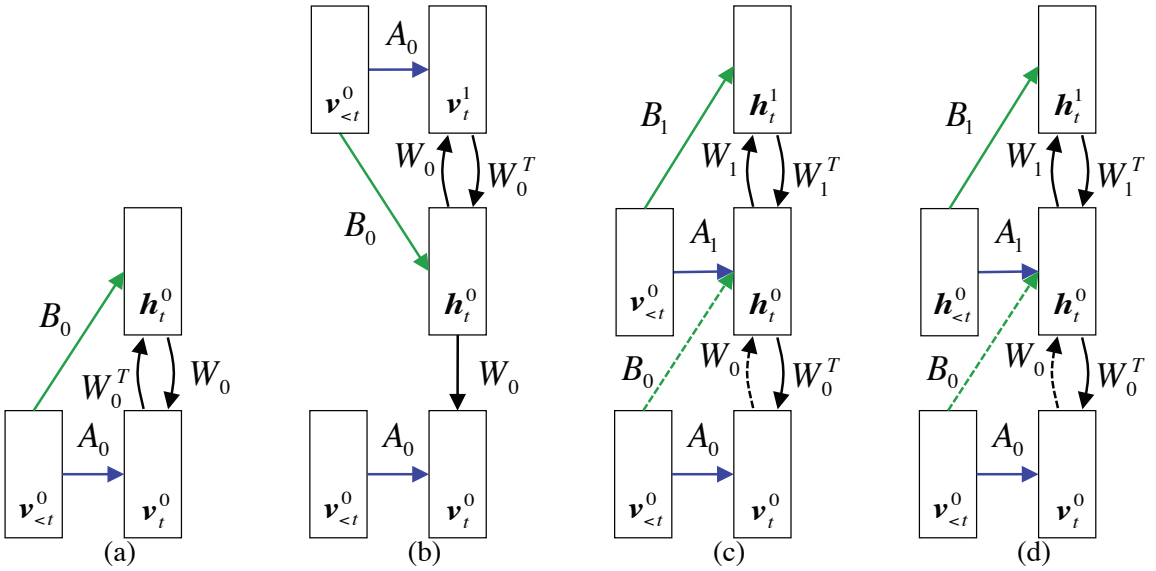


Figure 4: Building a conditional deep belief network. (a) The CRBM. (b) A generative model whose weights between layers are tied. It defines the same joint distribution over $\mathbf{v}_t^0$ and $\mathbf{h}_t^0$. The top two layers interact using symmetric connections while all other connections are directed. (c) Improving the model by untying the weights; holding W_0, A_0 and B_0 fixed and greedily training W_1, A_1 and B_1. Note that the "dashed" directed, bottom-up weights are not part of the generative model. They are used to infer factorial, approximate posterior distributions over $\mathbf{h}_t^0$ when $\mathbf{v}_t^0$ is clamped to the data. (d) The model we use in practice. Note the change from $\mathbf{v}_{<t}^0$ to $\mathbf{h}_{<t}^0$. We ignore uncertainty in the past hidden states.

Figure 4b shows a generative model that is equivalent to the original CRBM in the sense that their joint distributions over $\mathbf{v}_t^0$ and $\mathbf{h}_t^0$, conditional on $\mathbf{v}_{<t}^0$ are the same. We have added a second set of visible units, $\mathbf{v}_t^1$, identical to the first, and ensured that the undirected, symmetric weights between $\mathbf{v}_t^1$ and $\mathbf{h}_t^0$ are equal to the weights used in the original CRBM. Furthermore, we introduce a copy of $\mathbf{v}_{<t}^0$ and the autoregressive connections, A_0. The weights are therefore "tied" between the two layers. Additionally, the bottom-up weights between $\mathbf{v}_t^0$ and $\mathbf{h}_t^0$, W_0^T, are no longer part of the generative model in Figure 4b. Although the model defines the same joint distribution, its semantics are very different than the CRBM. To generate an observation, $\mathbf{v}_t^0$, conditional on $\mathbf{v}_{<t}^0$, we must reach equilibrium in the conditional associative memory formed by the top two layers and then perform a single down-pass using directed weights W_0 and A_0. The CRBM generates observations as explained in Section 3.3.3.

Note that if we observe $\mathbf{v}_t^0$, the units $\mathbf{h}_t^0$ are no longer conditionally independent because the undirected connections between $\mathbf{v}_t^0$ and $\mathbf{h}_t^0$ have been replaced by directed connections. The new model is therefore subject to the effects of "explaining away". However, because of the tied weights, the CRBM at the top two layers becomes a "complementary prior" (Hinton et al., 2006): meaning that when we multiply the likelihood term by the prior, the posterior is factorial. Researchers who are used to using directed models often assume that $W_0\mathbf{v}_t^0 + B_0\mathbf{v}_{<t}^0$ is computing a likelihood term.

This is incorrect. It is computing the product of the likelihood term and the prior term (i.e., the posterior). Both the likelihood term and the prior term are much more complicated since they are each far from being factorial.

If we hold W_0, A_0 and B_0 fixed, but "untie"[2] the weights between the top two layers (Figure 4c) we can improve the generative model by greedily learning W_1, A_1 and B_1, treating the activations of $\mathbf{h}_t^0$ while driven by the training data as a kind of "fully-observed" data. When the weights are untied, units in the topmost layer no longer represent the visible units, but another layer of latent features, $\mathbf{h}_t^1$. We can still use W_0^T and B_0 (which are not part of the generative model) to infer factorial *approximate* posterior distributions over the states of $\mathbf{h}_t^0$.

The joint distribution defined by the original CRBM, $p(\mathbf{v}_t^0, \mathbf{h}_t^0 | \mathbf{v}_{<t}^0)$, decomposes into a mapping from features to data, $p(\mathbf{v}_t^0 | \mathbf{h}_t^0, \mathbf{v}_{<t}^0)$, and an implicit prior over the features, $p(\mathbf{h}_t^0 | \mathbf{v}_{<t}^0)$ which is also determined by W_0. We can think of training the next layer as a means of improving the prior model. By fixing W_0, A_0, the distribution $p(\mathbf{v}_t^0 | \mathbf{h}_t^0, \mathbf{v}_{<t}^0)$ is unchanged. The gain from building a better model of $p(\mathbf{h}_t^0 | \mathbf{v}_{<t}^0)$ more than offsets the loss from having to perform approximate inference. This greedy learning algorithm can be applied recursively to any number of higher layers and is guaranteed to never decrease a variational lower bound on the log probability of the data under the full generative model (Hinton et al., 2006).

In practice, greedily training multiple layers of representation works well. However, there are a number of small changes we make to gain flexibility and improve the computational cost of performing inference and learning. Bending the rules as follows breaks the above guarantee:

1. We replace maximum likelihood learning with contrastive divergence (for obvious computational reasons).

2. The guarantee relies on initializing the weights of each successive layer with the weights in the layer below. This assumes that all odd layers are of equal size and all even layers of equal size. In practice, however, we typically violate this constraint and initialize the weights to small random values.

3. Rather than train each layer conditional on $\mathbf{v}_{<t}^0$ (which we assume to be the fully-observed recent past of the visible units), we train each layer using its own recent past as the conditioning input. $\mathbf{v}_{<t}^0, \mathbf{h}_{<t}^0, \ldots, \mathbf{h}_{<t}^{H-1}$ (where H is the number of hidden layers), always treating the past as fully-observed.

The model that we use in practice is shown in Figure 4d. It is a conditional deep belief network (CDBN). The inference we perform in this model, conditional on past visible states, is approximate because it ignores the future (it does not do smoothing). Because of the directed connections, exact inference within the model should include both a forward and backward pass through each sequence. We perform only a forward pass because smoothing is intractable in the multi-layer model. Effectively, at each layer we replace the full posterior by an approximate filtering distribution. However, there is no guarantee that this is a good approximation. Compared with an HMM, the lack of smoothing is a loss. But the deep model is still exponentially more powerful at using its hidden state to represent data.

2. A note on our naming convention: the "untied" A_0 becomes B_1 since it now represents a visible-to-hidden connection. The "untied" B_0 becomes A_1 since it will ultimately be a "visible-visible" connection when the hidden units are treated as observed during greedy learning.

3.4.1 ON-LINE GENERATION WITH HIGHER-LEVEL MODELS

The generative model for a conditional DBN consists of a top-level conditional associative memory (with symmetric weights and dynamic biases) and any number of directed lower layers (with top-down generative weights and dynamic generative biases). We also maintain bottom-up connections that are used in approximate inference. Like in a DBN, to generate a sample, $\mathbf{v}_t$, the associative memory must settle on a joint setting of the units in the top two hidden layers and then the top-down weights are used to generate the lower layers. Since the model is conditional, each layer must also consider the effect of the past via the dynamic biases. Note that in a deep network, all but the topmost hidden layer will have two sets of dynamic biases: recognition biases from when it was greedily trained as a hidden layer, and generative biases from when it was subsequently trained as a "visible" layer. During generation, we must be careful not to double-count the input to each layer (i.e., by including both types of biases when computing the total input to each unit); we use the recognition biases during inference and generative biases during generation.

As a concrete example, let us consider generating an observation from a conditional DBN built by greedily training two CRBMs (the same network shown in Figure 4d).

1. If the first CRBM is order N and the second CRBM is order M then we must initialize with $N+M$ frames, $\mathbf{v}_{1:(N+M)}$ (Figure 5a).

2. Next, we initialize M frames of the first hidden layer using a mean-field up-pass through the first CRBM (Figure 5b).

3. Then we initialize the first layer hidden units at $t = N+M+1$ to be a copy of the real-valued probabilities we have just inferred at $t = N+M$. We perform alternating Gibbs sampling in the 2nd layer CRBM. At each step, we stochastically activate the top-level hidden units, but on the final step, we suppress noise by using the real-valued probabilities of the top layer to obtain the real-valued probabilities of the first layer hidden units (Figure 5c).

4. We do a mean-field down-pass in the first layer CRBM to obtain the visible states at time $t = N+M+1$ (Figure 5d).

Again, we copy the real-valued probabilities of the first layer hidden units to initialize Gibbs sampling for the next frame, and repeat steps 3 and 4 above for as many frames as desired.

3.4.2 FINE-TUNING

Following greedy learning, both the weights and the simple inference procedure are suboptimal in all but the top layer of the network, as the weights have not changed in the lower layers since their respective stage of greedy training. We can, however, use a contrastive form of the "wake-sleep" algorithm (Hinton et al., 1995) called the "up-down" algorithm (Hinton et al., 2006) to fine-tune the generative model. In our experiments, we have observed that fine-tuning improves the visual quality of generated sequences at a modest additional computational cost.

3.4.3 TEMPORAL LINKS BETWEEN HIDDEN UNITS

In a conditional restricted Boltzmann machine the hidden state and visible state depend only on past instances of the visible variables. The CRBM is a special case of the temporal restricted Boltzmann machine (TRBM) (Sutskever and Hinton, 2007) in which there are no temporal connections

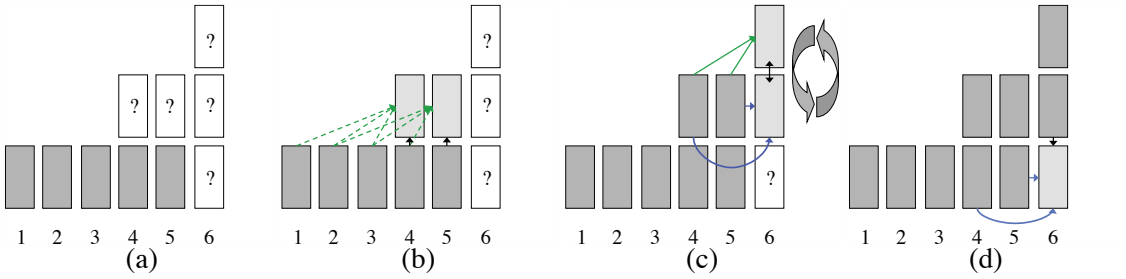


Figure 5: Generating from a conditional deep belief network with two hidden layers. For this example, we assume the first layer CRBM is third order and the second layer CRBM is second order. We provide five frames to initialize the model.

between hidden units. This makes filtering in the CRBM exact, and "mini-batch" learning possible, as training does not have to be done sequentially. This latter property can greatly speed up learning as well as smooth the learning signal, as the order of data vectors presented to the network can be randomized. This ensures that the training cases in each mini-batch are as uncorrelated as possible.

As soon as we introduce connections between hidden units, we must resort to approximate filtering or deterministic methods (Sutskever et al., 2009) even in a single layer model. In training higher-level models using CRBMs, we gain hidden-to-hidden links via the autoregressive connections of the higher layers. At each stage of greedy learning, filtering is exact within each CRBM. However, filtering in the overall multi-layer model is approximate.

3.5 Experiments

We have carried out a series of experiments training CRBM models on motion capture data from publicly available repositories. After learning a model using the updates described in Section 3.3, we can demonstrate in several ways what it has learned about the structure of human motion. Perhaps the most direct demonstration, which exploits the fact that it is a probability density model of sequences, is to use the model to generate *de-novo* a number of synthetic motion sequences. Supplemental video files of these sequences are available on the website mentioned in the abstract; these motions have not been retouched by hand in any motion editing software. Note that we also do not have to keep a reservoir of training data sequences for generation - we only need the weights of the trained model and N valid frames for initialization. Our model is, therefore, suitable for low-memory devices.[3] More importantly, we believe that compact models are likely to be better at generalization.

3.5.1 DATA SOURCE AND REPRESENTATION

The first data set used in these experiments was obtained from `http://mocap.cs.cmu.edu`. It will be hereafter referred to as the CMU data set. The second data set used in these experiments was released by Hsu et al. (2005). We obtained it from from `http://people.csail.mit.edu/ehsu/work/sig05stf/`. It will be hereafter referred to as the MIT data set. The data consisted of 3D joint angles derived from 30 (CMU) or 17 (MIT) markers plus a root orientation and displacement.

3. The level of compression obtained will of course vary with the number of free parameters and size of the data set.

Data was represented with the encoding described in Appendix A. The final dimensionality of our data vectors was 62 (CMU) and 49 (MIT).

One advantage of the CRBM is the fact that the data does not need to be heavily preprocessed or dimensionality reduced before learning. Other generative approaches (Brand and Hertzmann, 2000; Li et al., 2002) apply PCA to reduce noise and dimensionality. However, dimensionality reduction becomes problematic when a wider range of motions is to be modeled. The autoregressive connections can be thought of as doing a kind of "whitening" of the data.

3.5.2 DETAILS OF LEARNING

Except where noted, all CRBM models were trained as follows: Each training case was a window of $N+1$ consecutive frames and the order of the training cases was randomly permuted. The training cases were presented to the model as "mini-batches" of size 100 and the weights were updated after each mini-batch. Models were trained using CD-1 (see Section 3.1) for a fixed number of epochs (complete passes through the data). All parameters used a learning rate of 10^{-3}, except for the autoregressive weights which used a learning rate of 10^{-5}. A momentum term was also used: 0.9 of the previous accumulated gradient was added to the current gradient. All parameters (excluding biases) used L2 weight decay of 0.0002.

3.5.3 GENERATION OF WALKING AND RUNNING SEQUENCES FROM A SINGLE MODEL

In our first demonstration, we train a single CRBM on data containing both walking and running motions; we then use the learned model to generate both types of motion, depending on how it is initialized. We extracted 23 sequences of walking and 10 sequences of running from subject 35 in the CMU data set. After downsampling to 30Hz, the training data consisted of 2813 frames. We trained a 200 hidden-unit CRBM for 4000 passes through the training data, using a third-order model (for directed connections). The order of the sequences was randomly permuted such that walking and running sequences were distributed throughout the training data.

Figure 6 shows a walking sequence and a running sequence generated by the same model, using alternating Gibbs sampling (with the probability of hidden units being "on" conditional on the current and previous three visible vectors). Since the training data does not contain any transitions between walking and running (and *vice-versa*), the model will continue to generate walking or running motions depending on where it is initialized.

3.5.4 LEARNING TRANSITIONS BETWEEN WALKING AND RUNNING

In our second demonstration, we show that our model is capable of learning not only several types of homogeneous motion content but also the transitions between them when the training data itself contains examples of such transitions. We trained on 9 sequences (from the MIT database, file `Jog1_M`) containing long examples of walking and running, as well as a few transitions between the two gaits. After downsampling to 30Hz, this provided us with 2515 frames. Training was done as before, but after the model was trained, an identical 200 hidden-unit model was trained on top of the first model (see Section 3.4). The resulting two-level model was used to generate data. A video available on the website demonstrates our model's ability to stochastically transition between various types of motion during a single generated sequence.

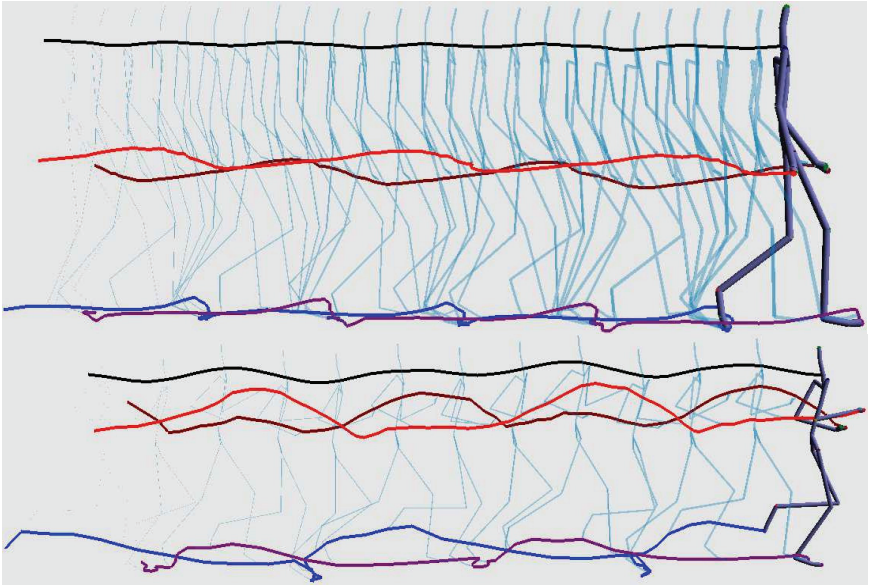


Figure 6: After training, the same model can generate walking (top) and running (bottom) motion (see supplemental videos). Each skeleton is 4 frames apart.

3.5.5 INTRODUCING TRANSITIONS USING NOISE

In our third demonstration, we show how transitions between different types of motion content can be generated even when such transitions are absent in the data. We use the same model and data as described in Section 3.5.3, where we have learned on separate sequences of walking and running. To generate, we use the same sampling procedure as before, except that at each time we stochastically choose the hidden states (given the current and previous three visible vectors) we add a small amount of Gaussian noise to the hidden state biases. This encourages the model to explore more of the hidden state space without deviating too far from the current motion. Applying this "noisy" sampling approach, we see that the generated motion occasionally transitions between learned gaits. These transitions appear natural (see the supplemental video).

3.5.6 LEARNING MOTION STYLE

We have demonstrated that the CRBM can generate and transition between different gaits, but what about its ability to capture more subtle stylistic variation within a particular gait? We also seek to show the CRBM's ability to learn on data at a higher frame-rate (60Hz), and from a much larger training corpus. Finally, we incorporate label information into our training procedure.

From the CMU data set, we extracted a series of 10 stylized walk sequences performed by subject 137. The walks were labeled as *cat, chicken, dinosaur, drunk, gangly, graceful, normal, old-man, sexy* and *strong*. We balanced the data set by repeating the sequences three to six times (depending on the original length) so that our final data set contained approximately 3000 frames of each style at 60fps.

In general, we used the same training procedure as above, but made a few important changes:

- At each iteration of CD learning, we performed 10 steps of alternating Gibbs sampling (CD-10)
- We added a sparsity term to the energy function to gently encourage the hidden units, while driven by the data, to have an average activation of 0.2 (details below)
- At each iteration of CD learning, we added Gaussian noise with $\sigma = 1$ to each dimension of the past history, $\mathbf{v}_{<t}$. This ensures that the generative model can cope with the type of noisy history that is produced when generating from the model. For the linear autoregressive parameters, A, this is equivalent to L2 regularization (Matsuoka, 1992). For the parameters which involve the binary hidden units it is not quite equivalent but has a very similar effect.

These experiments were carried out considerably later than the experiments described in Section 3.5.3 to 3.5.5 and so represent a refinement to our learning method. This allows us to cope with the higher frame rate and larger degree of variability in the training set. Recent work on estimating the partition functions of RBMs and evaluating the log probability of held-out sets has shown that models trained with CD$>$1 , although more computationally demanding to train, are significantly better generative models (Salakhutdinov and Murray, 2008). We have chosen CD-10 as a compromise between closely approximating maximum likelihood learning and minimizing computational cost.

The recent popularity of sparse, overcomplete latent representations has highlighted both the theoretical and practical motivations for their use in unsupervised learning (Olshausen and Field, 1997; Lee and Seung, 1999; Ranzato et al., 2006; Lee et al., 2008). Sparse representations are often more easy to interpret, and also more robust to noise. Furthermore, evidence suggests that they may be used in biological systems. Recent sparse "energy-based methods" (Ranzato et al., 2006, 2007, 2008) have proposed the use of sparsity as an alternative to contrastive divergence learning. The "contrastive term" in CD (which represents the derivative of the log partition function) corresponds to pulling up on the energy (or pushing down on the probability) of points outside the training set. Another way to ensure that the energy surface is low only around the training set is to eliminate the partition function and replace it with a term that limits the volume of the input space over which the energy surface can take low value (Ranzato et al., 2008). Using sparse overcomplete latent representations is a means of limiting this volume by minimizing the information content of the latent representation. Using both a contrastive term and sparsity, as we have done here, is a two-fold approach to sculpting energy surfaces.

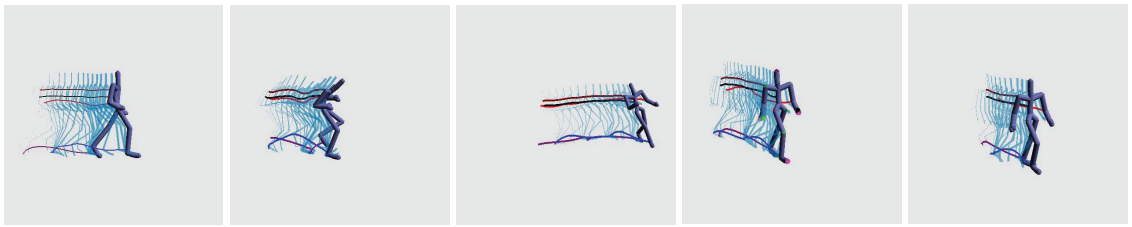
To implement sparsity, we maintained a damped "average activation" estimate for each hidden unit. Each element of this vector was initialized to the target activation, 0.2. Every time we presented a mini-batch, we updated the estimate to be 0.9 times its current value plus 0.1 times the average activation of the hidden units while the visible units were clamped to the data. The average was taken over the mini-batch. After we calculated the positive-phase (data) and negative-phase (after K steps of Gibbs sampling) statistics for each parameter, we added, to the original gradient, the gradient of the cross-entropy error between the updated activity estimate and the target, 0.2, with respect to that parameter. Note that updates for visible-only parameters (e.g., autoregressive weights and visible biases) were unaffected by the sparsity term. Our sparsity term is similar to the one used by Lee et al. (2008). However, they used a squared-error penalty between average activation and target while we used cross-entropy error (Nair and Hinton, 2009) which is more appropriate for logistic units.

1-layer Model. A single-layer CRBM with 1200 hidden units and $N = 12$ was trained for 200 epochs on data for 10 different walking styles, with the parameters being updated after every 100 training cases. Each training case was a window of 13 consecutive frames and the order of the training cases was randomly permuted. In addition to the real-valued mocap data, the hidden units received additive input from a "one-hot" encoding of the matching style label through another matrix of weights. Respecting the conditional nature of our application (generation of stylized motion, as opposed to, say classification) this label was not reconstructed during learning. After training the model, we generated motion by initializing with 12 frames of training data and holding the label units clamped to the style matching the initialization.

With a single layer we could generate high-quality motion of 9/10 styles (see the supplemental videos), however, the model failed to produce good generation of the *old-man* style. We believe that this relates to the subtle nature of this particular motion. In examining the activity of the hidden units over time while clamped to training data, we observed that the model devotes most of its hidden capacity to capturing the more "active" styles as it pays a higher cost for failing to model more pronounced frame-to-frame changes.

2-layer Model. We also learned a deeper network by first training a CRBM with 600 binary hidden units and real-valued visible units and then training a higher-level CRBM with 600 binary hidden and 600 binary visible units. Both models used $N = 12$. The data for training the higher-level CRBM consisted of the activation probabilities of the hidden units of the first CRBM while driven by the training data. Style labels were only connected to the top-layer of this network, while training the second level CRBM. The first-level model was trained, without style labels, for 300 epochs and the second-level model was trained for 120 epochs.

After training, the 2-hidden-layer network was able to generate high-quality walks of all styles, including *old-man* (see Figure 7 and the supplemental videos). The second level CRBM layer effectively replaces the prior over the first layer of hidden units, $p(\mathbf{h}_t|\mathbf{v}_{<t})$, that is implicitly defined by the parameters of the first CRBM. This provides a better model of the subtle correlations between the features that the first-level CRBM extracts from the motion. The superiority of the second layer may indeed be a result of its ability to capture longer-term dependencies in the data. Learning the *old-man* style is conditional on capturing longer-term dependencies since the signal (representing joint angles) changes more slowly. The 2-layer network has access to a wider temporal context and therefore is better able to model this particular style. We thank one of the anonymous reviewers for suggesting this explanation.



(a) Cat (b) Dinosaur (c) Graceful (d) Sexy (e) Strong

Figure 7: Generating different walking styles from the same conditional deep belief network with two hidden layers.

Qualitative Comparison to the GPDM. The Gaussian process dynamical model (GPDM) (Wang et al., 2008) was proposed for human motion synthesis and for use as a prior in tracking (Urtasun et al., 2006). It extends the Gaussian process latent variable model (GP-LVM) (Lawrence, 2004) with a GP-based dynamical model over the latent representations. However, as we demonstrate in this section, the model has difficulty in capturing multiple styles of motion due to its simple manifold structure and unimodal dynamics.

We used two publicly available GPDM implementations, each with its own suggested hyperparameters and structural settings. The first implementation was provided by Neil Lawrence's FGPLVM toolbox: `http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/fgplvm`. Lawrence recommends fixing by hand, rather than optimizing, the hyperparameters of the dynamics GP (Lawrence, 2006). This ensures a strong preference for smooth trajectories. For the dynamics, we used the default compound (RBF + white noise) kernel with recommended hyperparameter settings of $\alpha = [0.2, 0.001, 1 \times 10^{-6}]^T$. The observation model used a compound (RBF + bias + white noise) kernel whose hyperparameters were optimized.

The second implementation we employed was provided by Jack Wang: `http://www.dgp.toronto.edu/~jmwang/gpdm/`. This implementation differed from the first in a number of respects. First, it used a compound (linear + RBF + white noise) dynamics kernel whose hyperparameters were optimized rather than set by hand. The observation model used a compound (RBF + white noise) kernel whose hyperparameters were optimized. This GPDM also "balanced" the objective function by reweighting the dynamics term by the ratio of observed dimensions to latent dimensions (Wang et al., 2008). Similar to fixing hyperparameters, this encourages smoothness of the latent trajectories.

With each implementation we trained both a single model on the complete 10 walking styles data set as well as 10 style-specific models. The data was preprocessed identically to the CRBM experiments, however, we did not balance the data set by repeating sequences. It would have taken several weeks to train the GPDM on a corpora of approximately 30,000 frames. We tried each of the sparse approximations provided by the FGPLVM toolbox to reduce the computational complexity. In our experience, though drastically improving training time, all of the approximations led to far worse synthesized motion quality. In all results shown, we use the recommended 3 latent dimensions (Lawrence, 2006; Urtasun et al., 2006; Wang et al., 2008). We also experimented with 8 and 16 latent dimensions but found that this caused quality to decrease.

Similar to the online process used for drawing samples from a CRBM, we simulated the dynamical process one frame at a time, starting from training data (mapped to latent space). At each time step, we set the latent position to the mean latent position conditioned on the previous step. The latent trajectory then induces a per-frame Gaussian distribution over (normalized) poses (i.e., the reconstruction distribution). We take the mean of this distribution for each frame. Wang et al. (2008) recommend drawing fair samples of entire trajectories using hybrid Monte Carlo (HMC), using the simulated latent trajectory as an initialization. We did not observe any significant improvement in the quality of synthesized motion when using HMC. Moreover, it increased simulation time by an order of magnitude.

The supplemental videos show the result of synthesizing motion styles from the GPDM. For each model and style we show three sequences: 1) a sequence generated from the same initialization as we used for the CRBM; 2) the best sequence, as determined by visual inspection, over ten different initializations spaced uniformly over the training data; and 3) reconstructing the training data using the latent representation. We observed that when trained per-style, both implementations

of the GPDM could generate reasonable-looking motion, though not of the same quality as a 1 or 2-layer CRBM trained on all styles. Regardless of the implementation, a single GPDM trained on all styles failed to generate satisfactory motion. More recent extensions of the GP-LVM, such as Topologically-constrained GP-LVMs (Urtasun et al., 2008), multifactor GPs (Wang et al., 2007) and hierarchical GP-LVMs (Lawrence and Moore, 2007) may perform better at this task.

3.5.7 FILLING IN MISSING DATA

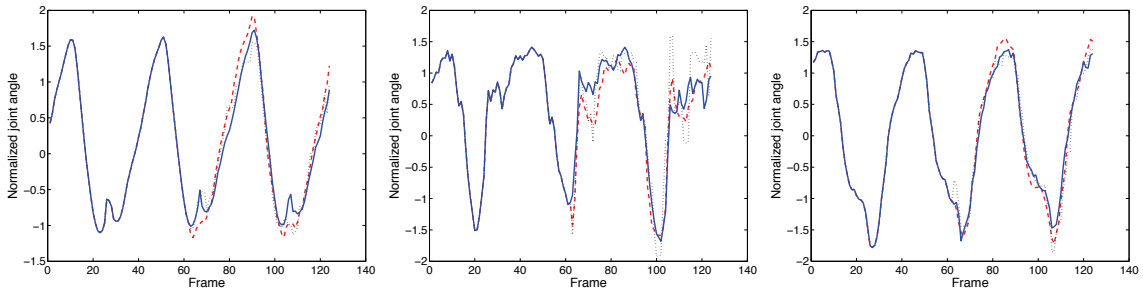


Figure 8: The model successfully fills in missing data using only the previous values of the joint angles (through the temporal connections) and the current angles of other joints (through the symmetric connections). Shown are the three angles of rotation for the left hip joint. The original data is shown as a solid line, the model's prediction is shown as a dashed line, and the results of nearest neighbour interpolation are shown as a dotted line.

Due to the nature of the motion capture process, which can be adversely affected by lighting and environmental effects, as well as noise during recording, motion capture data often contains missing or unusable data. Some markers may disappear ("dropout") for long periods of time due to sensor failure or occlusion. The majority of motion editing software packages contain interpolation methods to fill in missing data, but this leaves the data unnaturally smooth. These methods also rely on the starting and end points of the missing data. Hence, if a marker goes missing until the end of a sequence, naïve interpolation will not work. Such methods often only use the past and future data from the single missing marker to fill in that marker's missing values. Since joint angles are highly correlated, substantial information about the placement of one marker can be gained from the others. To demonstrate filling in, we trained a model exactly as described in Section 3.5.3, holding out one walking and one running sequence from the training data to be used as test data. For each of these walking and running test sequences, we erased two different sets of joint angles, starting halfway through the test sequence. These sets were the joints in (1) the left leg, and (2) the entire upper body. As seen in the supplemental video, the quality of the filled-in data is excellent and is hardly distinguishable from the original ground truth of the test sequence. Figure 8 demonstrates the model's ability to predict the three angles of rotation of the left hip.

We report results on the held-out walking sequence, of length 124 frames. We compared our model's performance to nearest neighbour interpolation, a simple method where for each frame, the values on known dimensions are compared to each example in the training set to find the closest match (measured by Euclidean distance in the normalized angle space). The unknown dimensions are then filled in using the matched example. As reconstruction from our model is stochastic,

we repeated the experiment 100 times and report the mean. For the missing leg, mean squared reconstruction error per joint using our model was 8.78, measured in normalized joint angle space, and summed over the 62 frames of interest. Using nearest neighbour interpolation, the error was greater: 11.68. For the missing upper body, mean squared reconstruction error per joint using our model was 20.52. Using nearest neighbour interpolation, again the error was greater: 22.20.

We note that by adding additional neighbouring points, the nearest neighbour prediction can be significantly improved. For filling in the left leg, we found that $K = 8$ neighbours gave minimal error (8.63), while for the missing upper body, using $K = 6$ neighbours gave minimal error (12.67). These scores, especially in the case of the missing upper body, are, in fact, an improvement over using the CRBM for prediction. However, we note that in practice we would not be able to fine-tune the number of nearest neighbours nor could we be expected to have access to a large database of extremely similar training data. In more realistic missing-data scenarios, we would expect the model-based approach to generalize much better. Furthermore, we have not optimized other tunable parameters such as the model order, number of Gibbs steps per CD iteration, and number of hidden units; all of which are expected to have an impact on the prediction error.

4. Factored Conditional Restricted Boltzmann Machines

In this section we present a different model, based on the CRBM, that explicitly preserves the CRBM's most important computational properties but includes multiplicative three-way interactions that allow the effective interaction weight between two units to be modulated by the dynamic state of a third unit. We factor the three-way weight tensor implied by the multiplicative model, greatly reducing the number of parameters.

4.1 Multiplicative Interactions

A major motivation for the use of RBMs is that they can be used as the building blocks of deep belief networks (DBN), which are learned efficiently by training greedily, layer-by-layer (see Section 3.4). DBNs have been shown to learn very good generative models of handwritten digits (Hinton et al., 2006), but they have difficulty modeling patches of natural images. This is because RBMs have no simple way to capture the smoothness constraint in natural images: a single pixel can usually be predicted very accurately by simply interpolating its neighbours.

To address this concern, Osindero and Hinton (2008) introduced the semi-restricted Boltzmann machine (SRBM). In an SRBM, the constraints on the connectivity of the RBM are relaxed to allow lateral connections between the *visible* units in order to model the pair-wise correlations between inputs, thus allowing the hidden units to focus on modeling higher-order structure. Semi-restricted Boltzmann machines also permit deep networks. Each time a new level is added, the previous top layer of units is given lateral connections, so, after the layer-by-layer learning is complete, all layers except the topmost contain lateral connections between units. SRBMs make it possible to learn deep belief nets that model image patches much better, but they still have strong limitations that can be seen by considering the overall generative model. The equilibrium sample generated at each layer influences the layer below by controlling its effective biases. The model would be much more powerful if the equilibrium sample at the higher level could also control the lateral interactions at the layer below using a three-way, multiplicative relationship. Memisevic and Hinton (2007) introduced the gated CRBM, which permitted such multiplicative interactions and showed that it was able to learn rich distributed representations of image transformations (see Section 4.3).

In this section, we explore the idea of multiplicative interactions in the context of a different type of CRBM. Instead of gating lateral interactions with hidden units, we allow a set of real-valued style variables to gate the three types of connections: autoregressive, past to hidden, and visible to hidden within the CRBM. We will use the term "sub-model" to refer to a set of connections of a given type. Our modification of the CRBM architecture does not change the desirable properties related to inference and learning but allows the style variables to modulate the interactions in the model.

Like the CRBM, the multiplicative model is applicable to general time series where conditional data is available (e.g., seasonal variables for modeling rainfall occurrences, economic indicators for modeling financial instruments). However, we are largely motivated by our success thus far in modeling mocap data. In Section 3 we showed that a CRBM could capture many different styles with a single set of parameters. Generation of different styles was purely based on initialization, and the model architecture did not allow control of transitions between styles nor did it permit style blending. By using explicit style variables to gate the connections of a CRBM, we can obtain a much more powerful generative model that permits controlled transitioning and blending. We demonstrate that in a conditional model, the gating approach is superior to simply using labels to bias the hidden units, which is the approach most commonly used in static models (Hinton et al., 2006).

4.2 Style and Content Separation

There has been a significant amount of work on the separation of style and content in motion. The ability to separately specify the style (e.g., sad) and the content (e.g., walk to location A) is highly desirable for animators. One approach to style and content separation is to guide a factor model (e.g., PCA, factor analysis, ICA) by giving it "side-information" related to the structure of the data. Tenenbaum and Freeman (2000) considered the problem of extracting exactly two types of factors, namely style and content, using a bilinear model. In a bilinear model, the effect of each factor on the output is linear when the other is held fixed, but together the effects are multiplicative. This model can be learned efficiently, but supports only a rigid, discrete definition of style and content requiring that the data be organized in a (style $\times$ content) grid.

Previous work has looked at applying user-specified style to an existing motion sequence (Urtasun et al., 2004; Hsu et al., 2005; Torresani et al., 2007). The drawback to these approaches is that the user must provide the content. We propose a generative model for content that adapts to stylistic controls. Recently, models based on the Gaussian process latent variable model (Lawrence, 2004) have been successfully applied to capturing style in human motion (Wang et al., 2007). The advantage of our approach over such methods is that our model does not need to retain the training data set (just a few frames for initialization). Furthermore, training time increases linearly with the number of frames of training data, and so our model can scale up to massive data sets, unlike the kernel-based methods which are cubic in the number of frames. The powerful distributed hidden state of our model means that it does not suffer from the limited representational power of HMM-based methods of modeling style (e.g., Brand and Hertzmann, 2000).

4.3 Gated Conditional Restricted Boltzmann Machines

Memisevic and Hinton (2007) introduced a way of implementing multiplicative interactions in a conditional model. The gated CRBM was developed in the context of learning transformations between image pairs. The idea is to model an observation (the output) given its previous instance

(the input). For example, the input and output might be neighbouring frames of video. The gated CRBM has two equivalent views: first, as gated regression (Figure 9a), where hidden units can blend "slices" of a transformation matrix into a linear regression, and second as modulated filters (Figure 9b) where input units gate a set of basis functions used to reconstruct the output. In the latter view, each setting of the input units defines an RBM. This means that conditional on the input, inference and learning in a gated CRBM are tractable.

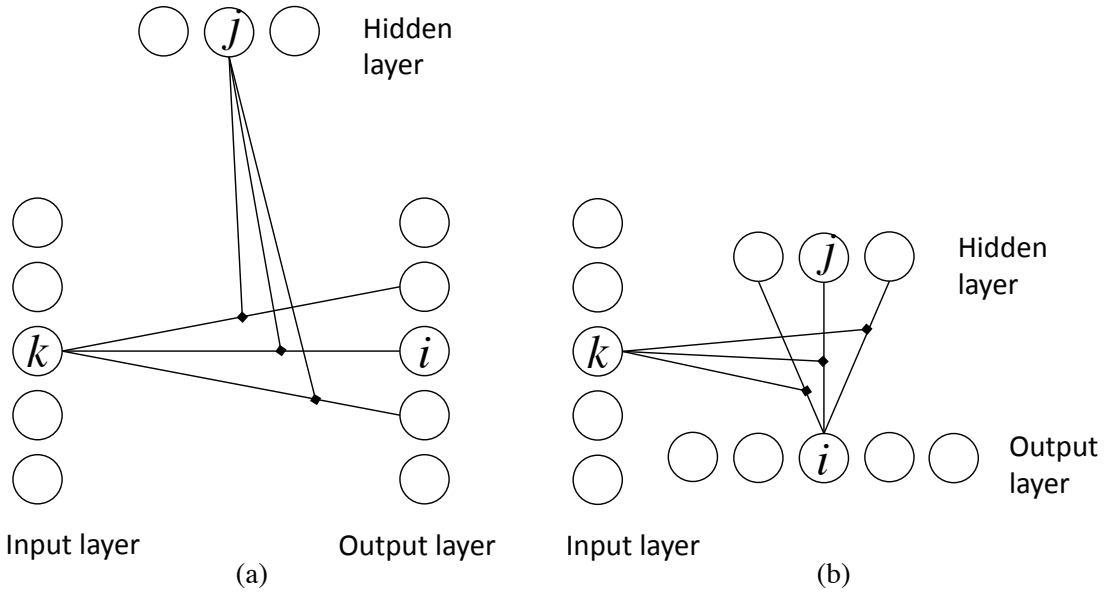


Figure 9: Two views of the gated CRBM, reproduced from the original paper (Memisevic and Hinton, 2007).

For ease of presentation, let us consider the case where all input, output, and hidden variables are binary (the extension to real-valued input and output variables is straightforward). As in Equation 15, the gated CRBM describes a joint probability distribution through exponentiating an energy function and renormalizing. This energy function captures all possible correlations between the components of the input, $\mathbf{x}$, the output, $\mathbf{v}$, and the hidden variables, $\mathbf{h}$:

$$E(\mathbf{v},\mathbf{h}|\mathbf{x}) = -\sum_{ijk} W_{ijk} v_i h_j x_k - \sum_{ij} c_{ij} v_i h_j - \sum_i a_i v_i - \sum_j b_j h_j \tag{20}$$

where a_i, b_j index the standard biases on each unit and c_{ij} index the gated biases, which shift the total input to a unit conditionally. The parameters W_{ijk} are the components of a three-way weight tensor. The CD weight updates for learning a gated CRBM are similar to a standard CRBM (Ackley et al., 1985). For example, the weight update rule for W_{ijk} is:

$$\Delta W_{ijk} \propto \langle v_i h_j x_k \rangle_{\text{data}} - \langle v_i h_j x_k \rangle_{\text{recon}}.$$

Considering the "modulated filters" view of the gated CRBM, we can fold the (given) inputs into the weights to express the input-dependent filters $\hat{W}_{ij} = \sum_k W_{ijk} x_k$. This allows us to rewrite the energy

function (Equation 20) as:

$$E(\mathbf{v},\mathbf{h}|\mathbf{x}) = -\sum_{ij}\hat{W}_{ij}v_i h_j - \sum_{ij}c_{ij}v_i h_j - \sum_i a_i v_i - \sum_j b_j h_j.$$

Fixing the input, the first term is bilinear in $\mathbf{v}$ and $\mathbf{h}$. Therefore at first glance, the model appears similar to the bilinear factor model (Tenenbaum and Freeman, 2000). However, the two models differ considerably in both their learning method and structure. Note that the bilinearity only occurs in the energy function: the gated CRBM permits the learned transformations to be highly nonlinear functions of the data.

4.4 Factoring

To model time-series, we can consider the output of a gated CRBM to be the current frame of data, $\mathbf{v} = \mathbf{v}_t$, and the input to be the previous frame (or frames), $\mathbf{x} = \mathbf{v}_{<t} = \mathbf{v}_{t-N:t-1}$. In this sense, the gated CRBM is a kind of autoregressive model where a transformation is composed from a set of basis transformations, with each binary hidden unit specifying whether or not to include one of the basis transformations. The number of possible compositions is exponential in the number of hidden units, but the componential nature of the hidden units prevents the number of parameters in the model from becoming exponential, as it would in a mixture model. Because of the three-way weight tensor, the number of parameters is cubic (assuming that the numbers of input, output and hidden units are comparable).

In many applications, including human motion modeling, strong underlying regularities in the data suggest that structure can be captured using three-way, multiplicative interactions but with less than the cubically many parameters implied by the weight tensor. This motivates us to factor the interaction tensor into a product of pairwise interactions (Figure 10). Factoring changes the energy function (Equation 20) to:

$$E(\mathbf{v},\mathbf{h}|\mathbf{x}) = -\sum_f \sum_{ijk} W^{\mathbf{v}}_{if} W^{\mathbf{h}}_{jf} W^{\mathbf{x}}_{kf} v_i h_j x_k - \sum_{ij} c_{ij} v_i h_j - \sum_i a_i v_i - \sum_j b_j h_j$$

where f indexes a set of deterministic factors. Superscripts differentiate the different types of pairwise interactions: $W^{\mathbf{v}}_{if}$ connect output units to factors (undirected), $W^{\mathbf{h}}_{jf}$ connect hidden units to factors (undirected), and $W^{\mathbf{x}}_{kf}$ connect input units to factors (directed).

The factors correspond to an intermediate layer of "simple cells" which modulate the interactions between units. Each factor is connected to all input units, all hidden units, and all output units. However, there are typically about as many factors as the number of each type of unit, so the introduction of factors corresponds to a kind of low-rank approximation to the interaction tensor, W, that uses about $3N^2$ parameters instead of N^3. Factors are deterministic, and are therefore very different than the visible and hidden units, which have stochastic states. Factors always send the product of the total input from two types of units to the remaining third type of unit. For example, during inference, each factor collects the total input arriving at it from the input and output layers, respectively, multiplies these quantities together, and sends this input to each hidden unit. During reconstruction, each factor collects the total input arriving at it from the input and hidden layers, respectively, multiplies these quantities together, and sends this input to each visible unit. This is in contrast to the visible and hidden units. These must be sampled before sending their stochastic states to the factors, and, unlike factors, they send the same message everywhere. Factors cannot be

replaced by a layer of nonlinear stochastic units because this would prevent the hidden states from being conditionally independent.

Although factoring has been motivated by the introduction of multiplicative interactions, models that only involve pairwise interactions can also be factored (e.g., Salakhutdinov et al., 2007). To factor the CRBM, we change the energy function in Equation 16 to:

$$E(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t}) = \frac{1}{2} \sum_i (v_{i,t} - \hat{a}_{i,t})^2 - \sum_f \sum_{ij} W^{\mathbf{v}}_{if} W^{\mathbf{h}}_{jf} v_{i,t} h_{j,t} - \sum_j \hat{b}_{j,t} h_{j,t}$$

and additionally, factor the weights of the dynamic biases $\hat{\mathbf{a}}_t$ and $\hat{\mathbf{b}}_t$:

$$\hat{a}_{i,t} = a_i + \sum_m \sum_k A^{\mathbf{v}}_{im} A^{\mathbf{v}_{<t}}_{km} v_{k,<t},$$
$$\hat{b}_{j,t} = b_j + \sum_n \sum_k B^{\mathbf{h}}_{jn} B^{\mathbf{v}_{<t}}_{kn} v_{k,<t}.$$

The indices m and n correspond to the factoring of directed connections, A and B. We may use a different number of factors for each of the three different types of connections in the CRBM. This procedure can be seen as a kind of learned low-rank matrix factorization on each of W,A, and B.

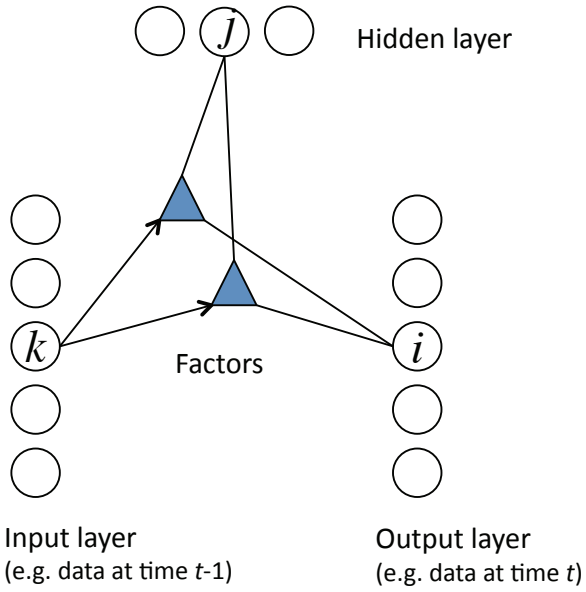
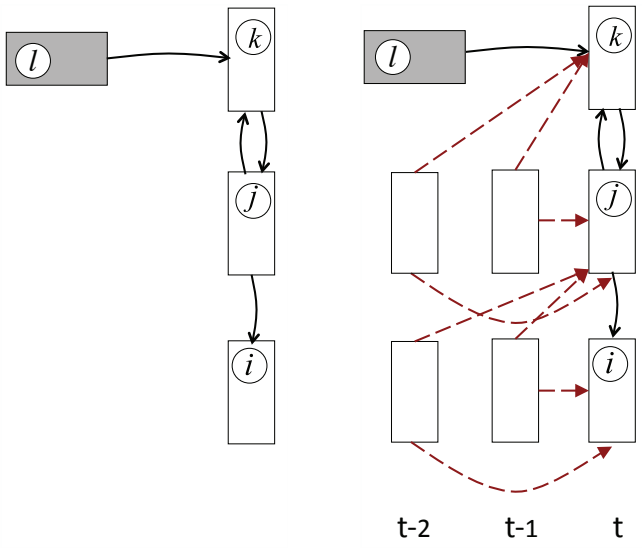


Figure 10: Factoring the gated CRBM.

4.5 A Style-Gated, Factored Model

We now consider modeling multiple styles of human motion using factored, multiplicative, three-way interactions. Hinton et al. (2006) showed that a good generative model of handwritten digits could be built by connecting a softmax label unit to the topmost hidden layer of a DBN (Figure 11a). After learning, clamping a label changes the energy landscape of the autoassociative model formed by the top two layers, so that performing alternating Gibbs sampling produces a joint sample compatible with a particular digit class. It is easy to extend this modification to the CRBM,

where discrete style labels bias the hidden units. In a CRBM, however, the hidden units are also conditioned on information from the past that is much stronger than the information coming from the label (Figure 11b). The model has learned to respect consistency of styles between frames and so will resist a transition introduced by changing the label units.



(a) (b)

Figure 11: a) In a deep belief network, clamping the label units changes the energy function. b) In a conditional model, label information is swamped by the signal coming from the past.

As in the gated CRBM, we are motivated to let style change the *interactions* of the units as opposed to simply their effective biases. Memisevic and Hinton (2010) used factored three-way interactions to allow the hidden units of a gated CRBM to control the effect of one video frame on the subsequent video frame. Figure 12 shows a different way of using factored three-way interactions to allow real-valued style features, derived from discrete style labels, to control three different sets of pairwise interactions. Like the standard CRBM (Equation 15), the model defines a joint probability distribution over $\mathbf{v}_t$ and $\mathbf{h}_t$, conditional on the past N observations, $\mathbf{v}_{<t}$. However, the distribution is also conditional on the style labels, $\mathbf{y}_t$, through a set of deterministic, real-valued features, $\mathbf{z}_t$. The features are a linear function of the "one-hot" encoded style labels:

$$z_{l,t} = \sum_p R_{pl} y_{p,t}.$$

This resembles the use of componential word-features used in Mnih and Hinton's language model (Mnih and Hinton, 2007).

Similar to our discussion of the CRBM, we assume binary stochastic hidden units and real-valued visible units with additive Gaussian noise and $\sigma_i = 1$. The energy function is:

$$E(\mathbf{v}_t, \mathbf{h}_t | \mathbf{v}_{<t}, \mathbf{y}_t) = \frac{1}{2} \sum_i (v_{i,t} - \hat{a}_{i,t})^2 - \sum_f \sum_{ijl} W^{\mathbf{v}}_{if} W^{\mathbf{h}}_{jf} W^{\mathbf{z}}_{lf} v_{i,t} h_{j,t} z_{l,t} - \sum_j \hat{b}_{j,t} h_{j,t}. \tag{21}$$

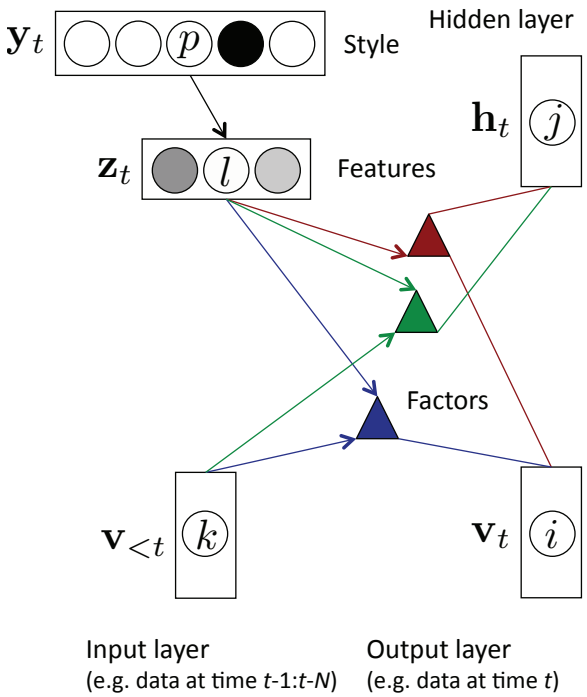


Figure 12: A factored CRBM whose interactions are gated by real-valued stylistic features.

The three terms in Equation 21 correspond to the three sub-models (the groups of links connected to each triangular factor in Figure 12). Note that for each sub-model, what was a matrix of weights is now replaced by three sets of weights connecting units to factors. The three types of weights are differentiated again by superscripts. For example, the matrix of undirected weights in the standard CRBM, W_{ij}, has been replaced by three matrices involved in a factored, multiplicative interaction: $W^{\mathbf{v}}_{if}$, $W^{\mathbf{h}}_{jf}$, and $W^{\mathbf{z}}_{lf}$. The same process is applied to the other two sub-models. Note that the three sub-models may have a different number of factors (which we index by f, m, and n).

The dynamic biases become:

$$\begin{aligned}
\hat{a}_{i,t} &= a_i + \sum_m \sum_{kl} A^{\mathbf{v}}_{im} A^{\mathbf{v}_{<t}}_{km} A^{\mathbf{z}}_{lm} v_{k,<t} z_{l,t} \\
&= a_i + \sum_m A^{\mathbf{v}}_{im} \sum_k A^{\mathbf{v}_{<t}}_{km} v_{k,<t} \sum_l A^{\mathbf{z}}_{lm} z_{l,t}, \qquad (22) \\
\hat{b}_{j,t} &= b_j + \sum_n \sum_{kl} B^{\mathbf{h}}_{jn} B^{\mathbf{v}_{<t}}_{kn} B^{\mathbf{z}}_{ln} v_{k,<t} z_{l,t} \\
&= b_j + \sum_n B^{\mathbf{h}}_{jn} \sum_k B^{\mathbf{v}_{<t}}_{kn} v_{k,<t} \sum_l B^{\mathbf{z}}_{ln} z_{l,t} \qquad (23)
\end{aligned}$$

where the dynamic component of Equation 22 and Equation 23 is simply the total input to the visible/hidden unit via the factors. The total input is a three-way product between the input to the factors (coming from the past and from the style features) and the weight from the factors to the visible/hidden unit. The dynamic biases include a static component, $\mathbf{a}$ and $\mathbf{b}$. As in the gated

CRBM, we could also add three types of gated biases, corresponding to the pairwise interactions in each of the sub-models. In our experiments, we have not used any gated biases.

4.5.1 INFERENCE AND LEARNING

Adding multiplicative interactions to the model and factoring does not change the property that the posterior distribution is factorial. Inference is performed by considering, in parallel, the total input to each hidden unit via the factors:

$$p(h_{j,t}=1|\mathbf{v}_t,\mathbf{v}_{<t},\mathbf{y}_t)=\frac{1}{1+\exp(-\hat{b}_{j,t}-\sum_f W^{\mathbf{h}}_{jf}\sum_i W^{\mathbf{v}}_{ij}v_{i,t}\sum_l W^{\mathbf{z}}_{lf}z_{l,t})}$$

where $\hat{b}_{j,t}$ is defined in Equation 23. The reconstruction distribution is found by considering the total input to each visible unit via the factors:

$$p(v_{i,t}|\mathbf{h}_t,\mathbf{v}_{<t},\mathbf{y}_t)=\mathcal{N}\left(\hat{a}_{i,t}+\sum_f W^{\mathbf{v}}_{if}\sum_j W^{\mathbf{h}}_{jf}h_{j,t}\sum_l W^{\mathbf{z}}_{lf}z_{l,t},1\right)$$

where $\hat{a}_{i,t}$ is defined in Equation 22.

As in the other models based on RBMs, exact maximum likelihood learning is intractable. However, applying contrastive divergence leads to a set of very simple gradient update rules which are the same for binary or real-valued Gaussian visible units. The gradient with respect to a weight that connects a unit to a factor is the difference of two expectations of products. Each product involves three terms: the activity of the respective unit, and the total input to the factor from each of the two other sets of units involved in the three-way relationship. For example:

$$\Delta W^{\mathbf{v}}_{if}\propto\sum_t\left(\langle v_{i,t}\sum_j W^{\mathbf{h}}_{jf}h_{j,t}\sum_l W^{\mathbf{z}}_{lf}z_{l,t}\rangle_{\text{data}}-\langle v_{i,t}\sum_j W^{\mathbf{h}}_{jf}h_{j,t}\sum_l W^{\mathbf{z}}_{lf}z_{l,t}\rangle_{\text{recon}}\right).$$

The complete set of update rules is given in Appendix C.

The weights connecting labels to features, R, can simply be learned by backpropagating the gradients obtained by CD. Since these weights affect all three sub-models, their updates are more complicated. Applying the chain rule, we obtain:

$$\Delta R_{pl}\propto\sum_t\left(\langle C_{l,t}y_{p,t}\rangle_{\text{data}}-\langle C_{l,t}y_{p,t}\rangle_{\text{recon}}\right),$$

$$C_{l,t}=\sum_f W^{\mathbf{z}}_{lf}\sum_i W^{\mathbf{v}}_{if}v_{i,t}\sum_j W^{\mathbf{h}}_{jf}h_{j,t}+\sum_m A^{\mathbf{z}}_{lm}\sum_i A^{\mathbf{v}}_{im}v_{i,t}\sum_k A^{\mathbf{v}_{<t}}_{km}v_{k,<t}+\sum_n B^{\mathbf{z}}_{ln}\sum_j B^{\mathbf{h}}_{jn}h_{j,t}\sum_k B^{\mathbf{v}_{<t}}_{kn}v_{k,<t}.$$

The updates for the static biases on the hidden and visible biases are the same as in the standard CRBM (Equation 13 and 14).

4.5.2 PARAMETER SHARING

In addition to the large reduction in the number of free parameters obtained by factoring, further savings may be obtained by tying some sets of parameters together. In the fully parameterized model (Figure 13a), there are 9 different sets (matrices) of weights but if we restrict the number of factors to be the same for each of the three sub-models, four sets of parameters are identical in dimension: the weights that originate from the inputs (past visible units), the outputs (visible units), the hidden

units and the features. Any combination of the compatible parameters may be tied. Figure 13b shows a fully-shared parameterization. This has slightly less than half the number of parameters of the fully parameterized model, assuming that the number of input, output, hidden, and feature units are comparable.

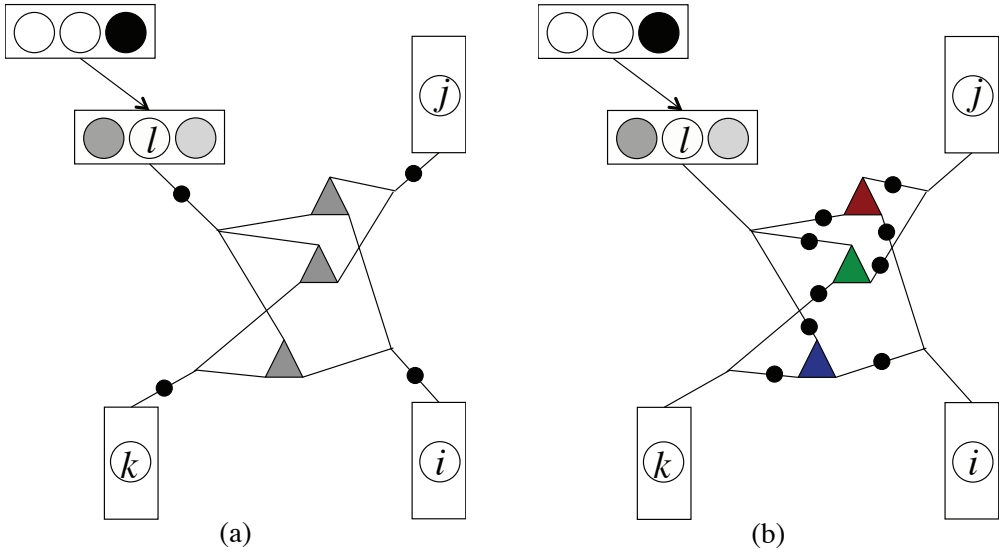


Figure 13: a) Fully parameterized model with each dot representing a different set of parameters and different colors denoting a different number of factors in each sub-model. b) Full parameter sharing where each dot represents a tied group of parameters. The number of factors is restricted to be the same for each sub-model.

In comparing different reduced parameterizations, tying only the feature-factor parameters, $W^{\mathbf{z}}_{lf}$,$A^{\mathbf{z}}_{lm}$, and $B^{\mathbf{z}}_{ln}$ led to models synthesizing the highest quality motion. When sharing the autoregressive weights $A^{\mathbf{v}_{<t}}_{km}$ and $A^{\mathbf{v}}_{im}$ with non-autoregressive weights $B^{\mathbf{v}_{<t}}_{km}$ and $W^{\mathbf{v}}_{if}$, respectively, we found that the component of the gradient related to the autoregressive model tended to dominate the weight update early in learning. This was due to the strength of the correlation between past and present compared to hidden and present or hidden and past. Witholding the autoregressive component of the gradient for the first 100 epochs, until the hidden units were able to extract interesting structure from the data, solved this problem. In our reported experiments we trained models with only the feature-factor parameters tied.

4.6 Experiments

CRBM models share a common deficiency: biasing the hidden units with a style label is not a true integration of context into their architecture. Despite our attempts, we cannot prevent spurious transitions (see Section 3.5.6), nor does a change of label during generation allow us to transition or blend between styles. We carry out a set of experiments that demonstrate that this shortcoming can be addressed by using factored, multiplicative interactions.

4.6.1 MODELING WITH DISCRETE STYLE LABELS

Using the 10-styles data set described in Section 3.5.6, we trained a factored CRBM with Gaussian visible units whose parameters were gated by 100 real-valued features driven by discrete style labels (Figure 12). This model had 600 hidden units, 200 factors per sub-model and $N = 12$. Feature-to-factor parameters were also tied between sub-models. All parameters used a learning rate of 10^{-2}, except for the autoregressive parameters, $A^{\mathbf{v}}_{im}$, $A^{\mathbf{v}_{<t}}_{km}$, $A^{\mathbf{z}}_{lm}$ and the label-to-feature parameters, R_{pl}, which used a learning rate of 10^{-3}. After training the model for 500 epochs, we tested its ability to synthesize realistic motion by initializing with 12 frames of training data and holding the label units clamped to the matching style. The single-layer model was able to generate stylized content as well as the 2-layer standard CRBM (see the supplemental videos). In addition, we were able to induce transitions between two or more styles by linearly blending the discrete style label from one setting to another over 200 frames.[4] We were further able to blend together styles (like *sexy* and *strong*) by applying a linear interpolation of the discrete labels. The resulting motion was more natural when a single style was dominant (e.g., an 0.8/0.2 blend). We believe this is simply a case of better performance when the desired motion more closely resembles the cases present in the training data set, so training on a few examples of blends should greatly improve their generation.

4.6.2 MODELING WITH REAL-VALUED STYLE PARAMETERS

The motions considered thus far have been described by a single, discrete label such as *gangly* or *drunk*. Motion style, however, can be characterized by multiple discrete labels or even continuous factors such as the level of flow, weight, time and space formally defined in Laban movement analysis (Torresani et al., 2007). In the case of multiple discrete labels, our real-valued feature units, $\mathbf{z}$, can receive input from multiple categories of labels. For continuous factors of style, we can connect real-valued style units to the real-valued feature units, or we can simply gate the model directly by the continuous description of style.

To test this latter configuration, we trained a model exactly as in Section 4.6.1, but instead of gating connections with 100 real-valued feature units, we gated with 2 real-valued style descriptors that were conditioned upon at every frame. Again we trained with walking data, but the data was captured specifically for this experiment. One style unit represented the speed of walking and the other, the stride length. The training data consisted of nine sequences at 60fps, each approximately 6000 frames corresponding to the cross-product of (slow, normal, fast) speed and (short,normal,long) stride length. The corresponding labels each had values of 1, 2 or 3. These values were chosen to avoid the special case of all gating units being set at zero and nullifying the effective weights of the model. The model was trained for 500 epochs.

After training, the model could, as before, generate realistic motion according to the nine discrete combinations of speed and stride-length with which it was trained based on initialization and setting the label units to match the labels in the training set. Furthermore, the model supported both interpolation and extrapolation along the speed and stride length axes and did not appear overly sensitive to initialization (see the supplemental videos).

4. The number of frames was selected empirically and provided a smooth transition, but the model is not sensitive to this number. A quick (e.g., frame-to-frame) change of labels will simply produce a "jerky" transition.

4.6.3 QUANTITATIVE EVALUATION

In our experiments so far, we have sought a qualitative comparison to the CRBM, based on the realism of synthesized motion. We have also focused on the ability of a factored model with multiplicative interactions to synthesize transitions as well as interpolate and extrapolate between styles present in the training data set. The application does not naturally present a quantitative comparison, but in the past, other time series models have been compared by their performance on the prediction of either full or partial held-out frames (e.g., Wang et al., 2008; Lawrence, 2007). We use the data set first proposed by Hsu et al. (2005) which consists of labeled sequences of seven types of walking: (*crouch*, *jog*, *limp*, *normal*, *side-right*, *sway*, *waddle*) each at three different speeds (*slow*, *medium*, *fast*). We preprocessed the data to remove missing or extremely noisy sections, and smoothed with a low-pass filter before downsampling from 120 to 30fps.

For each architecture: CRBM, factored CRBM, style-gated unfactored CRBM, and style-gated factored CRBM, we trained 21 different models on all style/speed pairs except one, which we held out for testing. Then, for each model, we attempted to predict every subsequence of length M in the test set, given the past $N = 6$ frames. We repeated the experiments for each architecture, each time reporting results averaged over the 21 models. Prediction could be performed by initializing with the previous frame and Gibbs sampling in the same way we generated, but this approach is subject to noise. We found that in all cases, integrating out the hidden units and following the gradient of the negative free energy with respect to the visible units gave less prediction error (see Section 3.3.3). We minimized the free energy using conjugate-gradient descent initialized with the previous frame. The architectures were subject to different learning rates and so the number of epochs for which to train each model was determined by setting aside 10% of the training set for validation.

We have also included a sixth-order autoregressive model as a baseline. This corresponds to the CRBM model without hidden units, except that it is trained using least squares instead of contrastive divergence.

Figure 14 presents the results. With almost half the number of free parameters, the 600-60 factored model performed as well as the fully parameterized CRBM. Gating with style information gives an advantage in longer-term prediction because it prevents the model from gradually changing the style. The unfactored model with style information performed slightly worse than the factored model and was extremely slow to train (it took two days to train whereas the other models were each trained in a few hours). The baseline autoregressive model performed extremely well in the short term, but was quickly eclipsed by the latent variable models for $N > 5$.

4.6.4 COMPUTATIONAL COMPLEXITY

The CRBM (1 or 2 layer) and FCRBM take a few hours to train on a modern single-core workstation. All of the models we have presented can generate motion at least as fast as 60fps (i.e., the visualizations we have produced were generated in real-time). Learning and inference in the CRBM and FCRBM are extremely efficient, with complexity linear in the number of training samples. In practice, this is slightly optimistic since larger and more complex data sets will require more hidden units, and learning and inference are also linear in the number of hidden units. The scale of corpora that we use in our experiments are problematic for GP-LVMs, since learning and inference for those models are $O(N^3)$ and $O(N^2)$, where N is the number of training samples.

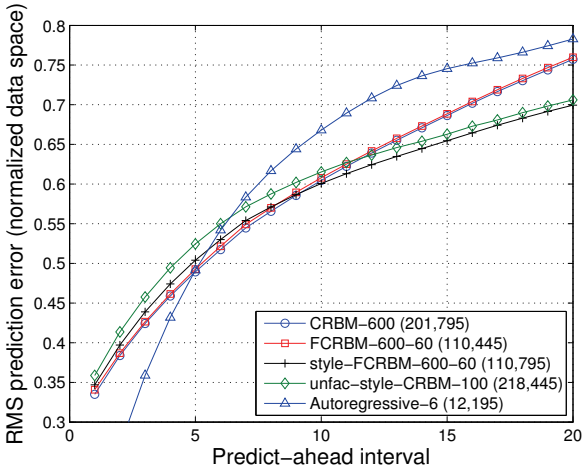


Figure 14: Prediction experiment. The number of free parameters are shown in parentheses. Error is reported in the normalized space in which the models are trained and is per-dimension, per-frame. The first two values for the autoregressive model (0.1506 and 0.2628) have been intentionally cut off.

5. Conclusion

We have introduced the conditional restricted Boltzmann machine (CRBM). The key properties of the CRBM are that it permits rich distributed representations to be learned from time series, and that exact inference is simple and efficient. We derived the contrastive divergence (CD) learning rules for CRBMs and showed how CRBMs can be stacked to form conditional deep belief nets. We demonstrated that a single model can generate many different styles of motion.

Perhaps the two greatest limitations of CRBMs (and RBMs in general) are first, evaluating the quality of trained models, and second, the learning algorithm with which they are trained. Though we have explored different methods of model evaluation, such as N-step forward prediction and the subjective assessment of synthesized data, the most natural way to evaluate a generative model is to compute the log-likelihood it assigns to a held-out test set. For all but the smallest models, this is impossible to do exactly due to the intractability of computing the partition function. Salakhutdinov and Murray (2008) have successfully applied annealed importance sampling (AIS) to RBMs. However, conditioning changes the partition function which implies that we would need to perform AIS for every possible configuration of N-frame histories (where N is the order of the CRBM) if we wish to evaluate the likelihood assigned by the model to an arbitrary sequence. Fortunately, to evaluate models we are often interested in computing likelihoods for a fixed test set rather than arbitrary sequences. This means that we need only to concern ourselves with conditioning on all possible N-frame histories in the test set. If we are evaluating M sequences whose maximum length is T, we would need to make on the order of $M(T-N)$ complete AIS estimates.[5]

A major criticism of contrastive divergence learning is that by "pulling up" on the energy of individual reconstructed data points, the algorithm fails to visit regions far away from the training

5. Note that for each of these "conditional" estimates we would still perform several runs of AIS.

data. Consequently, the bulk of the energy surface is left arbitrarily low. One solution is to abandon CD altogether, and pursue other learning methodologies, such as the sparse "energy-based methods" discussed in Section 3.5 or score matching (Hyvärinen, 2005). The alternative is to improve CD (e.g., Tieleman, 2008).

In Section 4 we extended the CRBM to permit context units to modulate the existing pairwise interactions. The resulting multiplicative model implies cardinality of parameters cubic in the number of units. However, we factorized the weights to make the parameterization quadratic and further reduced this number by tying weights. We demonstrated that the resulting model could capture several different motion styles, as well as transition and blend naturally between them. A sensible and natural extension of this work is to the fully unsupervised setting, where stylistic parameters are learned rather than provided (cf., Brand and Hertzmann, 2000).

Acknowledgments

An earlier version of this work appeared in two conference papers (Taylor et al., 2007; Taylor and Hinton, 2009). The authors thank NSERC and CIFAR for financial support. The authors also thank the anonymous reviewers for their helpful feedback. This work was primarily conducted while the first and third authors were at the University of Toronto.

Appendix A. Data Representation

The most statistically salient patterns of variation in the data may differ considerably from the patterns that humans find perceptually and expressively salient (Brand and Hertzmann, 2000). Therefore our learning algorithms can benefit from a carefully chosen representation that highlights important sources of variation and suppresses irrelevant sources of variation. Specifically, we aim to make our representation of motion invariant to rotation about the gravitational vertical (which we will simply call the vertical) and translation in the ground-plane. In the following discussion, we describe the steps taken to achieve a representation amenable to learning.

A.1 Original Representation

Data from a motion capture system typically consists of the 3D cartesian coordinates of 15-30 virtual markers (usually representing joint centres) for a series of discrete time-steps, which we call frames. The data is processed to remove missing and noisy markers and then converted to a joint angle hierarchy through an optimization that assumes constant limb lengths. For each frame, we obtain a vector of relative joint angle orientations, each 1-3 degrees of freedom (dof) plus a root orientation and translation in global coordinates (6 dof). The definition of the root depends on the data source, but typically it is the coccyx, near the base of the back. In our experiments, we used a variety of mocap sources, each of which provided the data already in a hierarchical "joint-angle" format.

A.2 Conversion to Exponential Maps

The most common representation for orientations in mocap data are Euler angles. Euler angles describe a one, two or three dof orientation by a sequence of rotations about axes in the global or

local coordinate system. The order of rotations is user-defined and is a common source of confusion, often differing between data sources. Euler angles do not permit distances between rotations to be directly computed nor do they support interpolation or optimization since the orientation space is highly nonlinear. It is also not trivial to ensure that similar poses are expressed by similar Euler angles. Euler angles also suffer from "gimbal lock", the loss of rotational degrees of freedom due to singularities in the parameter space. Equivalent representations such as the 3×3 rotation matrix or 4D quaternion are not well suited to optimization and synthesis as they require additional constraints to ensure that they remain valid. Therefore we convert joint angles to an exponential map parameterization (Grassia, 1998) before learning.

The exponential map parameterization is also known as "axis-angle" representation since it consists of a three-element vector, whose direction specifies an axis of rotation and whose magnitude specifies the angle by which to rotate about this axis. Exponential maps are well suited to interpolation, optimization and unconstrained synthesis since they are locally linear and every three-element vector maps to a valid rotation. The parameterization still contains singularities and therefore is subject to gimbal lock, but the singularities in the exponential map are often avoidable (Grassia, 1998).[6]

A.3 Conversion to Body-Centred Orientations

We treat the root specially because it encodes a transformation with respect to a fixed global coordinate system. At each frame, t, this transformation can be described by a 3×3 rotation matrix, R_t, and a translation vector,[7] $\begin{bmatrix} x_t & y_t & z_t \end{bmatrix}^T$. We will assume, for our discussion, that z corresponds to the vertical. When R_t is the identity matrix, this defines the "rest position" which is typically defined by skeleton meta-data that accompanies the joint angles. Without loss of generality, let us assume that in the rest position the subject is axis-aligned such that the dorsoventral axis (from spinal column to belly) aligns with the x axis:

$$\mathbf{u}_t^0 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T,$$

the lateral axis (from left to right side of body) aligns with the y axis:

$$\mathbf{v}_t^0 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T,$$

and the anteroposterior axis (from head to feet) aligns with the negative z axis:

$$\mathbf{w}_t^0 = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T.$$

When the root is rotated (i.e., R_t is not the identity) the body-centred coordinate system is no longer axis aligned. It becomes:

$$\begin{aligned} \mathbf{u}_t &= R_t^T \mathbf{u}_t^0, \\ \mathbf{v}_t &= R_t^T \mathbf{v}_t^0, \\ \mathbf{w}_t &= R_t^T \mathbf{w}_t^0 \end{aligned}$$

6. For joints with a single degree of freedom, the exponential map reduces to an Euler angle and so we do not convert. The orientation of the root does not need to be converted to exponential maps since we build an alternative representation in the following section which requires the orientation to be expressed as a 3×3 rotation matrix.
7. It is also common to represent the transformation by a 4×4 matrix.

where we have assumed a particular convention for the rotation matrix. Note that these axes are simply the rows of the rotation matrix under our chosen convention.

Measuring the angle that the dorsoventral axis makes with the vertical gives us a measure of pitch:

$$\phi_t = \cos^{-1}\left(\frac{\mathbf{u}_t \cdot \mathbf{w}_t^0}{||\mathbf{u}_t||\,||\mathbf{w}_t^0||}\right) = \cos^{-1}\left(\frac{\mathbf{u}_t \cdot \mathbf{w}_t^0}{||\mathbf{u}_t||}\right).$$

Similarly, measuring the angle that the lateral axis makes with the gravitational vertical gives us a measure of roll:

$$\psi_t = \cos^{-1}\left(\frac{\mathbf{v}_t \cdot \mathbf{w}_t^0}{||\mathbf{v}_t||\,||\mathbf{w}_t^0||}\right) = \cos^{-1}\left(\frac{\mathbf{v}_t \cdot \mathbf{w}_t^0}{||\mathbf{v}_t||}\right).$$

Both pitch and roll are invariant to rotation about the vertical and therefore can be thought of as "body-centred" rotations. By projecting $\mathbf{u}_t$ into the ground-plane, this provides a measure of yaw, or rotation about the vertical:

$$\theta_t = \tan^{-1}\left(\frac{u_t^y}{u_t^x}\right)$$

where u_t^x and u_t^y are the first two components of vector $\mathbf{u}_t$. Care should be taken to use the four-quadrant version of $\tan^{-1}$ (often called the `atan2` function). We unwrap θ_t to eliminate discontinuities.

A.4 Conversion to Incremental Changes

We represent the rotation about the vertical, as well as translations in the ground plane by their incremental changes (forward differences) and not their absolute values:

$$\begin{aligned}\dot{\theta}_t &= \theta_{t+1} - \theta_t,\\ \dot{x}_t &= x_{t+1} - x_t,\\ \dot{y}_t &= y_{t+1} - y_t.\end{aligned}$$

For the last frame, we can use the two preceding frames to make a constant-velocity prediction. To achieve translational invariance, we need to express velocity in the ground-plane with respect to body-centred and not global coordinates. We can represent velocity in the ground-plane by its magnitude:

$$\alpha_t = \sqrt{\dot{x}_t^{\,2} + \dot{y}_t^{\,2}}$$

and its angle with respect to the x-axis:

$$\beta_t = \tan^{-1}\left(\frac{\dot{y}_t}{\dot{x}_t}\right).$$

Again we make use of the four-quadrant version of $\tan^{-1}$. The velocity is then expressed with respect to the orientation about the vertical, θ_t, in both a forward and lateral component:

$$\begin{aligned}\dot{\gamma}_t &= \alpha_t \cos(\theta_t - \beta_t),\\ \dot{\xi}_t &= \alpha_t \sin(\theta_t - \beta_t)\end{aligned}$$

where we have used "dot" notation to imply that these quantities are incremental values. Taken collectively, $\begin{bmatrix}\dot{\gamma}_t & \dot{\xi}_t & z_t & \phi_t & \psi_t & \dot{\theta}_t\end{bmatrix}^T$ form our invariant representation of the root. Note that the height, z_t, is untouched.

A.5 Data Normalization

Any joint angle dimensions that have constant value are not modeled and removed from the training data (they are re-inserted before playback or export). Each component of the data is normalized to have zero mean and unit variance.

Appendix B. Approximations

In practice, we make several small modifications to the algorithms for both learning and generation. These rely on several approximations, most of which are chosen based on collective experience of training similar networks. The approximations typically replace sampled values with expected values, to reduce unnecessary noise.

While training a CRBM, we replace $v_{i,t}$ in Equation 10, 11 and 13 by its expected value and we also use the expected value of $v_{i,t}$ when computing the probability of activation of the hidden units (Equation 8). However, to compute each of the K reconstructions of the data (Equation 9), we use stochastically chosen binary values of the hidden units. This prevents the hidden activities from transmitting an unbounded amount of information from the data to the reconstruction (Teh and Hinton, 2001).

While updating the directed visible-to-hidden connections (Equation 12), the symmetric undirected connections (Equation 10), and the hidden biases (Equation 14), we use the stochastically chosen binary values of the hidden units in the first term (under the data), but replace $h_{j,t}$ by its expected value in the second term (under the reconstruction). We take this approach because the reconstruction of the data depends on the binary choices made when selecting hidden state. Thus, when we infer the hiddens from the reconstructed data, the probabilities are highly correlated with the binary hidden states inferred from the data. On the other hand, we stop after K reconstructions, so the binary choice of hiddens from the Kth reconstruction does not correlate with any other terms, and there is no reason to include this extra noise.

The alternating Gibbs sampling used when generating data is similar to the procedure we use to learn a CRBM. So we make similar approximations during generation: using stochastically chosen binary values of the hidden units but the expected values of the reconstructed visible units. As a further step to reduce noise, on the *final iteration* of Gibbs sampling, we use the real-valued probabilities of the hidden units when updating the visible units.

Appendix C. FCRBM Weight Updates

The CD updates for the parameters of the FCRBM have an intuitive form. The gradient with respect to a weight that connects a unit to a factor is the difference of two expectations of products. Each product involves three terms: the activity of the respective unit, and the total input to the factor from

each of the two other sets of units involved in the three-way relationship:

$$\Delta W^{\mathbf{v}}_{if} \propto \sum_t \left(\langle v_{i,t} \sum_j W^{\mathbf{h}}_{jf} h_{j,t} \sum_l W^{\mathbf{z}}_{lf} z_{l,t} \rangle_{\text{data}} - \langle v_{i,t} \sum_j W^{\mathbf{h}}_{jf} h_{j,t} \sum_l W^{\mathbf{z}}_{lf} z_{l,t} \rangle_{\text{recon}} \right),$$

$$\Delta W^{\mathbf{h}}_{jf} \propto \sum_t \left(\langle h_{j,t} \sum_i W^{\mathbf{v}}_{if} v_{i,t} \sum_l W^{\mathbf{z}}_{lf} z_{l,t} \rangle_{\text{data}} - \langle h_{j,t} \sum_i W^{\mathbf{v}}_{if} v_{i,t} \sum_l W^{\mathbf{z}}_{lf} z_{l,t} \rangle_{\text{recon}} \right),$$

$$\Delta W^{\mathbf{z}}_{lf} \propto \sum_t \left(\langle z_{l,t} \sum_i W^{\mathbf{v}}_{if} v_{i,t} \sum_j W^{\mathbf{h}}_{jf} h_{j,t} \rangle_{\text{data}} - \langle z_{l,t} \sum_i W^{\mathbf{v}}_{if} v_{i,t} \sum_j W^{\mathbf{h}}_{jf} h_{j,t} \rangle_{\text{recon}} \right),$$

$$\Delta A^{\mathbf{v}}_{im} \propto \sum_t \left(\langle v_{i,t} \sum_k A^{\mathbf{v}_{<t}}_{km} v_{k,<t} \sum_l A^{\mathbf{z}}_{lm} z_{l,t} \rangle_{\text{data}} - \langle v_{i,t} \sum_k A^{\mathbf{v}_{<t}}_{km} v_{k,<t} \sum_l A^{\mathbf{z}}_{lm} z_{l,t} \rangle_{\text{recon}} \right),$$

$$\Delta A^{\mathbf{v}_{<t}}_{km} \propto \sum_t \left(\langle v_{k,<t} \sum_i A^{\mathbf{v}}_{im} v_{i,t} \sum_l A^{\mathbf{z}}_{lm} z_{l,t} \rangle_{\text{data}} - \langle v_{k,<t} \sum_i A^{\mathbf{v}}_{im} v_{i,t} \sum_l A^{\mathbf{z}}_{lm} z_{l,t} \rangle_{\text{recon}} \right),$$

$$\Delta A^{\mathbf{z}}_{lm} \propto \sum_t \left(\langle z_{l,t} \sum_i A^{\mathbf{v}}_{im} v_{i,t} \sum_k A^{\mathbf{v}_{<t}}_{km} v_{k,<t} \rangle_{\text{data}} - \langle z_{l,t} \sum_i A^{\mathbf{v}}_{im} v_{i,t} \sum_k A^{\mathbf{v}_{<t}}_{km} v_{k,<t} \rangle_{\text{recon}} \right),$$

$$\Delta B^{\mathbf{h}}_{jn} \propto \sum_t \left(\langle h_{j,t} \sum_k B^{\mathbf{v}_{<t}}_{kn} v_{k,<t} \sum_l B^{\mathbf{z}}_{ln} z_{l,t} \rangle_{\text{data}} - \langle h_{j,t} \sum_k B^{\mathbf{v}_{<t}}_{kn} v_{k,<t} \sum_l B^{\mathbf{z}}_{ln} z_{l,t} \rangle_{\text{recon}} \right),$$

$$\Delta B^{\mathbf{v}_{<t}}_{kn} \propto \sum_t \left(\langle v_{k,<t} \sum_j B^{\mathbf{h}}_{jn} h_{j,t} \sum_l B^{\mathbf{z}}_{ln} z_{l,t} \rangle_{\text{data}} - \langle v_{k,<t} \sum_j B^{\mathbf{h}}_{jn} h_{j,t} \sum_l B^{\mathbf{z}}_{ln} z_{l,t} \rangle_{\text{recon}} \right),$$

$$\Delta B^{\mathbf{z}}_{ln} \propto \sum_t \left(\langle z_{l,t} \sum_j B^{\mathbf{h}}_{jn} h_{j,t} \sum_k B^{\mathbf{v}_{<t}}_{kn} v_{k,<t} \rangle_{\text{data}} - \langle z_{l,t} \sum_j B^{\mathbf{h}}_{jn} h_{j,t} \sum_k B^{\mathbf{v}_{<t}}_{kn} v_{k,<t} \rangle_{\text{recon}} \right),$$

$$\Delta a_i \propto \sum_t \left(\langle v_{i,t} \rangle_{\text{data}} - \langle v_{i,t} \rangle_{\text{recon}} \right),$$

$$\Delta b_j \propto \sum_t \left(\langle h_{j,t} \rangle_{\text{data}} - \langle h_{j,t} \rangle_{\text{recon}} \right).$$

References

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

O. Arikan and D. A. Forsyth. Interactive motion generation from examples. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2002)*, pages 483–490. ACM Press, 2002.

O. Arikan, D. A. Forsyth, and J. F. O'Brien. Motion synthesis from annotations. In *Proceedings of the 30th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2003)*, pages 402–408. ACM Press, 2003.

Y. Bengio and O. Delalleau. Justifying and generalizing contrastive divergence. *Neural Computation*, 21(1):1–21, 2008.

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems (NIPS 19): Proceedings of the 2006 Conference*, pages 153–160. MIT Press, 2007.

A. Bissacco. Modeling and learning contact dynamics in human motion. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, pages 421–428. IEEE, 2005.

M. Brand and A. Hertzmann. Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2000)*, pages 183–192. ACM Press, 2000.

M. Carreira-Perpinan and G. E. Hinton. On contrastive divergence learning. In *Proceedings of the 10th International Conference on Artificial Intelligence and Statistics (AISTATS 2005)*, pages 59–66, 2005.

Y. Freund and D. Haussler. Unsupervised learning of distributions of binary vectors using 2-layer networks. In J. Moody, S. H. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems (NIPS 4): Proceedings of the 1991 Conference*, pages 912–919. Morgan-Kaufmann, 1992.

P. V. Gehler, A. D. Holub, and M. Welling. The rate adapting poisson model for information retrieval and object recognition. In *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, pages 337–344. ACM Press, 2006.

Z. Ghahramani. Learning dynamic Bayesian networks. In C. Giles and M. Gori, editors, *Adaptive Processing of Sequences and Data Structures*, pages 168–197. Springer-Verlag, Berlin, 1998.

F. S. Grassia. Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

G. E. Hinton. Learning multiple layers of representation. *Trends in Cognitive Sciences*, 11(10): 428–434, 2007.

G. E. Hinton. *A practical guide to training restricted Boltzmann machines*. Technical Report UTML TR 2010-000, University of Toronto, 2010.

G. E. Hinton and A. D. Brown. Spiking Boltzmann machines. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems (NIPS 12): Proceedings of the 1999 Conference*, pages 122–128. MIT Press, 2000.

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

G. E. Hinton, P. Dayan, B. J. Frey, and R. Neal. The wake-sleep algorithm for self-organizing neural networks. *Science*, 268:1158–1161, 1995.

G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

E. Hsu, K. Pulli, and J. Popović. Style translation for human motion. In *Proceedings of the 32nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2005)*, pages 1082–1089. ACM Press, 2005.

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005. ISSN 1532-4435.

L. Kovar and M. Gleicher. Automated extraction and parameterization of motions in large data sets. In *Proceedings of the 31st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2004)*, pages 559–568. ACM Press, 2004.

L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2002)*, pages 473–482. ACM Press, 2002.

H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, pages 473–480. ACM Press, 2007.

N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems (NIPS 16): Proceedings of the 2003 Conference*, pages 329–326. MIT Press, 2004.

N. D. Lawrence. The Gaussian process latent variable model. Technical Report CS-06-05, University of Sheffield, 2006.

N. D. Lawrence. Learning for larger datasets with the Gaussian process latent variable model. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, pages 243–250, 2007.

N. D. Lawrence and A. J. Moore. Hierarchical Gaussian process latent variable models. In Z. Ghahramani, editor, *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, pages 481–488. ACM Press, 2007.

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.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area V2. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS 20): Proceedings of the 2007 Conference*, pages 873–880. MIT Press, 2008.

J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive control of avatars animated with human motion data. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2002)*, pages 491–500. ACM Press, 2002.

Y. Li, T. Wang, and H.-Y. Shum. Motion texture: A two-level statistical model for character motion synthesis. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2002)*, pages 465–472. ACM Press, 2002.

C. K. Liu, A. Hertzmann, and Z. Popovic. Learning physics-based motion style with nonlinear inverse optimization. In *Proceedings of the 32nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2005)*, pages 1071–1081. ACM Press, 2005.

K. Matsuoka. Noise injection into inputs in back-propagation learning. *IEEE Trans. on Systems, Man, and Cybernetics*, 22(3):436–440, 1992.

R. Memisevic and G. E. Hinton. Unsupervised learning of image transformations. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, 2007.

R. Memisevic and G. E. Hinton. Learning to represent spatial transformations with factored higher-order boltzmann machines. *Neural Computation*, 22:1473–1492, 2010.

A. Mnih and G. E. Hinton. Three new graphical models for statistical language modelling. In Z. Ghahramani, editor, *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, pages 641–648. ACM Press, 2007.

T. Mukai and S. Kuriyama. Geostatistical motion interpolation. In *Proceedings of the 32nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2005)*, pages 1062–1070. ACM Press, 2005.

K. P. Murphy. *Dynamic bayesian networks : representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

V. Nair and G. E. Hinton. 3-d object recognition with deep belief nets. In *Advances in Neural Information Processing Systems* **22**, pages 1339–1347. MIT Press, Cambridge, MA, 2009.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23):3311–3325, 1997.

S. Osindero and G. E. Hinton. Modeling image patches with a directed hierarchy of Markov random fields. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS 20): Proceedings of the 2007 Conference*, pages 1121–1128. MIT Press, 2008.

S. I. Park, H. J. Shin, and S. Y. Shin. On-line locomotion generation based on motion blending. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 02)*, pages 105–111. ACM Press, 2002.

V. Pavlovic, J. M. Rehg, and J. MacCormick. Learning switching linear models of human motion. In *Advances in Neural Information Processing Systems (NIPS 13): Proceedings of the 2000 Conference*, pages 981–987. MIT Press, 2001.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference*. Morgan Kaufmann, Santa Mateo, CA, USA, September 1988.

K. Pullen and C. Bregler. Motion capture assisted animation: Texturing and synthesis. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2002)*, pages 501–508. ACM Press, 2002.

M. Ranzato, C. S. Poultney, S. Chopra, and Y. LeCun. Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems (NIPS 19): Proceedings of the 2006 Conference*, pages 1137–1144. MIT Press, 2006.

M. Ranzato, Y. Boureau, S. Chopra, and Y. LeCun. A unified energy-based framework for unsupervised learning. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, 2007.

M. Ranzato, Y. Boureau, and Y. LeCun. Sparse feature learning for deep belief networks. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS 20): Proceedings of the 2007 Conference*. MIT Press, 2008.

C. Rose, M. F. Cohen, and B. Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications*, 18(5):32–40, 1998.

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, pages 872–879. ACM Press, 2008.

R. Salakhutdinov, A. Mnih, and G. E. Hinton. Restricted Boltzmann machines for collaborative filtering. In Z. Ghahramani, editor, *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, pages 791–798. ACM Press, 2007.

P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rumelhart, J. L. McClelland, et al., editors, *Parallel Distributed Processing: Volume 1: Foundations*, pages 194–281. MIT Press, Cambridge, MA, 1986.

I. Sutskever and G. E. Hinton. Learning multilevel distributed representations for high-dimensional sequences. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, 2007.

I. Sutskever and G. E. Hinton. Deep narrow sigmoid belief networks are universal approximators. *Neural Computation*, 20(11):2629–2636, 2008.

I. Sutskever, G. E. Hinton, and G. W. Taylor. The recurrent temporal restricted Boltzmann machine. In *Advances in Neural Information Processing Systems (NIPS 21): Proceedings of the 2008 Conference*, volume 21. MIT Press, 2009.

L. Tanco and A. Hilton. Realistic synthesis of novel human movements from a database of motion capture examples. In *Proceedings of the Workshop on Human Motion (HUMO '00)*, pages 137–142. IEEE Computer Society, 2000.

G. Taylor and G. Hinton. Factored conditional restricted Boltzmann machines for modeling motion style. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, pages 1025–1032, 2009.

G. W. Taylor, G. E. Hinton, and S. T. Roweis. Modeling human motion using binary latent variables. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems (NIPS 19): Proceedings of the 2006 Conference*, pages 1345–1352. MIT Press, 2007.

Y. W. Teh and G. E. Hinton. Rate-coded restricted Boltzmann machines for face recognition. In *Advances in Neural Information Processing Systems (NIPS 13): Proceedings of the 2000 Conference*. MIT Press, 2001.

J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, pages 1064–1071. ACM Press, 2008.

T. Tieleman and G. E. Hinton. Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*. ACM Press, 2009.

L. Torresani, P. Hackney, and C. Bregler. Learning motion style synthesis from perceptual observations. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems (NIPS 19): Proceedings of the 2006 Conference*, pages 1393–1400. MIT Press, 2007.

R. Urtasun, P. Glardon, R. Boulic, D. Thalmann, and P. Fua. Style-based motion synthesis. *Computer Graphics Forum*, 23(4):1–14, 2004.

R. Urtasun, D. Fleet, and P. F. P. 3D people tracking with gaussian process dynamical models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, pages 238–245. IEEE, 2006.

R. Urtasun, D. J. Fleet, A. Geiger, J. Popović, T. Darrell, and N. D. Lawrence. Topologically-constrained latent variable models. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, pages 1080–1087. ACM Press, 2008.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Multifactor Gaussian process models for style-content separation. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, pages 975–982. ACM Press, 2007.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

M. Welling, M. Rosen-Zvi, and G. E. Hinton. Exponential family harmoniums with an application to information retrieval. In *Advances in Neural Information Processing Systems (NIPS 17): Proceedings of the 2004 Conference*, pages 1481–1488. MIT Press, 2005.

Journal of Machine Learning Research 12 (2011) 1069-1109 Submitted 6/10; Revised 2/11; Published 3/11

Differentially Private Empirical Risk Minimization

Kamalika Chaudhuri KCHAUDHURI@UCSD.EDU
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093, USA

Claire Monteleoni CMONTEL@CCLS.COLUMBIA.EDU
Center for Computational Learning Systems
Columbia University
New York, NY 10115, USA

Anand D. Sarwate ASARWATE@UCSD.EDU
Information Theory and Applications Center
University of California, San Diego
La Jolla, CA 92093-0447, USA

Editor: Nicolas Vayatis

Abstract

Privacy-preserving machine learning algorithms are crucial for the increasingly common setting in which personal data, such as medical or financial records, are analyzed. We provide general techniques to produce privacy-preserving approximations of classifiers learned via (regularized) empirical risk minimization (ERM). These algorithms are private under the ε-*differential privacy* definition due to Dwork et al. (2006). First we apply the output perturbation ideas of Dwork et al. (2006), to ERM classification. Then we propose a new method, *objective perturbation*, for privacy-preserving machine learning algorithm design. This method entails perturbing the objective function before optimizing over classifiers. If the loss and regularizer satisfy certain convexity and differentiability criteria, we prove theoretical results showing that our algorithms preserve privacy, and provide generalization bounds for linear and nonlinear kernels. We further present a privacy-preserving technique for tuning the parameters in general machine learning algorithms, thereby providing end-to-end privacy guarantees for the training process. We apply these results to produce privacy-preserving analogues of regularized logistic regression and support vector machines. We obtain encouraging results from evaluating their performance on real demographic and benchmark data sets. Our results show that both theoretically and empirically, objective perturbation is superior to the previous state-of-the-art, output perturbation, in managing the inherent tradeoff between privacy and learning performance.

Keywords: privacy, classification, optimization, empirical risk minimization, support vector machines, logistic regression

1. Introduction

Privacy has become a growing concern, due to the massive increase in personal information stored in electronic databases, such as medical records, financial records, web search histories, and social network data. Machine learning can be employed to discover novel population-wide patterns, however the results of such algorithms may reveal certain individuals' sensitive information, thereby

©2011 Kamalika Chaudhuri, Claire Monteleoni and Anand D. Sarwate.

violating their privacy. Thus, an emerging challenge for machine learning is how to learn from data sets that contain sensitive personal information.

At the first glance, it may appear that simple anonymization of private information is enough to preserve privacy. However, this is often not the case; even if obvious identifiers, such as names and addresses, are removed from the data, the remaining fields can still form unique "signatures" that can help re-identify individuals. Such attacks have been demonstrated by various works, and are possible in many realistic settings, such as when an adversary has side information (Sweeney, 1997; Narayanan and Shmatikov, 2008; Ganta et al., 2008), and when the data has structural properties (Backstrom et al., 2007), among others. Moreover, even releasing statistics on a sensitive data set may not be sufficient to preserve privacy, as illustrated on genetic data (Homer et al., 2008; Wang et al., 2009). Thus, there is a great need for designing machine learning algorithms that also preserve the privacy of individuals in the data sets on which they train and operate.

In this paper we focus on the problem of classification, one of the fundamental problems of machine learning, when the training data consists of sensitive information of individuals. Our work addresses the empirical risk minimization (ERM) framework for classification, in which a classifier is chosen by minimizing the average over the training data of the prediction loss (with respect to the label) of the classifier in predicting each training data point. In this work, we focus on regularized ERM in which there is an additional term in the optimization, called the regularizer, which penalizes the complexity of the classifier with respect to some metric. Regularized ERM methods are widely used in practice, for example in logistic regression and support vector machines (SVMs), and many also have theoretical justification in the form of generalization error bounds with respect to independently, identically distributed (i.i.d.) data (see Vapnik, 1998 for further details).

For our privacy measure, we use a definition due to Dwork et al. (2006b), who have proposed a measure of quantifying the privacy-risk associated with computing functions of sensitive data. Their ε-*differential privacy* model is a strong, cryptographically-motivated definition of privacy that has recently received a significant amount of research attention for its robustness to known attacks, such as those involving side information (Ganta et al., 2008). Algorithms satisfying ε-differential privacy are randomized; the output is a random variable whose distribution is conditioned on the data set. A statistical procedure satisfies ε-differential privacy if changing a single data point does not shift the output distribution by too much. Therefore, from looking at the output of the algorithm, it is difficult to infer the value of any particular data point.

In this paper, we develop methods for approximating ERM while guaranteeing ε-differential privacy. Our results hold for loss functions and regularizers satisfying certain differentiability and convexity conditions. An important aspect of our work is that we develop methods for *end-to-end privacy*; each step in the learning process can cause additional risk of privacy violation, and we provide algorithms with quantifiable privacy guarantees for training as well as parameter tuning. For training, we provide two privacy-preserving approximations to ERM. The first is *output perturbation*, based on the *sensitivity method* proposed by Dwork et al. (2006b). In this method noise is added to the output of the standard ERM algorithm. The second method is novel, and involves adding noise to the regularized ERM objective function prior to minimizing. We call this second method *objective perturbation*. We show theoretical bounds for both procedures; the theoretical performance of objective perturbation is superior to that of output perturbation for most problems. However, for our results to hold we require that the regularizer be strongly convex (ruling L_1 regularizers) and additional constraints on the loss function and its derivatives. In practice, these additional

constraints do not affect the performance of the resulting classifier; we validate our theoretical results on data sets from the UCI repository.

In practice, parameters in learning algorithms are chosen via a holdout data set. In the context of privacy, we must guarantee the privacy of the holdout data as well. We exploit results from the theory of differential privacy to develop a privacy-preserving parameter tuning algorithm, and demonstrate its use in practice. Together with our training algorithms, this parameter tuning algorithm guarantees privacy to all data used in the learning process.

Guaranteeing privacy incurs a cost in performance; because the algorithms must cause some uncertainty in the output, they increase the loss of the output predictor. Because the ϵ-differential privacy model requires robustness against all data sets, we make no assumptions on the underlying data for the purposes of making privacy guarantees. However, to prove the impact of privacy constraints on the generalization error, we assume the data is i.i.d. according to a fixed but unknown distribution, as is standard in the machine learning literature. Although many of our results hold for ERM in general, we provide specific results for classification using logistic regression and support vector machines. Some of the former results were reported in Chaudhuri and Monteleoni (2008); here we generalize them to ERM and extend the results to kernel methods, and provide experiments on real data sets.

More specifically, the contributions of this paper are as follows:

- We derive a computationally efficient algorithm for ERM classification, based on the sensitivity method due to Dwork et al. (2006b). We analyze the accuracy of this algorithm, and provide an upper bound on the number of training samples required by this algorithm to achieve a fixed generalization error.

- We provide a general technique, *objective perturbation*, for providing computationally efficient, differentially private approximations to regularized ERM algorithms. This extends the work of Chaudhuri and Monteleoni (2008), which follows as a special case, and corrects an error in the arguments made there. We apply the general results on the sensitivity method and objective perturbation to logistic regression and support vector machine classifiers. In addition to privacy guarantees, we also provide generalization bounds for this algorithm.

- For kernel methods with nonlinear kernel functions, the optimal classifier is a linear combination of kernel functions centered at the training points. This form is inherently non-private because it reveals the training data. We adapt a random projection method due to Rahimi and Recht (2007, 2008b), to develop privacy-preserving kernel-ERM algorithms. We provide theoretical results on generalization performance.

- Because the holdout data is used in the process of training and releasing a classifier, we provide a privacy-preserving parameter tuning algorithm based on a randomized selection procedure (McSherry and Talwar, 2007) applicable to general machine learning algorithms. This guarantees end-to-end privacy during the learning procedure.

- We validate our results using experiments on two data sets from the UCI Machine Learning repositories (Asuncion and Newman, 2007) and KDDCup (Hettich and Bay, 1999). Our results show that objective perturbation is generally superior to output perturbation. We also demonstrate the impact of end-to-end privacy on generalization error.

1.1 Related Work

There has been a significant amount of literature on the ineffectiveness of simple anonymization procedures. For example, Narayanan and Shmatikov (2008) show that a small amount of auxiliary information (knowledge of a few movie-ratings, and approximate dates) is sufficient for an adversary to re-identify an individual in the Netflix data set, which consists of anonymized data about Netflix users and their movie ratings. The same phenomenon has been observed in other kinds of data, such as social network graphs (Backstrom et al., 2007), search query logs (Jones et al., 2007) and others. Releasing statistics computed on sensitive data can also be problematic; for example, Wang et al. (2009) show that releasing R^2-values computed on high-dimensional genetic data can lead to privacy breaches by an adversary who is armed with a small amount of auxiliary information.

There has also been a significant amount of work on privacy-preserving data mining (Agrawal and Srikant, 2000; Evfimievski et al., 2003; Sweeney, 2002; Machanavajjhala et al., 2006), spanning several communities, that uses privacy models other than differential privacy. Many of the models used have been shown to be susceptible to *composition attacks*, attacks in which the adversary has some reasonable amount of prior knowledge (Ganta et al., 2008). Other work (Mangasarian et al., 2008) considers the problem of privacy-preserving SVM classification when separate agents have to share private data, and provides a solution that uses random kernels, but does provide any formal privacy guarantee.

An alternative line of privacy work is in the secure multiparty computation setting due to Yao (1982), where the sensitive data is split across multiple hostile databases, and the goal is to compute a function on the union of these databases. Zhan and Matwin (2007) and Laur et al. (2006) consider computing privacy-preserving SVMs in this setting, and their goal is to design a distributed protocol to learn a classifier. This is in contrast with our work, which deals with a setting where the algorithm has access to the entire data set.

Differential privacy, the formal privacy definition used in our paper, was proposed by the seminal work of Dwork et al. (2006b), and has been used since in numerous works on privacy (Chaudhuri and Mishra, 2006; McSherry and Talwar, 2007; Nissim et al., 2007; Barak et al., 2007; Chaudhuri and Monteleoni, 2008; Machanavajjhala et al., 2008). Unlike many other privacy definitions, such as those mentioned above, differential privacy has been shown to be resistant to composition attacks (attacks involving side-information) (Ganta et al., 2008). Some follow-up work on differential privacy includes work on differentially-private combinatorial optimization, due to Gupta et al. (2010), and differentially-private contingency tables, due to Barak et al. (2007) and Kasivishwanathan et al. (2010). Wasserman and Zhou (2010) provide a more statistical view of differential privacy, and Zhou et al. (2009) provide a technique of generating synthetic data using compression via random linear or affine transformations.

Previous literature has also considered learning with differential privacy. One of the first such works is Kasiviswanathan et al. (2008), which presents a general, although computationally inefficient, method for PAC-learning finite concept classes. Blum et al. (2008) presents a method for releasing a database in a differentially-private manner, so that certain fixed classes of queries can be answered accurately, provided the class of queries has a bounded VC-dimension. Their methods can also be used to learn classifiers with a fixed VC-dimension (Kasiviswanathan et al., 2008) but the resulting algorithm is also computationally inefficient. Some sample complexity lower bounds in this setting have been provided by Beimel et al. (2010). In addition, Dwork and Lei (2009) explore a connection between differential privacy and robust statistics, and provide an algorithm

for privacy-preserving regression using ideas from robust statistics. Their algorithm also requires a running time which is exponential in the data dimension, which is computationally inefficient.

This work builds on our preliminary work in Chaudhuri and Monteleoni (2008). We first show how to extend the sensitivity method, a form of *output perturbation*, due to Dwork et al. (2006b), to classification algorithms. In general, output perturbation methods alter the output of the function computed on the database, before releasing it; in particular the sensitivity method makes an algorithm differentially private by adding noise to its output. In the classification setting, the noise protects the privacy of the training data, but increases the prediction error of the classifier. Recently, independent work by Rubinstein et al. (2009) has reported an extension of the sensitivity method to linear and kernel SVMs. Their utility analysis differs from ours, and thus the analogous generalization bounds are not comparable. Because Rubinstein et al. use techniques from algorithmic stability, their utility bounds compare the private and non-private classifiers using the same value for the regularization parameter. In contrast, our approach takes into account how the value of the regularization parameter might change due to privacy constraints. In contrast, we propose the *objective perturbation* method, in which noise is added to the *objective function* before optimizing over the space classifiers. Both the sensitivity method and objective perturbation result in computationally efficient algorithms for our specific case. In general, our theoretical bounds on sample requirement are incomparable with the bounds of Kasiviswanathan et al. (2008) because of the difference between their setting and ours.

Our approach to privacy-preserving tuning uses the exponential mechanism of McSherry and Talwar (2007) by training classifiers with different parameters on disjoint subsets of the data and then randomizing the selection of which classifier to release. This bears a superficial resemblance to the sample-and-aggregate (Nissim et al., 2007) and V-fold cross-validation, but only in the sense that only a part of the data is used to train the classifier. One drawback is that our approach requires significantly more data in practice. Other approaches to selecting the regularization parameter could benefit from a more careful analysis of the regularization parameter, as in Hastie et al. (2004).

2. Model

We will use $\|\mathbf{x}\|$, $\|\mathbf{x}\|_\infty$, and $\|\mathbf{x}\|_{\mathcal{H}}$ to denote the ℓ_2-norm, ℓ_∞-norm, and norm in a Hilbert space $\mathcal{H}$, respectively. For an integer n we will use $[n]$ to denote the set $\{1,2,\ldots,n\}$. Vectors will typically be written in boldface and sets in calligraphic type. For a matrix A, we will use the notation $\|A\|_2$ to denote the L_2 norm of A.

2.1 Empirical Risk Minimization

In this paper we develop privacy-preserving algorithms for *regularized empirical risk minimization*, a special case of which is learning a classifier from labeled examples. We will phrase our problem in terms of classification and indicate when more general results hold. Our algorithms take as input *training data* $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y} : i = 1,2,\ldots,n\}$ of n data-label pairs. In the case of binary classification the data space $\mathcal{X} = \mathbb{R}^d$ and the label set $\mathcal{Y} = \{-1,+1\}$. We will assume throughout that $\mathcal{X}$ is the unit ball so that $\|\mathbf{x}_i\|_2 \leq 1$.

We would like to produce a *predictor* $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$. We measure the quality of our predictor on the training data via a nonnegative *loss function* $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$.

In regularized empirical risk minimization (ERM), we choose a predictor $\mathbf{f}$ that minimizes the regularized empirical loss:

$$J(\mathbf{f}, \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{f}(\mathbf{x}_i), \mathbf{y}_i) + \Lambda N(\mathbf{f}). \tag{1}$$

This minimization is performed over $\mathbf{f}$ in an hypothesis class $\mathcal{H}$. The regularizer $N(\cdot)$ prevents over-fitting. For the first part of this paper we will restrict our attention to linear predictors and with some abuse of notation we will write $\mathbf{f}(\mathbf{x}) = \mathbf{f}^T\mathbf{x}$.

2.2 Assumptions on Loss and Regularizer

The conditions under which we can prove results on privacy and generalization error depend on analytic properties of the loss and regularizer. In particular, we will require certain forms of convexity (see Rockafellar and Wets, 1998).

Definition 1 *A function $H(\mathbf{f})$ over $\mathbf{f} \in \mathbb{R}^d$ is said to be* strictly convex *if for all $\alpha \in (0,1)$, $\mathbf{f}$, and $\mathbf{g}$,*

$$H(\alpha\mathbf{f} + (1-\alpha)\mathbf{g}) < \alpha H(\mathbf{f}) + (1-\alpha)H(\mathbf{g}).$$

It is said to be λ-strongly convex *if for all $\alpha \in (0,1)$, $\mathbf{f}$, and $\mathbf{g}$,*

$$H(\alpha\mathbf{f} + (1-\alpha)\mathbf{g}) \leq \alpha H(\mathbf{f}) + (1-\alpha)H(\mathbf{g}) - \frac{1}{2}\lambda\alpha(1-\alpha)\left\|\mathbf{f}-\mathbf{g}\right\|_2^2.$$

A strictly convex function has a unique minimum (Boyd and Vandenberghe, 2004). Strong convexity plays a role in guaranteeing our privacy and generalization requirements. For our privacy results to hold we will also require that the regularizer $N(\cdot)$ and loss function $\ell(\cdot,\cdot)$ be differentiable functions of $\mathbf{f}$. This excludes certain classes of regularizers, such as the ℓ_1-norm regularizer $N(\mathbf{f}) = \left\|\mathbf{f}\right\|_1$, and classes of loss functions such as the hinge loss $\ell_{\text{SVM}}(\mathbf{f}^T\mathbf{x}, y) = (1 - y\mathbf{f}^T\mathbf{x})^+$. In some cases we can prove privacy guarantees for approximations to these non-differentiable functions.

2.3 Privacy Model

We are interested in producing a classifier in a manner that preserves the privacy of individual entries of the data set $\mathcal{D}$ that is used in training the classifier. The notion of privacy we use is the ε-*differential privacy model*, developed by Dwork et al. (2006b) (see also Dwork (2006)), which defines a notion of privacy for a randomized algorithm $\mathcal{A}(\mathcal{D})$. Suppose $\mathcal{A}(\mathcal{D})$ produces a classifier, and let $\mathcal{D}'$ be another data set that differs from $\mathcal{D}$ in one entry (which we assume is the private value of one person). That is, $\mathcal{D}'$ and $\mathcal{D}$ have $n-1$ points $(\mathbf{x}_i, y_i)$ in common. The algorithm $\mathcal{A}$ provides differential privacy if for any set $\mathcal{S}$, the likelihood that $\mathcal{A}(\mathcal{D}) \in \mathcal{S}$ is close to the likelihood $\mathcal{A}(\mathcal{D}') \in \mathcal{S}$, (where the likelihood is over the randomness in the algorithm). That is, any single entry of the data set does not affect the output distribution of the algorithm by much; dually, this means that an adversary, who knows all but one entry of the data set, cannot gain much additional information about the last entry by observing the output of the algorithm.

The following definition of differential privacy is due to Dwork et al. (2006b), paraphrased from Wasserman and Zhou (2010).

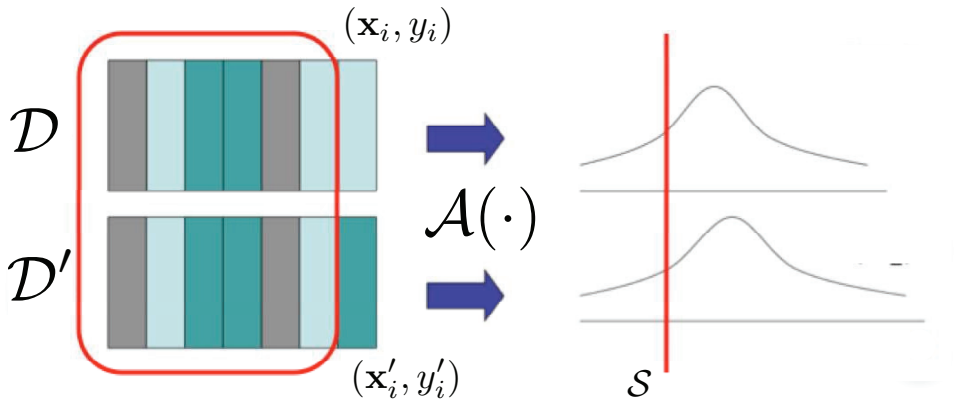


Figure 1: An algorithm which is differentially private. When data sets which are identical except for a single entry are input to the algorithm $\mathcal{A}$, the two distributions on the algorithm's output are close. For a fixed measurable $\mathcal{S}$ the ratio of the measures (or densities) should be bounded.

Definition 2 *An algorithm $\mathcal{A}(\mathcal{B})$ taking values in a set $\mathcal{T}$ provides ε_p-differential privacy if*

$$\sup_{\mathcal{S}} \sup_{\mathcal{D},\mathcal{D}'} \frac{\mu(\mathcal{S} \mid \mathcal{B} = \mathcal{D})}{\mu(\mathcal{S} \mid \mathcal{B} = \mathcal{D}')} \leq e^{\varepsilon_p}, \tag{2}$$

where the first supremum is over all measurable $\mathcal{S} \subseteq \mathcal{T}$, the second is over all data sets $\mathcal{D}$ and $\mathcal{D}'$ differing in a single entry, and $\mu(\cdot|\mathcal{B})$ is the conditional distribution (measure) on $\mathcal{T}$ induced by the output $\mathcal{A}(\mathcal{B})$ given a data set $\mathcal{B}$. The ratio is interpreted to be 1 whenever the numerator and denominator are both 0.

Note that if $\mathcal{S}$ is a set of measure 0 under the conditional measures induced by $\mathcal{D}$ and $\mathcal{D}'$, the ratio is automatically 1. A more measure-theoretic definition is given in Zhou et al. (2009). An illustration of the definition is given in Figure 1.

The following form of the definition is due to Dwork et al. (2006a).

Definition 3 *An algorithm $\mathcal{A}$ provides ε_p-differential privacy if for any two data sets $\mathcal{D}$ and $\mathcal{D}'$ that differ in a single entry and for any set $\mathcal{S}$,*

$$\exp(-\varepsilon_p)\mathbb{P}(\mathcal{A}(\mathcal{D}') \in \mathcal{S}) \leq \mathbb{P}(\mathcal{A}(\mathcal{D}) \in \mathcal{S}) \leq \exp(\varepsilon_p)\mathbb{P}(\mathcal{A}(\mathcal{D}') \in \mathcal{S}), \tag{3}$$

where $\mathcal{A}(\mathcal{D})$ (resp. $\mathcal{A}(\mathcal{D}')$) is the output of $\mathcal{A}$ on input $\mathcal{D}$ (resp. $\mathcal{D}'$).

We observe that an algorithm $\mathcal{A}$ that satisfies Equation 2 also satisfies Equation 3, and as a result, Definition 2 is stronger than Definition 3.

From this definition, it is clear that the $\mathcal{A}(\mathcal{D})$ that outputs the minimizer of the ERM objective (1) does not provide ε_p-differential privacy for any ε_p. This is because an ERM solution is a linear combination of some selected training samples "near" the decision boundary. If $\mathcal{D}$ and $\mathcal{D}'$ differ in one of these samples, then the classifier will change completely, making the likelihood ratio in (3)

infinite. Regularization helps by penalizing the L_2 norm of the change, but does not account how the direction of the minimizer is sensitive to changes in the data.

Dwork et al. (2006b) also provide a standard recipe for computing privacy-preserving approximations to functions by adding noise with a particular distribution to the output of the function. We call this recipe the *sensitivity method*. Let $g : (\mathbb{R}^m)^n \to \mathbb{R}$ be a scalar function of $z_1, \ldots, z_n$, where $z_i \in \mathbb{R}^m$ corresponds to the private value of individual i; then the sensitivity of g is defined as follows.

Definition 4 *The sensitivity of a function $g : (\mathbb{R}^m)^n \to \mathbb{R}$ is maximum difference between the values of the function when one input changes. More formally, the sensitivity $S(g)$ of g is defined as:*

$$S(g) = \max_i \max_{z_1,\ldots,z_n,z_i'} \left| g(z_1,\ldots,z_{i-1},z_i,z_{i+1},\ldots,z_n) - g(z_1,\ldots,z_{i-1},z_i',z_{i+1},\ldots,z_n) \right|.$$

To compute a function g on a data set $\mathcal{D} = \{z_1, \ldots, z_n\}$, the sensitivity method outputs $g(z_1, \ldots, z_n) + \eta$, where η is a random variable drawn according to the Laplace distribution, with mean 0 and standard deviation $\frac{S(g)}{\varepsilon_p}$. It is shown in Dwork et al. (2006b) that such a procedure is ε_p-differentially private.

3. Privacy-preserving ERM

Here we describe two approaches for creating privacy-preserving algorithms from (1).

3.1 Output Perturbation: The Sensitivity Method

Algorithm 1 is derived from the *sensitivity method* of Dwork et al. (2006b), a general method for generating a privacy-preserving approximation to any function $A(\cdot)$. In this section the norm $\|\cdot\|$ is the L_2-norm unless otherwise specified. For the function $A(\mathcal{D}) = \operatorname{argmin} J(\mathbf{f}, \mathcal{D})$, Algorithm 1 outputs a vector $A(\mathcal{D}) + \mathbf{b}$, where $\mathbf{b}$ is random noise with density

$$\nu(\mathbf{b}) = \frac{1}{\alpha} e^{-\beta\|\mathbf{b}\|}, \tag{4}$$

where α is a normalizing constant. The parameter β is a function of ε_p, and the *L_2-sensitivity* of $A(\cdot)$, which is defined as follows.

Definition 5 *The L_2-sensitivity of a vector-valued function is defined as the maximum change in the L_2 norm of the value of the function when one input changes. More formally,*

$$S(A) = \max_i \max_{z_1,\ldots,z_n,z_i'} \left\| A(z_1,\ldots,z_i,\ldots) - A(z_1,\ldots,z_i',\ldots) \right\|.$$

The interested reader is referred to Dwork et al. (2006b) for further details. Adding noise to the output of $A(\cdot)$ has the effect of masking the effect of any particular data point. However, in some applications the sensitivity of the minimizer $\operatorname{argmin} J(\mathbf{f}, \mathcal{D})$ may be quite high, which would require the sensitivity method to add noise with high variance.

Algorithm 1 ERM with output perturbation (sensitivity)

Inputs: Data $\mathcal{D} = \{z_i\}$, parameters ε_p, Λ.
Output: Approximate minimizer $\mathbf{f}_{\text{priv}}$.
Draw a vector $\mathbf{b}$ according to (4) with $\beta = \frac{n\Lambda\varepsilon_p}{2}$.
Compute $\mathbf{f}_{\text{priv}} = \text{argmin} J(\mathbf{f}, \mathcal{D}) + \mathbf{b}$.

3.2 Objective Perturbation

A different approach, first proposed by Chaudhuri and Monteleoni (2008), is to add noise to the objective function itself and then produce the minimizer of the perturbed objective. That is, we can minimize

$$J_{\text{priv}}(\mathbf{f}, \mathcal{D}) = J(\mathbf{f}, \mathcal{D}) + \frac{1}{n}\mathbf{b}^T\mathbf{f},$$

where $\mathbf{b}$ has density given by (4), with $\beta = \varepsilon_p$. Note that the privacy parameter here does not depend on the sensitivity of the of the classification algorithm.

Algorithm 2 ERM with objective perturbation

Inputs: Data $\mathcal{D} = \{z_i\}$, parameters ε_p, Λ, c.
Output: Approximate minimizer $\mathbf{f}_{\text{priv}}$.
Let $\varepsilon'_p = \varepsilon_p - \log(1 + \frac{2c}{n\Lambda} + \frac{c^2}{n^2\Lambda^2})$.
If $\varepsilon'_p > 0$, then $\Delta = 0$, else $\Delta = \frac{c}{n(e^{\varepsilon_p/4}-1)} - \Lambda$, and $\varepsilon'_p = \varepsilon_p/2$.
Draw a vector $\mathbf{b}$ according to (4) with $\beta = \varepsilon'_p/2$.
Compute $\mathbf{f}_{\text{priv}} = \text{argmin} J_{\text{priv}}(\mathbf{f}, \mathcal{D}) + \frac{1}{2}\Delta||\mathbf{f}||^2$.

The algorithm requires a certain slack, $\log(1 + \frac{2c}{n\Lambda} + \frac{c^2}{n^2\Lambda^2})$, in the privacy parameter. This is due to additional factors in bounding the ratio of the densities. The "If" statement in the algorithm is from having to consider two cases in the proof of Theorem 9, which shows that the algorithm is differentially private.

3.3 Privacy Guarantees

In this section, we establish the conditions under which Algorithms 1 and 2 provide ε_p-differential privacy. First, we establish guarantees for Algorithm 1.

3.3.1 PRIVACY GUARANTEES FOR OUTPUT PERTURBATION

Theorem 6 *If $N(\cdot)$ is differentiable, and 1-strongly convex, and ℓ is convex and differentiable, with $|\ell'(z)| \leq 1$ for all z, then, Algorithm 1 provides ε_p-differential privacy.*

The proof of Theorem 6 follows from Corollary 8, and Dwork et al. (2006b). The proof is provided here for completeness.

Proof From Corollary 8, if the conditions on $N(\cdot)$ and ℓ hold, then the L_2-sensivity of ERM with regularization parameter Λ is at most $\frac{2}{n\Lambda}$. We observe that when we pick $||\mathbf{b}||$ from the distribution in Algorithm 1, for a specific vector $\mathbf{b_0} \in \mathbb{R}^d$, the density at $\mathbf{b_0}$ is proportional to $e^{-\frac{n\Lambda\varepsilon_p}{2}||\mathbf{b_0}||}$. Let $\mathcal{D}$

and $\mathcal{D}'$ be any two data sets that differ in the value of one individual. Then, for any $\mathbf{f}$,

$$\frac{g(\mathbf{f}|\mathcal{D})}{g(\mathbf{f}|\mathcal{D}')} = \frac{\nu(\mathbf{b}_1)}{\nu(\mathbf{b}_2)} = e^{-\frac{n\Lambda\varepsilon_p}{2}(||\mathbf{b}_1||-||\mathbf{b}_2||)},$$

where $\mathbf{b}_1$ and $\mathbf{b}_2$ are the corresponding noise vectors chosen in Step 1 of Algorithm 1, and $g(\mathbf{f}|\mathcal{D})$ ($g(\mathbf{f}|\mathcal{D}')$ respectively) is the density of the output of Algorithm 1 at $\mathbf{f}$, when the input is $\mathcal{D}$ ($\mathcal{D}'$ respectively). If $\mathbf{f}_1$ and $\mathbf{f}_2$ are the solutions respectively to non-private regularized ERM when the input is $\mathcal{D}$ and $\mathcal{D}'$, then, $\mathbf{b}_2 - \mathbf{b}_1 = \mathbf{f}_2 - \mathbf{f}_1$. From Corollary 8, and using a triangle inequality,

$$||\mathbf{b}_1|| - ||\mathbf{b}_2|| \leq ||\mathbf{b}_1 - \mathbf{b}_2|| = ||\mathbf{f}_2 - \mathbf{f}_1|| \leq \frac{2}{n\Lambda}.$$

Moreover, by symmetry, the density of the directions of $\mathbf{b}_1$ and $\mathbf{b}_2$ are uniform. Therefore, by construction, $\frac{\nu(\mathbf{b}_1)}{\nu(\mathbf{b}_2)} \leq e^{\varepsilon_p}$. The theorem follows. ■

The main ingredient of the proof of Theorem 6 is a result about the sensitivity of regularized ERM, which is provided below.

Lemma 7 *Let $G(\mathbf{f})$ and $g(\mathbf{f})$ be two vector-valued functions, which are continuous, and differentiable at all points. Moreover, let $G(\mathbf{f})$ and $G(\mathbf{f}) + g(\mathbf{f})$ be λ-strongly convex. If $\mathbf{f}_1 = \operatorname{argmin}_{\mathbf{f}} G(\mathbf{f})$ and $\mathbf{f}_2 = \operatorname{argmin}_{\mathbf{f}} G(\mathbf{f}) + g(\mathbf{f})$, then*

$$\|\mathbf{f}_1 - \mathbf{f}_2\| \leq \frac{1}{\lambda} \max_{\mathbf{f}} \|\nabla g(\mathbf{f})\|.$$

Proof Using the definition of $\mathbf{f}_1$ and $\mathbf{f}_2$, and the fact that G and g are continuous and differentiable everywhere,

$$\nabla G(\mathbf{f}_1) = \nabla G(\mathbf{f}_2) + \nabla g(\mathbf{f}_2) = \mathbf{0}. \tag{5}$$

As $G(\mathbf{f})$ is λ-strongly convex, it follows from Lemma 14 of Shalev-Shwartz (2007) that:

$$(\nabla G(\mathbf{f}_1) - \nabla G(\mathbf{f}_2))^T (\mathbf{f}_1 - \mathbf{f}_2) \geq \lambda \|\mathbf{f}_1 - \mathbf{f}_2\|^2.$$

Combining this with (5) and the Cauchy-Schwartz inequality, we get that

$$\|\mathbf{f}_1 - \mathbf{f}_2\| \cdot \|\nabla g(\mathbf{f}_2)\| \geq (\mathbf{f}_1 - \mathbf{f}_2)^T \nabla g(\mathbf{f}_2) = (\nabla G(\mathbf{f}_1) - \nabla G(\mathbf{f}_2))^T (\mathbf{f}_1 - \mathbf{f}_2) \geq \lambda \|\mathbf{f}_1 - \mathbf{f}_2\|^2.$$

The conclusion follows from dividing both sides by $\lambda \|\mathbf{f}_1 - \mathbf{f}_2\|$. ■

Corollary 8 *If $N(\cdot)$ is differentiable and 1-strongly convex, and ℓ is convex and differentiable with $|\ell'(z)| \leq 1$ for all z, then, the L_2-sensitivity of $J(\mathbf{f}, \mathcal{D})$ is at most $\frac{2}{n\Lambda}$.*

Proof Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ and $\mathcal{D}' = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}'_n, y'_n)\}$ be two data sets that differ in the value of the n-th individual. Moreover, we let $G(\mathbf{f}) = J(\mathbf{f}, \mathcal{D})$, $g(\mathbf{f}) = J(\mathbf{f}, \mathcal{D}') - J(\mathbf{f}, \mathcal{D})$, $\mathbf{f}_1 = \operatorname{argmin}_{\mathbf{f}} J(\mathbf{f}, \mathcal{D})$, and $\mathbf{f}_2 = \operatorname{argmin}_{\mathbf{f}} J(\mathbf{f}, \mathcal{D}')$. Finally, we set $g(\mathbf{f}) = \frac{1}{n}(\ell(y'_n \mathbf{f}^T \mathbf{x}'_n) - \ell(y_n \mathbf{f}^T \mathbf{x}_n))$.

We observe that due to the convexity of ℓ, and 1-strong convexity of $N(\cdot)$, $G(\mathbf{f}) = J(\mathbf{f}, \mathcal{D})$ is Λ-strongly convex. Moreover, $G(\mathbf{f}) + g(\mathbf{f}) = J(\mathbf{f}, \mathcal{D}')$ is also Λ-strongly convex. Finally, due to the differentiability of $N(\cdot)$ and ℓ, $G(\mathbf{f})$ and $g(\mathbf{f})$ are also differentiable at all points. We have:

$$\nabla g(\mathbf{f}) = \frac{1}{n}(y_n \ell'(y_n \mathbf{f}^T \mathbf{x}_n)\mathbf{x}_n - y'_n \ell'(y'_n \mathbf{f}^T \mathbf{x}'_n)\mathbf{x}'_n).$$

As $y_i \in [-1, 1]$, $|\ell'(z)| \leq 1$, for all z, and $||\mathbf{x}_i|| \leq 1$, for any $\mathbf{f}$, $||\nabla g(\mathbf{f})|| \leq \frac{1}{n}(||\mathbf{x}_n - \mathbf{x}'_n||) \leq \frac{1}{n}(||\mathbf{x}_n|| + ||\mathbf{x}'_n||) \leq \frac{2}{n}$. The proof now follows by an application of Lemma 7. ■

3.3.2 PRIVACY GUARANTEES FOR OBJECTIVE PERTURBATION

In this section, we show that Algorithm 2 is ε_p-differentially private. This proof requires stronger assumptions on the loss function than were required in Theorem 6. In certain cases, some of these assumptions can be weakened; for such an example, see Section 3.4.2.

Theorem 9 *If $N(\cdot)$ is 1-strongly convex and doubly differentiable, and $\ell(\cdot)$ is convex and doubly differentiable, with $|\ell'(z)| \leq 1$ and $|\ell''(z)| \leq c$ for all z, then Algorithm 2 is ε_p-differentially private.*

Proof Consider an $\mathbf{f}_{\text{priv}}$ output by Algorithm 2. We observe that given *any* fixed $\mathbf{f}_{\text{priv}}$ and a fixed data set $\mathcal{D}$, there always exists a $\mathbf{b}$ such that Algorithm 2 outputs $\mathbf{f}_{\text{priv}}$ on input $\mathcal{D}$. Because ℓ is differentiable and convex, and $N(\cdot)$ is differentiable, we can take the gradient of the objective function and set it to $\mathbf{0}$ at $\mathbf{f}_{\text{priv}}$. Therefore,

$$\mathbf{b} = -n\Lambda\nabla N(\mathbf{f}_{\text{priv}}) - \sum_{i=1}^{n} y_i \ell'(y_i \mathbf{f}_{\text{priv}}^T \mathbf{x}_i)\mathbf{x}_i - n\Delta\mathbf{f}_{\text{priv}}. \tag{6}$$

Note that (6) holds because for any $\mathbf{f}$, $\nabla\ell(\mathbf{f}^T\mathbf{x}) = \ell'(\mathbf{f}^T\mathbf{x})\mathbf{x}$.

We claim that as ℓ is differentiable and $J(\mathbf{f}, \mathcal{D}) + \frac{\Lambda}{2}||\mathbf{f}||^2$ is strongly convex, given a data set $\mathcal{D} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, there is a bijection between $\mathbf{b}$ and $\mathbf{f}_{\text{priv}}$. The relation (6) shows that two different $\mathbf{b}$ values cannot result in the same $\mathbf{f}_{\text{priv}}$. Furthermore, since the objective is strictly convex, for a fixed $\mathbf{b}$ and $\mathcal{D}$, there is a unique $\mathbf{f}_{\text{priv}}$; therefore the map from $\mathbf{b}$ to $\mathbf{f}_{\text{priv}}$ is injective. The relation (6) also shows that for any $\mathbf{f}_{\text{priv}}$ there exists a $\mathbf{b}$ for which $\mathbf{f}_{\text{priv}}$ is the minimizer, so the map from $\mathbf{b}$ to $\mathbf{f}_{\text{priv}}$ is surjective.

To show ε_p-differential privacy, we need to compute the ratio $g(\mathbf{f}_{\text{priv}}|\mathcal{D})/g(\mathbf{f}_{\text{priv}}|\mathcal{D}')$ of the densities of $\mathbf{f}_{\text{priv}}$ under the two data sets $\mathcal{D}$ and $\mathcal{D}'$. This ratio can be written as (Billingsley, 1995)

$$\frac{g(\mathbf{f}_{\text{priv}}|\mathcal{D})}{g(\mathbf{f}_{\text{priv}}|\mathcal{D}')} = \frac{\mu(\mathbf{b}|\mathcal{D})}{\mu(\mathbf{b}'|\mathcal{D}')} \cdot \frac{|\det(\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}))|^{-1}}{|\det(\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}'|\mathcal{D}'))|^{-1}},$$

where $\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D})$, $\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}')$ are the Jacobian matrices of the mappings from $\mathbf{f}_{\text{priv}}$ to $\mathbf{b}$, and $\mu(\mathbf{b}|\mathcal{D})$ and $\mu(\mathbf{b}|\mathcal{D}')$ are the densities of $\mathbf{b}$ given the output $\mathbf{f}_{\text{priv}}$, when the data sets are $\mathcal{D}$ and $\mathcal{D}'$ respectively.

First, we bound the ratio of the Jacobian determinants. Let $\mathbf{b}^{(j)}$ denote the j-th coordinate of $\mathbf{b}$. From (6) we have

$$\mathbf{b}^{(j)} = -n\Lambda\nabla N(\mathbf{f}_{\text{priv}})^{(j)} - \sum_{i=1}^{n} \ell'(y_i \mathbf{f}_{\text{priv}}^T \mathbf{x}_i)\mathbf{x}_i^{(j)} - n\Delta\mathbf{f}_{\text{priv}}^{(j)}.$$

Given a data set $\mathcal{D}$, the (j,k)-th entry of the Jacobian matrix $\mathbf{J}(\mathbf{f} \to \mathbf{b}|\mathcal{D})$ is

$$\frac{\partial \mathbf{b}^{(j)}}{\partial \mathbf{f}_{\text{priv}}^{(k)}} = -n\Lambda\nabla^2 N(\mathbf{f}_{\text{priv}})^{(j,k)} - \sum_i y_i^2 \ell''(y_i \mathbf{f}_{\text{priv}}^T \mathbf{x}_i)\mathbf{x}_i^{(j)}\mathbf{x}_i^{(k)} - n\Delta 1(j=k),$$

where $1(\cdot)$ is the indicator function. We note that the Jacobian is defined for all $\mathbf{f}_{\text{priv}}$ because $N(\cdot)$ and ℓ are globally doubly differentiable.

Let $\mathcal{D}$ and $\mathcal{D}'$ be two data sets which differ in the value of the n-th item such that $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n-1}, y_{n-1}), (\mathbf{x}_n, y_n)\}$ and $\mathcal{D}' = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n-1}, y_{n-1}), (\mathbf{x}'_n, y'_n)\}$. Moreover, we define matrices A and E as follows:

$$\begin{aligned} A &= n\Lambda\nabla^2 N(\mathbf{f}_{\text{priv}}) + \sum_{i=1}^{n} y_i^2 \ell''(y_i \mathbf{f}_{\text{priv}}^T \mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^T + n\Delta I_d, \\ E &= -y_n^2 \ell''(y_n \mathbf{f}_{\text{priv}}^T \mathbf{x}_n)\mathbf{x}_n\mathbf{x}_n^T + (y'_n)^2 \ell''(y'_n \mathbf{f}_{\text{priv}}^T \mathbf{x}'_n)\mathbf{x}'_n{\mathbf{x}'_n}^T. \end{aligned}$$

Then, $\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}) = -A$, and $\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}') = -(A+E)$.

Let $\lambda_1(M)$ and $\lambda_2(M)$ denote the largest and second largest eigenvalues of a matrix M. As E has rank at most 2, from Lemma 10,

$$\begin{aligned} \frac{|\det(\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}'))|}{|\det(\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}))|} &= \frac{|\det(A+E)|}{|\det A|} \\ &= |1 + \lambda_1(A^{-1}E) + \lambda_2(A^{-1}E) + \lambda_1(A^{-1}E)\lambda_2(A^{-1}E)|. \end{aligned}$$

For a 1-strongly convex function N, the Hessian $\nabla^2 N(\mathbf{f}_{\text{priv}})$ has eigenvalues greater than 1 (Boyd and Vandenberghe, 2004). Since we have assumed ℓ is doubly differentiable and convex, any eigenvalue of A is therefore at least $n\Lambda + n\Delta$; therefore, for $j = 1, 2$, $|\lambda_j(A^{-1}E)| \leq \frac{|\lambda_j(E)|}{n(\Lambda+\Delta)}$. Applying the triangle inequality to the trace norm:

$$|\lambda_1(E)| + |\lambda_2(E)| \leq |y_n^2 \ell''(y_n \mathbf{f}_{\text{priv}}^T \mathbf{x}_n)| \cdot \|\mathbf{x}_n\| + |-(y'_n)^2 \ell''(y'_n \mathbf{f}_{\text{priv}}^T \mathbf{x}'_n)| \cdot \left\|\mathbf{x}'_n\right\|.$$

Then upper bounds on $|y_i|$, $||\mathbf{x}_i||$, and $|\ell''(z)|$ yield

$$|\lambda_1(E)| + |\lambda_2(E)| \leq 2c.$$

Therefore, $|\lambda_1(E)| \cdot |\lambda_2(E)| \leq c^2$, and

$$\frac{|\det(A+E)|}{|\det(A)|} \leq 1 + \frac{2c}{n(\Lambda+\Delta)} + \frac{c^2}{n^2(\Lambda+\Delta)^2} = \left(1 + \frac{c}{n(\Lambda+\Delta)}\right)^2.$$

We now consider two cases. In the first case, $\Delta = 0$, and by definition, in that case, $1 + \frac{2c}{n\Lambda} + \frac{c^2}{n^2\Lambda^2} \leq e^{\varepsilon_p - \varepsilon'_p}$. In the second case, $\Delta > 0$, and in this case, by definition of Δ, $(1 + \frac{c}{n(\Lambda+\Delta)})^2 = e^{\varepsilon_p/2} = e^{\varepsilon_p - \varepsilon'_p}$.

Next, we bound the ratio of the densities of $\mathbf{b}$. We observe that as $|\ell'(z)| \leq 1$, for any z and $|y_i|, ||\mathbf{x}_i|| \leq 1$, for data sets $\mathcal{D}$ and $\mathcal{D}'$ which differ by one value,

$$\mathbf{b}' - \mathbf{b} = y_n \ell'(y_n \mathbf{f}_{\text{priv}}^T \mathbf{x}_n)\mathbf{x}_n - y'_n \ell'(y_n \mathbf{f}_{\text{priv}}^T \mathbf{x}'_n)\mathbf{x}'_n.$$

This implies that:

$$\|\mathbf{b}\| - \left\|\mathbf{b}'\right\| \leq \left\|\mathbf{b} - \mathbf{b}'\right\| \leq 2.$$

We can write:

$$\frac{\mu(\mathbf{b}|\mathcal{D})}{\mu(\mathbf{b}'|\mathcal{D}')} = \frac{||\mathbf{b}||^{d-1} e^{-\varepsilon'_p||\mathbf{b}||/2} \cdot \frac{1}{\text{surf}(||\mathbf{b}||)}}{||\mathbf{b}'||^{d-1} e^{-\varepsilon'_p||\mathbf{b}'||/2} \cdot \frac{1}{\text{surf}(||\mathbf{b}'||)}} \leq e^{\varepsilon'_p(||\mathbf{b}||-||\mathbf{b}'||)/2} \leq e^{\varepsilon'_p},$$

where $\text{surf}(x)$ denotes the surface area of the sphere in d dimensions with radius x. Here the last step follows from the fact that $\text{surf}(x) = \text{surf}(1)x^{d-1}$, where $\text{surf}(1)$ is the surface area of the unit sphere in $\mathbb{R}^d$.

Finally, we are ready to bound the ratio of densities:

$$\begin{aligned}
\frac{g(\mathbf{f}_{\text{priv}}|\mathcal{D})}{g(\mathbf{f}_{\text{priv}}|\mathcal{D}')} &= \frac{\mu(\mathbf{b}|\mathcal{D})}{\mu(\mathbf{b}'|\mathcal{D}')} \cdot \frac{|\det(\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}|\mathcal{D}'))|}{|\det(\mathbf{J}(\mathbf{f}_{\text{priv}} \to \mathbf{b}'|\mathcal{D}))|} \\
&= \frac{\mu(\mathbf{b}|\mathcal{D})}{\mu(\mathbf{b}'|\mathcal{D}')} \cdot \frac{|\det(A+E)|}{|\det A|} \\
&\leq e^{\varepsilon'_p} \cdot e^{\varepsilon_p - \varepsilon'_p} \\
&\leq e^{\varepsilon_p}.
\end{aligned}$$

Thus, Algorithm 2 satisfies Definition 2. ■

Lemma 10 *If A is full rank, and if E has rank at most 2, then,*

$$\frac{\det(A+E) - \det(A)}{\det(A)} = \lambda_1(A^{-1}E) + \lambda_2(A^{-1}E) + \lambda_1(A^{-1}E)\lambda_2(A^{-1}E),$$

where $\lambda_j(Z)$ is the j-th eigenvalue of matrix Z.

Proof Note that E has rank at most 2, so $A^{-1}E$ also has rank at most 2. Using the fact that $\lambda_i(I + A^{-1}E) = 1 + \lambda_i(A^{-1}E)$,

$$\begin{aligned}
\frac{\det(A+E) - \det(A)}{\det A} &= \det(I + A^{-1}E) - 1 \\
&= (1 + \lambda_1(A^{-1}E))(1 + \lambda_2(A^{-1}E)) - 1 \\
&= \lambda_1(A^{-1}E) + \lambda_2(A^{-1}E) + \lambda_1(A^{-1}E)\lambda_2(A^{-1}E).
\end{aligned}$$

■

3.4 Application to Classification

In this section, we show how to use our results to provide privacy-preserving versions of logistic regression and support vector machines.

3.4.1 LOGISTIC REGRESSION

One popular ERM classification algorithm is regularized logistic regression. In this case, $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, and the loss function is $\ell_{\mathrm{LR}}(z) = \log(1+e^{-z})$. Taking derivatives and double derivatives,

$$\ell'_{\mathrm{LR}}(z) = \frac{-1}{(1+e^z)},$$
$$\ell''_{\mathrm{LR}}(z) = \frac{1}{(1+e^{-z})(1+e^z)}.$$

Note that ℓ_{LR} is continuous, differentiable and doubly differentiable, with $c \leq \frac{1}{4}$. Therefore, we can plug in logistic loss directly to Theorems 6 and 9 to get the following result.

Corollary 11 *The output of Algorithm 1 with $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, $\ell = \ell_{\mathrm{LR}}$ is an ε_p-differentially private approximation to logistic regression. The output of Algorithm 2 with $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, $c = \frac{1}{4}$, and $\ell = \ell_{\mathrm{LR}}$, is an ε_p-differentially private approximation to logistic regression.*

We quantify how well the outputs of Algorithms 1 and 2 approximate (non-private) logistic regression in Section 4.

3.4.2 SUPPORT VECTOR MACHINES

Another very commonly used classifier is L_2-regularized support vector machines. In this case, again, $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, and

$$\ell_{\mathrm{SVM}}(z) = \max(0, 1-z).$$

Notice that this loss function is continuous, but not differentiable, and thus it does not satisfy conditions in Theorems 6 and 9.

There are two alternative solutions to this. First, we can approximate ℓ_{SVM} by a different loss function, which is doubly differentiable, as follows (see also Chapelle, 2007):

$$\ell_s(z) = \begin{cases} 0 & \text{if} \quad z > 1+h \\ -\frac{(1-z)^4}{16h^3} + \frac{3(1-z)^2}{8h} + \frac{1-z}{2} + \frac{3h}{16} & \text{if} \quad |1-z| \leq h \\ 1-z & \text{if} \quad z < 1-h. \end{cases}$$

As $h \to 0$, this loss approaches the hinge loss. Taking derivatives, we observe that:

$$\ell'_s(z) = \begin{cases} 0 & \text{if} \quad z > 1+h \\ \frac{(1-z)^3}{4h^3} - \frac{3(1-z)}{4h} - \frac{1}{2} & \text{if} \quad |1-z| \leq h \\ -1 & \text{if} \quad z < 1-h. \end{cases}$$

Moreover,

$$\ell''_s(z) = \begin{cases} 0 & \text{if} \quad z > 1+h \\ -\frac{3(1-z)^2}{4h^3} + \frac{3}{4h} & \text{if} \quad |1-z| \leq h \\ 0 & \text{if} \quad z < 1-h. \end{cases}$$

Observe that this implies that $|\ell''_s(z)| \leq \frac{3}{4h}$ for all h and z. Moreover, ℓ_s is convex, as $\ell''_s(z) \geq 0$ for all z. Therefore, ℓ_s can be used in Theorems 6 and 9, which gives us privacy-preserving approximations to regularized support vector machines.

Corollary 12 *The output of Algorithm 1 with $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, and $\ell = \ell_s$ is an ε_p-differentially private approximation to support vector machines. The output of Algorithm 2 with $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, $c = \frac{3}{4h}$, and $\ell = \ell_s$ is an ε_p-differentially private approximation to support vector machines.*

The second solution is to use Huber Loss, as suggested by Chapelle (2007), which is defined as follows:

$$\ell_{\text{Huber}}(z) = \begin{cases} 0 & \text{if} \quad z > 1+h \\ \frac{1}{4h}(1+h-z)^2 & \text{if} \quad |1-z| \leq h \\ 1-z & \text{if} \quad z < 1-h. \end{cases} \tag{7}$$

Observe that Huber loss is convex and differentiable, and piecewise doubly-differentiable, with $c = \frac{1}{2h}$. However, it is not globally doubly differentiable, and hence the Jacobian in the proof of Theorem 9 is undefined for certain values of $\mathbf{f}$. However, we can show that in this case, Algorithm 2, when run with $c = \frac{1}{2h}$ satisfies Definition 3.

Let G denote the map from $\mathbf{f}_{\text{priv}}$ to $\mathbf{b}$ in (6) under $\mathcal{B} = \mathcal{D}$, and H denote the map under $\mathcal{B} = \mathcal{D}'$. By definition, the probability $\mathbb{P}(\mathbf{f}_{\text{priv}} \in \mathcal{S} \mid \mathcal{B} = \mathcal{D}) = \mathbb{P}_{\mathbf{b}}(\mathbf{b} \in G(\mathcal{S}))$.

Corollary 13 *Let $\mathbf{f}_{\text{priv}}$ be the output of Algorithm 2 with $\ell = \ell_{\text{Huber}}$, $c = \frac{1}{2h}$, and $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||_2^2$. For any set $\mathcal{S}$ of possible values of $\mathbf{f}_{\text{priv}}$, and any pair of data sets $\mathcal{D}$, $\mathcal{D}'$ which differ in the private value of one person $(\mathbf{x}_n, y_n)$,*

$$e^{-\varepsilon_p}\mathbb{P}(\mathcal{S} \mid \mathcal{B} = \mathcal{D}') \leq \mathbb{P}(\mathcal{S} \mid \mathcal{B} = \mathcal{D}) \leq e^{\varepsilon_p}\mathbb{P}(\mathcal{S} \mid \mathcal{B} = \mathcal{D}').$$

Proof Consider the event $\mathbf{f}_{\text{priv}} \in \mathcal{S}$. Let $\mathcal{T} = G(\mathcal{S})$ and $\mathcal{T}' = H(\mathcal{S})$. Because G is a bijection, we have

$$\mathbb{P}(\mathbf{f}_{\text{priv}} \in \mathcal{S} \mid \mathcal{B} = \mathcal{D}) = \mathbb{P}_{\mathbf{b}}(\mathbf{b} \in \mathcal{T} \mid \mathcal{B} = \mathcal{D}),$$

and a similar expression when $\mathcal{B} = \mathcal{D}'$.

Now note that $\ell'_{\text{Huber}}(z)$ is only non-differentiable for a finite number of values of z. Let $\mathcal{Z}$ be the set of these values of z.

$$\mathcal{C} = \{\mathbf{f} : y\mathbf{f}^T\mathbf{x} = z \in \mathcal{Z},\ (\mathbf{x}, y) \in \mathcal{D} \cup \mathcal{D}'\}.$$

Pick a tuple $(z, (\mathbf{x}, y)) \in \mathcal{Z} \times (\mathcal{D} \cup \mathcal{D}')$. The set of $\mathbf{f}$ such that $y\mathbf{f}^T\mathbf{x} = z$ is a hyperplane in $\mathbb{R}^d$. Since $\nabla N(\mathbf{f}) = \mathbf{f}/2$ and ℓ' is piecewise linear, from (6) we see that the set of corresponding $\mathbf{b}$'s is also piecewise linear, and hence has Lebesgue measure 0. Since the measure corresponding to $\mathbf{b}$ is absolutely continuous with respect to the Lebesgue measure, this hyperplane has probability 0 under $\mathbf{b}$ as well. Since $\mathcal{C}$ is a finite union of such hyperplanes, we have $\mathbb{P}(\mathbf{b} \in G(\mathcal{C})) = 0$.

Thus we have $\mathbb{P}_{\mathbf{b}}(\mathcal{T} \mid \mathcal{B} = \mathcal{D}) = \mathbb{P}_{\mathbf{b}}(G(\mathcal{S} \setminus \mathcal{C}) \mid \mathcal{B} = \mathcal{D})$, and similarly for $\mathcal{D}'$. From the definition of G and H, for $\mathbf{f} \in \mathcal{S} \setminus \mathcal{C}$,

$$H(\mathbf{f}) = G(\mathbf{f}) + y_n\ell'(y_n\mathbf{f}^T\mathbf{x}_n)\mathbf{x}_n - y'_n\ell'(y'_n\mathbf{f}^T\mathbf{x}'_n)\mathbf{x}'_n.$$

since $\mathbf{f} \notin \mathcal{C}$, this mapping shows that if $\mathbb{P}_{\mathbf{b}}(G(\mathcal{S} \setminus \mathcal{C}) \mid \mathcal{B} = \mathcal{D}) = 0$ then we must have $\mathbb{P}_{\mathbf{b}}(H(\mathcal{S} \setminus \mathcal{C}) \mid \mathcal{B} = \mathcal{D}) = 0$. Thus the result holds for sets of measure 0. If $\mathcal{S} \setminus \mathcal{C}$ has positive measure we can

calculate the ratio of the probabilities for $\mathbf{f}_{\text{priv}}$ for which the loss is twice-differentiable. For such $\mathbf{f}_{\text{priv}}$ the Jacobian is also defined, and we can use a method similar to Theorem 9 to prove the result. ■

Remark: Because the privacy proof for Algorithm 1 does not require the analytic properties of 2, we can also use Huber loss in Algorithm 1 to get an ε_g-differentially private approximation to the SVM. We quantify how well the outputs of Algorithms 1 and 2 approximate private support vector machines in Section 4. These approximations to the hinge loss are necessary because of the analytic requirements of Theorems 6 and 9 on the loss function. Because the requirements of Theorem 9 are stricter, it may be possible to use an approximate loss in Algorithm 1 that would not be admissible in Algorithm 2.

4. Generalization Performance

In this section, we provide guarantees on the performance of privacy-preserving ERM algorithms in Section 3. We provide these bounds for L_2-regularization. To quantify this performance, we will assume that the n entries in the data set $\mathcal{D}$ are drawn i.i.d. according to a fixed distribution $P(\mathbf{x}, y)$. We measure the performance of these algorithms by the number of samples n required to acheive error $L^* + \varepsilon_g$, where L^* is the loss of a reference ERM predictor $\mathbf{f}_0$. This resulting bound on ε_g will depend on the norm $\|\mathbf{f}_0\|$ of this predictor. By choosing an upper bound ν on the norm, we can interpret the result as saying that the privacy-preserving classifier will have error ε_g more than that of any predictor with $\|\mathbf{f}_0\| \leq \nu$.

Given a distribution P the expected loss $L(\mathbf{f})$ for a classifier $\mathbf{f}$ is

$$L(\mathbf{f}) = \mathbb{E}_{(\mathbf{x},y)\sim P}\left[\ell(\mathbf{f}^T\mathbf{x}, y)\right].$$

The sample complexity for generalization error ε_g against a classifier $\mathbf{f}_0$ is number of samples n required to achieve error $L(\mathbf{f}_0) + \varepsilon_g$ under any data distribution P. We would like the sample complexity to be low.

For a fixed P we define the following function, which will be useful in our analysis:

$$J(\mathbf{f}) = L(\mathbf{f}) + \frac{\Lambda}{2}\|\mathbf{f}\|^2.$$

The function $J(\mathbf{f})$ is the expectation (over P) of the non-private L_2-regularized ERM objective evaluated at $\mathbf{f}$.

For non-private ERM, Shalev-Shwartz and Srebro (2008) show that for a given $\mathbf{f}_0$ with loss $L(\mathbf{f}_0) = L^*$, if the number of data points satisfies

$$n > C\frac{||\mathbf{f}_0||^2 \log(\frac{1}{\delta})}{\varepsilon_g^2}$$

for some constant C, then the excess loss of the L_2-regularized SVM solution $\mathbf{f}_{svm}$ satisfies $L(\mathbf{f}_{svm}) \leq L(\mathbf{f}_0) + \varepsilon_g$. This order growth will hold for our results as well. It also serves as a reference against which we can compare the additional burden on the sample complexity imposed by the privacy constraints.

For most learning problems, we require the generalization error $\varepsilon_g < 1$. Moreover, it is also typically the case that for more difficult learning problems, $||\mathbf{f}_0||$ is higher. For example, for regularized

SVM, $\frac{1}{||\mathbf{f}_0||}$ is the margin of classification, and as a result, $||\mathbf{f}_0||$ is higher for learning problems with smaller margin. From the bounds provided in this section, we note that the dominating term in the sample requirement for objective perturbation has a better dependence on $||\mathbf{f}_0||$ as well as $\frac{1}{\varepsilon_g}$; as a result, for more difficult learning problems, we expect objective perturbation to perform better than output perturbation.

4.1 Output Perturbation

First, we provide performance guarantees for Algorithm 1, by providing a bound on the number of samples required for Algorithm 1 to produce a classifier with low error.

Definition 14 *A function $g(z) : \mathbb{R} \to \mathbb{R}$ is c-Lipschitz if for all pairs (z_1, z_2) we have $|g(z_1) - g(z_2)| \leq c|z_1 - z_2|$.*

Recall that if a function $g(z)$ is differentiable, with $|g'(z)| \leq r$ for all z, then $g(z)$ is also r-Lipschitz.

Theorem 15 *Let $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, and let $\mathbf{f}_0$ be a classifier such that $L(\mathbf{f}_0) = L^*$, and let $\delta > 0$. If ℓ is differentiable and continuous with $|\ell'(z)| \leq 1$, the derivative ℓ' is c-Lipschitz, the data $\mathcal{D}$ is drawn i.i.d. according to P, then there exists a constant C such that if the number of training samples satisfies*

$$n > C \max\left(\frac{||\mathbf{f}_0||^2 \log(\frac{1}{\delta})}{\varepsilon_g^2}, \frac{d \log(\frac{d}{\delta})||\mathbf{f}_0||}{\varepsilon_g \varepsilon_p}, \frac{d \log(\frac{d}{\delta}) c^{1/2} ||\mathbf{f}_0||^2}{\varepsilon_g^{3/2} \varepsilon_p} \right), \tag{8}$$

where d is the dimension of the data space, then the output $\mathbf{f}_{\text{priv}}$ of Algorithm 1 satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\text{priv}}) \leq L^* + \varepsilon_g\right) \geq 1 - 2\delta.$$

Proof Let

$$\mathbf{f}_{\text{rtr}} = \operatorname*{argmin}_{\mathbf{f}} J(\mathbf{f}),$$
$$\mathbf{f}^* = \operatorname*{argmin}_{\mathbf{f}} J(\mathbf{f}, \mathcal{D}),$$

and $\mathbf{f}_{\text{priv}}$ denote the output of Algorithm 1. Using the analysis method of Shalev-Shwartz and Srebro (2008) shows

$$L(\mathbf{f}_{\text{priv}}) = L(\mathbf{f}_0) + (J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}})) + (J(\mathbf{f}_{\text{rtr}}) - J(\mathbf{f}_0)) + \frac{\Lambda}{2}||\mathbf{f}_0||^2 - \frac{\Lambda}{2}||\mathbf{f}_{\text{priv}}||^2. \tag{9}$$

We will bound the terms on the right-hand side of (9).

For a regularizer $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$ the Hessian satisfies $||\nabla^2 N(\mathbf{f})||_2 \leq 1$. Therefore, from Lemma 16, with probability $1 - \delta$ over the privacy mechanism,

$$J(\mathbf{f}_{\text{priv}}, \mathcal{D}) - J(\mathbf{f}^*, \mathcal{D}) \leq \frac{8d^2 \log^2(d/\delta)(c + \Lambda)}{\Lambda^2 n^2 \varepsilon_p^2}.$$

Furthermore, the results of Sridharan et al. (2008) show that with probability $1-\delta$ over the choice of the data distribution,

$$J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}}) \leq 2(J(\mathbf{f}_{\text{priv}}, \mathcal{D}) - J(\mathbf{f}^*, \mathcal{D})) + O\left(\frac{\log(1/\delta)}{\Lambda n}\right).$$

The constant in the last term depends on the derivative of the loss and the bound on the data points, which by assumption are bounded. Combining the preceeding two statements, with probability $1-2\delta$ over the noise in the privacy mechanism and the data distribution, the second term in the right-hand-side of (9) is at most:

$$J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}}) \leq \frac{16d^2\log^2(d/\delta)(c+\Lambda)}{\Lambda^2 n^2 \varepsilon_p^2} + O\left(\frac{\log(1/\delta)}{\Lambda n}\right). \tag{10}$$

By definition of $\mathbf{f}_{\text{rtr}}$, the difference $(J(\mathbf{f}_{\text{rtr}}) - J(\mathbf{f}_0)) \leq 0$. Setting $\Lambda = \frac{\varepsilon_g}{||\mathbf{f}_0||^2}$ in (9) and using (10), we obtain

$$L(\mathbf{f}_{\text{priv}}) \leq L(\mathbf{f}_0) + \frac{16||\mathbf{f}_0||^4 d^2 \log^2(d/\delta)(c+\varepsilon_g/||\mathbf{f}_0||^2)}{n^2\varepsilon_g^2\varepsilon_p^2} + O\left(||\mathbf{f}_0||^2\frac{\log(1/\delta)}{n\varepsilon_g}\right) + \frac{\varepsilon_g}{2}.$$

Solving for n to make the total excess error equal to ε_g yields (8). ■

Lemma 16 *Suppose $N(\cdot)$ is doubly differentiable with $||\nabla^2 N(\mathbf{f})||_2 \leq \eta$ for all $\mathbf{f}$, and suppose that ℓ is differentiable and has continuous and c-Lipschitz derivatives. Given training data $\mathcal{D}$, let $\mathbf{f}^*$ be a classifier that minimizes $J(\mathbf{f}, \mathcal{D})$ and let $\mathbf{f}_{\text{priv}}$ be the classifier output by Algorithm 1. Then*

$$\mathbb{P}_{\mathbf{b}}\left(J(\mathbf{f}_{\text{priv}}, \mathcal{D}) \leq J(\mathbf{f}^*, \mathcal{D}) + \frac{2d^2(c+\Lambda\eta)\log^2(d/\delta)}{\Lambda^2 n^2 \varepsilon_p^2}\right) \geq 1-\delta,$$

where the probability is taken over the randomness in the noise $\mathbf{b}$ of Algorithm 1.

Note that when ℓ is doubly differentiable, c is an upper bound on the double derivative of ℓ, and is the same as the constant c in Theorem 9.

Proof Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, and recall that $||\mathbf{x}_i|| \leq 1$, and $|y_i| \leq 1$. As $N(\cdot)$ and ℓ are differentiable, we use the Mean Value Theorem to show that for some t between 0 and 1,

$$\begin{aligned} J(\mathbf{f}_{\text{priv}}, \mathcal{D}) - J(\mathbf{f}^*, \mathcal{D}) &= (\mathbf{f}_{\text{priv}} - \mathbf{f}^*)^T \nabla J(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}}) \\ &\leq ||\mathbf{f}_{\text{priv}} - \mathbf{f}^*|| \cdot ||\nabla J(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})||, \end{aligned} \tag{11}$$

where the second step follows by an application of the Cauchy-Schwartz inequality. Recall that

$$\nabla J(\mathbf{f}, \mathcal{D}) = \Lambda \nabla N(\mathbf{f}) + \frac{1}{n}\sum_i y_i \ell'(y_i \mathbf{f}^T \mathbf{x}_i)\mathbf{x}_i.$$

Moreover, recall that $\nabla J(\mathbf{f}^*, \mathcal{D}) = 0$, from the optimality of $\mathbf{f}^*$. Therefore,

$$\begin{aligned} \nabla J(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}}, \mathcal{D}) &= \nabla J(\mathbf{f}^*, \mathcal{D}) - \Lambda(\nabla N(\mathbf{f}^*) - \nabla N(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})) \\ &\quad - \frac{1}{n}\sum_i y_i \left(\ell'(y_i(\mathbf{f}^*)^T\mathbf{x}_i) - \ell'(y_i(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})^T\mathbf{x}_i)\right)\mathbf{x}_i. \end{aligned} \tag{12}$$

Now, from the Lipschitz condition on ℓ, for each i we can upper bound each term in the summation above:

$$\begin{aligned}
&\left\| y_i \left(\ell'(y_i(\mathbf{f}^*)^T\mathbf{x}_i) - \ell'(y_i(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})^T\mathbf{x}_i) \right) \mathbf{x}_i \right\| \\
&\qquad \leq |y_i| \cdot ||\mathbf{x}_i|| \cdot |\ell'(y_i(\mathbf{f}^*)^T\mathbf{x}_i) - \ell'(y_i(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})^T\mathbf{x}_i)| \\
&\qquad \leq |y_i| \cdot ||\mathbf{x}_i|| \cdot c \cdot |y_i(1-t)(\mathbf{f}^* - \mathbf{f}_{\text{priv}})^T\mathbf{x}_i| \\
&\qquad \leq c(1-t)|y_i|^2 \cdot ||\mathbf{x}_i||^2 \cdot ||\mathbf{f}^* - \mathbf{f}_{\text{priv}}|| \\
&\qquad \leq c(1-t)||\mathbf{f}^* - \mathbf{f}_{\text{priv}}||. \qquad (13)
\end{aligned}$$

The third step follows because ℓ' is c-Lipschitz and the last step follows from the bounds on $|y_i|$ and $||\mathbf{x}_i||$. Because N is doubly differentiable, we can apply the Mean Value Theorem again to conclude that

$$||\nabla N(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}}) - \nabla N(\mathbf{f}^*)|| \leq (1-t)||\mathbf{f}_{\text{priv}} - \mathbf{f}^*|| \cdot ||\nabla^2 N(\mathbf{f}'')||_2 \qquad (14)$$

for some $\mathbf{f}'' \in \mathbb{R}^d$.

As $0 \leq t \leq 1$, we can combine (12), (13), and (14) to obtain

$$\begin{aligned}
\left\| \nabla J(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}}, \mathcal{D}) \right\| &\leq \left\| \Lambda(\nabla N(\mathbf{f}^*) - \nabla N(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})) \right\| \\
&\quad + \left\| \frac{1}{n} \sum_i y_i (\ell'(y_i(\mathbf{f}^*)^T\mathbf{x}_i) - \ell'(y_i(t\mathbf{f}^* + (1-t)\mathbf{f}_{\text{priv}})^T\mathbf{x}_i))\mathbf{x}_i \right\| \\
&\leq (1-t) \left\| \mathbf{f}_{\text{priv}} - \mathbf{f}^* \right\| \cdot \left(\Lambda\eta + \frac{1}{n} \cdot n \cdot c \right) \\
&\leq \left\| \mathbf{f}_{\text{priv}} - \mathbf{f}^* \right\| (\Lambda\eta + c). \qquad (15)
\end{aligned}$$

From the definition of Algorithm 1, $\mathbf{f}_{\text{priv}} - \mathbf{f}^* = \mathbf{b}$, where $\mathbf{b}$ is the noise vector. Now we can apply Lemma 17 to $||\mathbf{f}_{\text{priv}} - \mathbf{f}^*||$, with parameters $k = d$, and $\theta = \frac{2}{\Lambda n \varepsilon_p}$. From Lemma 17, with probability $1 - \delta$, $||\mathbf{f}_{\text{priv}} - \mathbf{f}^*|| \leq \frac{2d \log(\frac{d}{\delta})}{\Lambda n \varepsilon_p}$. The Lemma follows by combining this with Equations 15 and 11. ■

Lemma 17 *Let X be a random variable drawn from the distribution $\Gamma(k, \theta)$, where k is an integer. Then,*

$$\mathbb{P}\left(X < k\theta \log\left(\frac{k}{\delta} \right) \right) \geq 1 - \delta.$$

Proof Since k is an integer, we can decompose X distributed according to $\Gamma(k, \theta)$ as a summation

$$X = X_1 + \ldots + X_k,$$

where $X_1, X_2, \ldots, X_k$ are independent exponential random variables with mean θ. For each i we have $\mathbb{P}(X_i \geq \theta \log(k/\delta)) = \delta/k$. Now,

$$\begin{aligned}\mathbb{P}(X < k\theta\log(k/\delta)) &\geq \mathbb{P}(X_i < \theta\log(k/\delta)\ i = 1,2,\ldots,k)\\ &= (1-\delta/k)^k\\ &\geq 1-\delta.\end{aligned}$$

■

4.2 Objective Perturbation

We now establish performance bounds on Algorithm 2. The bound can be summarized as follows.

Theorem 18 *Let $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$, and let $\mathbf{f}_0$ be a classifier with expected loss $L(\mathbf{f}_0) = L^*$. Let ℓ be convex, doubly differentiable, and let its derivatives satisfy $|\ell'(z)| \leq 1$ and $|\ell''(z)| \leq c$ for all z. Then there exists a constant C such that for $\delta > 0$, if the n training samples in $\mathcal{D}$ are drawn i.i.d. according to P, and if*

$$n > C\max\left(\frac{||\mathbf{f}_0||^2\log(1/\delta)}{\varepsilon_g^2}, \frac{c||\mathbf{f}_0||^2}{\varepsilon_g\varepsilon_p}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||}{\varepsilon_g\varepsilon_p}\right),$$

then the output $\mathbf{f}_{\text{priv}}$ of Algorithm 2 satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\text{priv}}) \leq L^* + \varepsilon_g\right) \geq 1 - 2\delta.$$

Proof Let

$$\begin{aligned}\mathbf{f}_{\text{rtr}} &= \operatorname*{argmin}_{\mathbf{f}} J(\mathbf{f}),\\ \mathbf{f}^* &= \operatorname*{argmin}_{\mathbf{f}} J(\mathbf{f}, \mathcal{D}),\end{aligned}$$

and $\mathbf{f}_{\text{priv}}$ denote the output of Algorithm 1. As in Theorem 15, the analysis of Shalev-Shwartz and Srebro (2008) shows

$$L(\mathbf{f}_{\text{priv}}) = L(\mathbf{f}_0) + (J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}})) + (J(\mathbf{f}_{\text{rtr}}) - J(\mathbf{f}_0)) + \frac{\Lambda}{2}||\mathbf{f}_0||^2 - \frac{\Lambda}{2}||\mathbf{f}_{\text{priv}}||^2. \tag{16}$$

We will bound each of the terms on the right-hand-side.

If $n > \frac{c||\mathbf{f}_0||^2}{\varepsilon_g\varepsilon_p}$ and $\Lambda > \frac{\varepsilon_g}{4||\mathbf{f}_0||^2}$, then $n\Lambda > \frac{c}{4\varepsilon_p}$, so from the definition of ε_p' in Algorithm 2,

$$\varepsilon_p' = \varepsilon_p - 2\log\left(1 + \frac{c}{n\Lambda}\right) = \varepsilon_p - 2\log\left(1 + \frac{\varepsilon_p}{4}\right) \geq \varepsilon_p - \frac{\varepsilon_p}{2},$$

where the last step follows because $\log(1+x) \leq x$ for $x \in [0,1]$. Note that for these values of Λ we have $\varepsilon_p' > 0$.

Therefore, we can apply Lemma 19 to conclude that with probability at least $1-\delta$ over the privacy mechanism,

$$J(\mathbf{f}_{\text{priv}}, \mathcal{D}) - J(\mathbf{f}^*, \mathcal{D}) \leq \frac{4d^2 \log^2(d/\delta)}{\Lambda n^2 \varepsilon_p^2}.$$

From Sridharan et al. (2008),

$$\begin{aligned} J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}}) &\leq 2(J(\mathbf{f}_{\text{priv}}, \mathcal{D}) - J(\mathbf{f}^*, \mathcal{D})) + O\left(\frac{\log(1/\delta)}{\Lambda n}\right) \\ &\leq \frac{8d^2 \log^2(d/\delta)}{\Lambda n^2 \varepsilon_p^2} + O\left(\frac{\log(1/\delta)}{\Lambda n}\right). \end{aligned}$$

By definition of $\mathbf{f}^*$, we have $J(\mathbf{f}_{\text{rtr}}) - J(\mathbf{f}_0) \leq 0$. If Λ is set to be $\frac{\varepsilon_g}{||\mathbf{f}_0||^2}$, then, the fourth quantity in Equation 16 is at most $\frac{\varepsilon_g}{2}$. The theorem follows by solving for n to make the total excess error at most ε_g. ■

The following lemma is analogous to Lemma 16, and it establishes a bound on the distance between the output of Algorithm 2, and non-private regularized ERM. We note that this bound holds when Algorithm 2 has $\varepsilon'_p > 0$, that is, when $\Delta = 0$. Ensuring that $\Delta = 0$ requires an additional condition on n, which is stated in Theorem 18.

Lemma 19 *Let $\varepsilon'_p > 0$. Let $\mathbf{f}^* = \text{argmin} J(\mathbf{f}, \mathcal{D})$, and let $\mathbf{f}_{\text{priv}}$ be the classifier output by Algorithm 2. If $N(\cdot)$ is 1-strongly convex and globally differentiable, and if ℓ is convex and differentiable at all points, with $|\ell'(z)| \leq 1$ for all z, then*

$$\mathbb{P}_{\mathbf{b}}\left(J(\mathbf{f}_{\text{priv}}, \mathcal{D}) \leq J(\mathbf{f}^*, \mathcal{D}) + \frac{4d^2 \log^2(d/\delta)}{\Lambda n^2 \varepsilon_p^2}\right) \geq 1 - \delta,$$

where the probability is taken over the randomness in the noise $\mathbf{b}$ of Algorithm 2.

Proof By the assumption $\varepsilon'_p > 0$, the classifier $\mathbf{f}_{\text{priv}}$ minimizes the objective function $J(\mathbf{f}, \mathcal{D}) + \frac{1}{n}\mathbf{b}^T\mathbf{f}$, and therefore

$$J(\mathbf{f}_{\text{priv}}, \mathcal{D}) \leq J(\mathbf{f}^*, \mathcal{D}) + \frac{1}{n}\mathbf{b}^T(\mathbf{f}^* - \mathbf{f}_{\text{priv}}).$$

First, we try to bound $||\mathbf{f}^* - \mathbf{f}_{\text{priv}}||$. Recall that $\Lambda N(\cdot)$ is Λ-strongly convex and globally differentiable, and ℓ is convex and differentiable. We can therefore apply Lemma 7 with $G(\mathbf{f}) = J(\mathbf{f}, \mathcal{D})$ and $g(\mathbf{f}) = \frac{1}{n}\mathbf{b}^T\mathbf{f}$ to obtain the bound

$$||\mathbf{f}^* - \mathbf{f}_{\text{priv}}|| \leq \frac{1}{\Lambda}\left\|\nabla(\frac{1}{n}\mathbf{b}^T\mathbf{f})\right\| \leq \frac{||\mathbf{b}||}{n\Lambda}.$$

Therefore by the Cauchy-Schwartz inequality,

$$J(\mathbf{f}_{\text{priv}}, \mathcal{D}) - J(\mathbf{f}^*, \mathcal{D}) \leq \frac{||\mathbf{b}||^2}{n^2\Lambda}.$$

Since $||\mathbf{b}||$ is drawn from a $\Gamma(d, \frac{2}{\varepsilon_p})$ distribution, from Lemma 17, with probability $1-\delta$, $||\mathbf{b}|| \leq \frac{2d\log(d/\delta)}{\varepsilon_p}$. The Lemma follows by plugging this in to the previous equation. ■

4.3 Applications

In this section, we examine the sample requirement of privacy-preserving regularized logistic regression and support vector machines. Recall that in both these cases, $N(\mathbf{f}) = \frac{1}{2}||\mathbf{f}||^2$.

Corollary 20 (Logistic Regression) *Let training data $\mathcal{D}$ be generated i.i.d. according to a distribution P and let $\mathbf{f}_0$ be a classifier with expected loss $L(\mathbf{f}_0) = L^*$. Let the loss function $\ell = \ell_{\mathrm{LR}}$ defined in Section 3.4.1. Then the following two statements hold:*

1. *There exists a C_1 such that if*

$$n > C_1 \max\left(\frac{||\mathbf{f}_0||^2 \log(\frac{1}{\delta})}{\varepsilon_g^2}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||}{\varepsilon_g \varepsilon_p}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||^2}{\varepsilon_g^{3/2}\varepsilon_p}\right),$$

then the output $\mathbf{f}_{\mathrm{priv}}$ of Algorithm 1 satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\mathrm{priv}}) \leq L^* + \varepsilon_g\right) \geq 1 - \delta.$$

2. *There exists a C_2 such that if*

$$n > C \max\left(\frac{||\mathbf{f}_0||^2 \log(1/\delta)}{\varepsilon_g^2}, \frac{||\mathbf{f}_0||^2}{\varepsilon_g \varepsilon_p}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||}{\varepsilon_g \varepsilon_p}\right),$$

then the output $\mathbf{f}_{\mathrm{priv}}$ of Algorithm 2 with $c = \frac{1}{4}$ satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\mathrm{priv}}) \leq L^* + \varepsilon_g\right) \geq 1 - \delta.$$

Proof Since ℓ_{LR} is convex and doubly differentiable for any z_1, z_2,

$$\ell'_{\mathrm{LR}}(z_1) - \ell'_{\mathrm{LR}}(z_2) \leq \ell''_{\mathrm{LR}}(z^*)(z_1 - z_2)$$

for some $z^* \in [z_1, z_2]$. Moreover, $|\ell''_{\mathrm{LR}}(z^*)| \leq c = \frac{1}{4}$, so ℓ' is $\frac{1}{4}$-Lipschitz. The corollary now follows from Theorems 15 and 18. ■

For SVMs we state results with $\ell = \ell_{\mathrm{Huber}}$, but a similar bound can be shown for ℓ_s as well.

Corollary 21 (Huber Support Vector Machines) *Let training data $\mathcal{D}$ be generated i.i.d. according to a distribution P and let $\mathbf{f}_0$ be a classifier with expected loss $L(\mathbf{f}_0) = L^*$. Let the loss function $\ell = \ell_{\mathrm{Huber}}$ defined in* (7)*. Then the following two statements hold:*

1. *There exists a C_1 such that if*

$$n > C_1 \max\left(\frac{||\mathbf{f}_0||^2 \log(\frac{1}{\delta})}{\varepsilon_g^2}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||}{\varepsilon_g \varepsilon_p}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||^2}{h^{1/2}\varepsilon_g^{3/2}\varepsilon_p}\right),$$

then the output $\mathbf{f}_{\mathrm{priv}}$ of Algorithm 1 satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\mathrm{priv}}) \leq L^* + \varepsilon_g\right) \geq 1 - \delta.$$

2. *There exists a C_2 such that if*

$$n > C \max\left(\frac{||\mathbf{f}_0||^2 \log(1/\delta)}{\varepsilon_g^2}, \frac{||\mathbf{f}_0||^2}{h\varepsilon_g\varepsilon_p}, \frac{d\log(\frac{d}{\delta})||\mathbf{f}_0||}{\varepsilon_g\varepsilon_p}\right),$$

then the output $\mathbf{f}_{\text{priv}}$ of Algorithm 2 with $c = \frac{1}{4}$ satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\text{priv}}) \leq L^* + \varepsilon_g\right) \geq 1 - \delta.$$

Proof The Huber loss is convex and differentiable with continuous derivatives. Moreover, since the derivative of the Huber loss is piecewise linear with slope 0 or at most $\frac{1}{2h}$, for any z_1, z_2,

$$|\ell'_{\text{Huber}}(z_1) - \ell'_{\text{Huber}}(z_2)| \leq \frac{1}{2h}|z_1 - z_2|,$$

so ℓ'_{Huber} is $\frac{1}{2h}$-Lipschitz. The first part of the corollary follows from Theorem 15.

For the second part of the corollary, we observe that from Corollary 13, we do not need ℓ to be globally double differentiable, and the bound on $|\ell''(z)|$ in Theorem 18 is only needed to ensure that $\varepsilon'_p > 0$; since ℓ_{Huber} is double differentiable except in a set of Lebesgue measure 0, with $|\ell''_{\text{Huber}}(z)| \leq \frac{1}{2h}$, the corollary follows by an application of Theorem 18. ■

5. Kernel Methods

A powerful methodology in learning problems is the "kernel trick," which allows the efficient construction of a predictor $\mathbf{f}$ that lies in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ associated to a positive definite kernel function $k(\cdot,\cdot)$. The representer theorem (Kimeldorf and Wahba, 1970) shows that the regularized empirical risk in (1) is minimized by a function $\mathbf{f}(\mathbf{x})$ that is given by a linear combination of kernel functions centered at the data points:

$$\mathbf{f}^*(\mathbf{x}) = \sum_{i=1}^{n} a_i k(\mathbf{x}(i), \mathbf{x}). \tag{17}$$

This elegant result is important for both theoretical and computational reasons. Computationally, one releases the values a_i corresponding to the $\mathbf{f}$ that minimizes the empirical risk, along with the data points $\mathbf{x}(i)$; the user classifies a new $\mathbf{x}$ by evaluating the function in (17).

A crucial difficulty in terms of privacy is that this directly releases the private values $\mathbf{x}(i)$ of some individuals in the training set. Thus, even if the classifier is computed in a privacy-preserving way, any classifier released by this process requires revealing the data. We provide an algorithm that avoids this problem, using an approximation method (Rahimi and Recht, 2007, 2008b) to approximate the kernel function using random projections.

5.1 Mathematical Preliminaries

Our approach works for kernel functions which are translation invariant, so $k(\mathbf{x},\mathbf{x}') = k(\mathbf{x}-\mathbf{x}')$. The key idea in the random projection method is from Bochner's Theorem, which states that a

continuous translation invariant kernel is positive definite if and only if it is the Fourier transform of a nonnegative measure. This means that the Fourier transform $K(\theta)$ of translation-invariant kernel function $k(\mathbf{t})$ can be normalized so that $K(\theta) = K(\theta)/\|K(\theta)\|_1$ is a probability measure on the transform space Θ. We will assume $K(\theta)$ is uniformly bounded over θ.

In this representation

$$k(\mathbf{x},\mathbf{x}') = \int_\Theta \phi(\mathbf{x};\theta)\phi(\mathbf{x}';\theta)K(\theta)d\theta, \tag{18}$$

where we will assume the feature functions $\phi(\mathbf{x};\theta)$ are bounded:

$$|\phi(\mathbf{x};\theta)| \leq \zeta \qquad \forall \mathbf{x} \in \mathcal{X},\ \forall \theta \in \Theta.$$

A function $\mathbf{f} \in \mathcal{H}$ can be written as

$$\mathbf{f}(\mathbf{x}) = \int_\Theta a(\theta)\phi(\mathbf{x};\theta)K(\theta)d\theta.$$

To prove our generalization bounds we must show that bounded classifiers $\mathbf{f}$ induce bounded functions $a(\theta)$. Writing the evaluation functional as an inner product with $k(\mathbf{x},\mathbf{x}')$ and (18) shows

$$\mathbf{f}(\mathbf{x}) = \int_\Theta \left(\int_\mathcal{X} \mathbf{f}(\mathbf{x}')\phi(\mathbf{x}';\theta)d\mathbf{x}' \right) \phi(\mathbf{x};\theta)K(\theta)d\theta.$$

Thus we have

$$a(\theta) = \int_\mathcal{X} \mathbf{f}(\mathbf{x}')\phi(\mathbf{x}';\theta)d\mathbf{x}'$$
$$|a(\theta)| \leq \mathrm{Vol}(\mathcal{X}) \cdot \zeta \cdot \|\mathbf{f}\|_\infty .$$

This shows that $a(\theta)$ is bounded uniformly over Θ when $\mathbf{f}(\mathbf{x})$ is bounded uniformly over $\mathcal{X}$. The volume of the unit ball is $\mathrm{Vol}(\mathcal{X}) = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}$ (see Ball, 1997 for more details). For large d this is $(\sqrt{\frac{2\pi e}{d}})^d$ by Stirling's formula. Furthermore, we have

$$\|\mathbf{f}\|_\mathcal{H}^2 = \int_\Theta a(\theta)^2 K(\theta)d\theta.$$

5.2 A Reduction to the Linear Case

We now describe how to apply Algorithms 1 and 2 for classification with kernels, by transforming to linear classification. Given $\{\theta_j\}$, let $R : \mathcal{X} \to \mathbb{R}^D$ be the map that sends $\mathbf{x}(i)$ to a vector $\mathbf{v}(i) \in \mathbb{R}^D$ where $\mathbf{v}_j(i) = \phi(\mathbf{x}(i);\theta_j)$ for $j \in [D]$. We then use Algorithm 1 or Algorithm 2 to compute a privacy-preserving linear classifier $\mathbf{f}$ in $\mathbb{R}^D$. The algorithm releases R and $\tilde{\mathbf{f}}$. The overall classifier is $\mathbf{f}_{\mathrm{priv}}(\mathbf{x}) = \tilde{\mathbf{f}}(R(\mathbf{x}))$.

As an example, consider the Gaussian kernel

$$k(\mathbf{x},\mathbf{x}') = \exp\left(-\gamma \left\|\mathbf{x}-\mathbf{x}'\right\|_2^2\right).$$

The Fourier transform of a Gaussian is a Gaussian, so we can sample $\theta_j = (\omega,\psi)$ according to the distribution $\mathrm{Uniform}[-\pi,\pi] \times \mathcal{N}(0,2\gamma I_d)$ and compute $v_j = \cos(\omega^T\mathbf{x}+\psi)$. The random phase is used to produce a real-valued mapping. The paper of Rahimi and Recht (2008a) has more examples of transforms for other kernel functions.

Algorithm 3 Private ERM for nonlinear kernels

Inputs: Data $\{(\mathbf{x}_i, y_i) : i \in [n]\}$, positive definite kernel function $k(\cdot,\cdot)$, sampling function $K(\theta)$, parameters $\varepsilon_p, \Lambda, D$
Outputs: Predictor $\mathbf{f}_{\text{priv}}$ and pre-filter $\{\theta_j : j \in [D]\}$.
Draw $\{\theta_j : j = 1,2,\ldots,D\}$ iid according to $K(\theta)$.
Set $\mathbf{v}(i) = \sqrt{2/D}[\phi(\mathbf{x}(i);\theta_1)\cdots\phi(\mathbf{x}(i);\theta_D)]^T$ for each i.
Run Algorithm 1 or Algorithm 2 with data $\{(\mathbf{v}(i), y(i))\}$ and parameters ε_p, Λ.

5.3 Privacy Guarantees

Because the workhorse of Algorithm 3 is a differentially-private version of ERM for linear classifiers (either Algorithm 1 or Algorithm 2), and the points $\{\theta_j : j \in [D]\}$ are independent of the data, the privacy guarantees for Algorithm 3 follow trivially from Theorems 6 and 9.

Theorem 22 *Given data $\{(\mathbf{x}(i), y(i)) : i = 1,2,\ldots,n\}$ with $(\mathbf{x}(i), y(i))$ and $\|\mathbf{x}(i)\| \leq 1$, the outputs $(\mathbf{f}_{\text{priv}}, \{\theta_j : j \in [D]\})$ of Algorithm 3 guarantee ε_p-differential privacy.*

The proof trivially follows from a combination of Theorems 6, 9, and the fact that the θ_j's are drawn independently of the input data set.

5.4 Generalization Performance

We now turn to generalization bounds for Algorithm 3. We will prove results using objective perturbation (Algorithm 2) in Algorithm 3, but analogous results for output perturbation (Algorithm 1) are simple to prove. Our comparisons will be against arbitrary predictors $\mathbf{f}_0$ whose norm is bounded in some sense. That is, given an $\mathbf{f}_0$ with some properties, we will choose regularization parameter Λ, dimension D, and number of samples n so that the predictor $\mathbf{f}_{\text{priv}}$ has expected loss close to that of $\mathbf{f}_0$.

In this section we will assume $N(\mathbf{f}) = \frac{1}{2}\|\mathbf{f}\|^2$ so that $N(\cdot)$ is 1-strongly convex, and that the loss function ℓ is convex, differentiable and $|\ell'(z)| \leq 1$ for all z.

Our first generalization result is the simplest, since it assumes a strong condition that gives easy guarantees on the projections. We would like the predictor produced by Algorithm 3 to be competitive against an $\mathbf{f}_0$ such that

$$\mathbf{f}_0(\mathbf{x}) = \int_\Theta a_0(\theta)\phi(\mathbf{x};\theta)K(\theta)d\theta, \tag{19}$$

and $|a_0(\theta)| \leq C$ (see Rahimi and Recht, 2008b). Our first result provides the technical building block for our other generalization results. The proof makes use of ideas from Rahimi and Recht (2008b) and techniques from Sridharan et al. (2008); Shalev-Shwartz and Srebro (2008).

Lemma 23 *Let $\mathbf{f}_0$ be a predictor such that $|a_0(\theta)| \leq C$, for all θ, where $a_0(\theta)$ is given by (19), and suppose $L(\mathbf{f}_0) = L^*$. Moreover, suppose that $\ell'(\cdot)$ is c-Lipschitz. If the data $\mathcal{D}$ is drawn i.i.d. according to P, then there exists a constant C_0 such that if*

$$n > C_0 \cdot \max\left(\frac{C^2\sqrt{\log(1/\delta)}}{\varepsilon_p\varepsilon_g^2}\cdot\log\frac{C\log(1/\delta)}{\varepsilon_g\delta}, \frac{c\varepsilon_g}{\varepsilon_p\log(1/\delta)}\right), \tag{20}$$

then Λ and D can be chosen such that the output $\mathbf{f}_{\text{priv}}$ of Algorithm 3 using Algorithm 2 satisfies

$$\mathbb{P}\left(L(\mathbf{f}_{\text{priv}}) - L^* \leq \varepsilon_g\right) \geq 1 - 4\delta.$$

Proof Since $|a_0(\theta)| \leq C$ and the $K(\theta)$ is bounded, we have (Rahimi and Recht, 2008b, Theorem 1) that with probability $1-2\delta$ there exists an $\mathbf{f}_p \in \mathbb{R}^D$ such that

$$L(\mathbf{f}_p) \leq L(\mathbf{f}_0) + O\left(\left(\frac{1}{\sqrt{n}} + \frac{1}{\sqrt{D}}\right) C\sqrt{\log\frac{1}{\delta}}\right), \tag{21}$$

We will choose D to make this loss small. Furthermore, $\mathbf{f}_p$ is guaranteed to have $\|\mathbf{f}_p\|_\infty \leq C/D$, so

$$\|\mathbf{f}_p\|_2^2 \leq \frac{C^2}{D}. \tag{22}$$

Now given such an $\mathbf{f}_p$ we must show that $\mathbf{f}_{\text{priv}}$ will have true risk close to that of $\mathbf{f}_p$ as long as there are enough data points. This can be shown using the techniques in Shalev-Shwartz and Srebro (2008). Let

$$J(\mathbf{f}) = L(\mathbf{f}) + \frac{\Lambda}{2}\|\mathbf{f}\|_2^2,$$

and let

$$\mathbf{f}_{\text{rtr}} = \operatorname*{argmin}_{\mathbf{f}\in\mathbb{R}^D} J(\mathbf{f})$$

minimize the regularized true risk. Then

$$J(\mathbf{f}_{\text{priv}}) = J(\mathbf{f}_p) + (J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}})) + (J(\mathbf{f}_{\text{rtr}}) - J(\mathbf{f}_p)).$$

Now, since $J(\cdot)$ is minimized by $\mathbf{f}_{\text{rtr}}$, the last term is negative and we can disregard it. Then we have

$$L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_p) \leq (J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}})) + \frac{\Lambda}{2}\|\mathbf{f}_p\|_2^2 - \frac{\Lambda}{2}\left\|\mathbf{f}_{\text{priv}}\right\|_2^2. \tag{23}$$

From Lemma 19, with probability at least $1-\delta$ over the noise $\mathbf{b}$,

$$J(\mathbf{f}_{\text{priv}}) - J\left(\operatorname*{argmin}_{\mathbf{f}} J(\mathbf{f})\right) \leq \frac{4D^2\log^2(D/\delta)}{\Lambda n^2\varepsilon_p^2}. \tag{24}$$

Now we can bound the term $(J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}}))$ by twice the gap in the regularized empirical risk difference (24) plus an additional term (Sridharan et al., 2008, Corollary 2). That is, with probability $1-\delta$:

$$J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}}) \leq 2(J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}})) + O\left(\frac{\log(1/\delta)}{\Lambda n}\right). \tag{25}$$

If we set $n > \frac{c}{4\varepsilon_p\Lambda}$, then $\varepsilon_p' > 0$, and we can plug Lemma 19 into (25) to obtain:

$$J(\mathbf{f}_{\text{priv}}) - J(\mathbf{f}_{\text{rtr}}) \leq \frac{8D^2\log^2(D/\delta)}{\Lambda n^2\varepsilon_p^2} + O\left(\frac{\log(1/\delta)}{\Lambda n}\right). \tag{26}$$

Plugging (26) into (23), discarding the negative term involving $\left\|\mathbf{f}_{\text{priv}}\right\|_2^2$ and setting $\Lambda = \varepsilon_g / \|\mathbf{f}_p\|^2$ gives

$$L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_p) \leq \frac{8\|\mathbf{f}_p\|_2^2 D^2 \log^2(D/\delta)}{n^2\varepsilon_p^2\varepsilon_g} + O\left(\frac{\|\mathbf{f}_p\|_2^2 \log\frac{1}{\delta}}{n\varepsilon_g}\right) + \frac{\varepsilon_g}{2}. \tag{27}$$

Now we have, using (21) and (27), that with probability $1-4\delta$:

$$\begin{aligned} L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_0) &\leq (L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_p)) + (L(\mathbf{f}_p) - L(\mathbf{f}_0)) \\ &\leq \frac{8\|\mathbf{f}_p\|_2^2 D^2 \log^2(D/\delta)}{n^2\varepsilon_p^2\varepsilon_g} + O\left(\frac{\|\mathbf{f}_p\|_2^2 \log(1/\delta)}{n\varepsilon_g}\right) + \frac{\varepsilon_g}{2} \\ &\quad + O\left(\left(\frac{1}{\sqrt{n}} + \frac{1}{\sqrt{D}}\right) C\sqrt{\log\frac{1}{\delta}}\right), \end{aligned}$$

Substituting (22), we have

$$\begin{aligned} L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_0) &\leq \frac{8C^2 D \log^2(D/\delta)}{n^2\varepsilon_p^2\varepsilon_g} + O\left(\frac{C^2\log(1/\delta)}{Dn\varepsilon_g}\right) + \frac{\varepsilon_g}{2} \\ &\quad + O\left(\left(\frac{1}{\sqrt{n}} + \frac{1}{\sqrt{D}}\right) C\sqrt{\log\frac{1}{\delta}}\right). \end{aligned}$$

To set the remaining parameters, we will choose $D < n$ so that

$$L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_0) \leq \frac{8C^2 D \log^2(D/\delta)}{n^2\varepsilon_p^2\varepsilon_g} + O\left(\frac{C^2\log(1/\delta)}{Dn\varepsilon_g}\right) + \frac{\varepsilon_g}{2} + O\left(\frac{C\sqrt{\log(1/\delta)}}{\sqrt{D}}\right).$$

We set $D = O(C^2\log(1/\delta)/\varepsilon_g^2)$ to make the last term $\varepsilon_g/6$, and:

$$L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_0) \leq O\left(\frac{C^4 \log\frac{1}{\delta} \log^2 \frac{C^2\log(1/\delta)}{\varepsilon_g^2\delta}}{n^2\varepsilon_p^2\varepsilon_g^3}\right) + O\left(\frac{\varepsilon_g}{n}\right) + \frac{2\varepsilon_g}{3}.$$

Setting n as in (20) proves the result. Moreover, setting $n > \frac{c\left\|\mathbf{f}_p\right\|^2}{4\varepsilon_p\varepsilon_g} = C_0 \cdot \frac{c\varepsilon_g}{\varepsilon_p \log(1/\delta)}$ ensures that $n > \frac{c}{4\Lambda\varepsilon_p}$. ■

We can adapt the proof procedure to show that Algorithm 3 is competitive against any classifier $\mathbf{f}_0$ with a given bound on $\|\mathbf{f}_0\|_\infty$. It can be shown that for some constant ζ that $|a_0(\theta)| \leq \text{Vol}(\mathcal{X})\zeta\|\mathbf{f}_0\|_\infty$. Then we can set this as C in (20) to obtain the following result.

Theorem 24 *Let $\mathbf{f}_0$ be a classifier with norm $\|\mathbf{f}_0\|_\infty$, and let $\ell'(\cdot)$ be c-Lipschitz. Then for any distribution P, there exists a constant C_0 such that if*

$$n > C_0 \cdot \max\left(\frac{\|\mathbf{f}_0\|_\infty^2 \zeta^2 (\text{Vol}(\mathcal{X}))^2 \sqrt{\log(1/\delta)}}{\varepsilon_p\varepsilon_g^2} \cdot \log\frac{\|\mathbf{f}_0\|_\infty \text{Vol}(\mathcal{X})\zeta\log(1/\delta)}{\varepsilon_g\delta\Gamma(\frac{d}{2}+1)}, \frac{c\varepsilon_g}{\varepsilon_p\log(1/\delta)}\right), \tag{28}$$

then Λ and D can be chosen such that the output $\mathbf{f}_{\text{priv}}$ of Algorithm 3 with Algorithm 2 satisfies $\mathbb{P}\left(L(\mathbf{f}_{\text{priv}}) - L(\mathbf{f}_0) \leq \varepsilon_g\right) \geq 1 - 4\delta$.

Proof Substituting $C = \mathrm{Vol}(\mathcal{X})\zeta\|\mathbf{f}_0\|_\infty$ in Lemma 23 we get the result. ■

We can also derive a generalization result with respect to classifiers with bounded $\|\mathbf{f}_0\|_{\mathcal{H}}$.

Theorem 25 *Let $\mathbf{f}_0$ be a classifier with norm $\|\mathbf{f}_0\|_{\mathcal{H}}$, and let ℓ' be c-Lipschitz. Then for any distribution P, there exists a constant C_0 such that if,*

$$n = C_0 \cdot \max\left(\frac{\|\mathbf{f}_0\|_{\mathcal{H}}^4 \zeta^2 (\mathrm{Vol}(\mathcal{X}))^2 \sqrt{\log(1/\delta)}}{\varepsilon_p \varepsilon_g^4} \cdot \log \frac{\|\mathbf{f}_0\|_{\mathcal{H}} \mathrm{Vol}(\mathcal{X})\zeta \log(1/\delta)}{\varepsilon_g \delta \Gamma(\frac{d}{2}+1)}, \frac{c\varepsilon_g}{\varepsilon_p \log(1/\delta)} \right),$$

then Λ and D can be chosen such that the output of Algorithm 3 run with Algorithm 2 satisfies $\mathbb{P}\left(L(\mathbf{f}_{\mathrm{priv}}) - L(\mathbf{f}_0) \leq \varepsilon_g\right) \geq 1 - 4\delta$.

Proof Let $\mathbf{f}_0$ be a classifier with norm $\|\mathbf{f}_0\|_{\mathcal{H}}^2$ and expected loss $L(\mathbf{f}_0)$. Now consider

$$\mathbf{f}_{\mathrm{rtr}} = \operatorname*{argmin}_{\mathbf{f}} L(\mathbf{f}) + \frac{\Lambda_{\mathrm{rtr}}}{2}\|\mathbf{f}\|_{\mathcal{H}}^2,$$

for some Λ_{rtr} to be specified later. We will first need a bound on $\|\mathbf{f}_{\mathrm{rtr}}\|_\infty$ in order to use our previous sample complexity results. Since $\mathbf{f}_{\mathrm{rtr}}$ is a minimizer, we can take the derivative of the regularized expected loss and set it to 0 to get:

$$\begin{aligned}
\mathbf{f}_{\mathrm{rtr}}(\mathbf{x}') &= \frac{-1}{\Lambda_{\mathrm{rtr}}} \left(\frac{\partial}{\partial \mathbf{f}} \int_{\mathcal{X}} \ell(\mathbf{f}(\mathbf{x}'), y) dP(\mathbf{x}, y) \right) \\
&= \frac{-1}{\Lambda_{\mathrm{rtr}}} \left(\int_{\mathcal{X}} \left(\frac{\partial}{\partial \mathbf{f}(\mathbf{x}')} \ell(\mathbf{f}(\mathbf{x}), y) \right) \cdot \left(\frac{\partial}{\partial \mathbf{f}(\mathbf{x}')} \mathbf{f}(\mathbf{x}) \right) dP(\mathbf{x}, y) \right),
\end{aligned}$$

where $P(\mathbf{x}, y)$ is a distribution on pairs $(\mathbf{x}, y)$. Now, using the representer theorem, $\frac{\partial}{\partial \mathbf{f}(\mathbf{x}')}\mathbf{f}(\mathbf{x}) = k(\mathbf{x}', \mathbf{x})$. Since the kernel function is bounded and the derivative of the loss is always upper bounded by 1, so the integrand can be upper bounded by a constant. Since $P(\mathbf{x}, y)$ is a probability distribution, we have for all $\mathbf{x}'$ that $|\mathbf{f}_{\mathrm{rtr}}(\mathbf{x}')| = O(1/\Lambda_{\mathrm{rtr}})$. Now we set $\Lambda_{\mathrm{rtr}} = \varepsilon_g / \|\mathbf{f}_0\|_{\mathcal{H}}^2$ to get

$$\|\mathbf{f}_{\mathrm{rtr}}\|_\infty = O\left(\frac{\|\mathbf{f}_0\|_{\mathcal{H}}^2}{\varepsilon_g} \right).$$

We now have two cases to consider, depending on whether $L(\mathbf{f}_0) < L(\mathbf{f}_{\mathrm{rtr}})$ or $L(\mathbf{f}_0) > L(\mathbf{f}_{\mathrm{rtr}})$.

Case 1: Suppose that $L(\mathbf{f}_0) < L(\mathbf{f}_{\mathrm{rtr}})$. Then by the definition of $\mathbf{f}_{\mathrm{rtr}}$,

$$L(\mathbf{f}_{\mathrm{rtr}}) + \frac{\varepsilon_g}{2} \cdot \frac{\|\mathbf{f}_{\mathrm{rtr}}\|_{\mathcal{H}}^2}{\|\mathbf{f}_0\|_{\mathcal{H}}^2} \leq L(\mathbf{f}_0) + \frac{\varepsilon_g}{2}.$$

Since $\frac{\varepsilon_g}{2} \cdot \frac{\|\mathbf{f}_{\mathrm{rtr}}\|_{\mathcal{H}}^2}{\|\mathbf{f}_0\|_{\mathcal{H}}^2} \geq 0$, we have $L(\mathbf{f}_{\mathrm{rtr}}) - L(\mathbf{f}_0) \leq \frac{\varepsilon_g}{2}$.

Case 2: Suppose that $L(\mathbf{f}_0) > L(\mathbf{f}_{\mathrm{rtr}})$. Then the regularized classifier has better generalization performance than the original, so we have trivially that $L(\mathbf{f}_{\mathrm{rtr}}) - L(\mathbf{f}_0) \leq \frac{\varepsilon_g}{2}$.

Therefore in both cases we have a bound on $\|\mathbf{f}_{\mathrm{rtr}}\|_\infty$ and a generalization gap of $\varepsilon_g/2$. We can now apply Theorem 24 to show that for n satisfying (28) we have

$$\mathbb{P}\left(L(\mathbf{f}_{\mathrm{priv}}) - L(\mathbf{f}_0) \leq \varepsilon_g\right) \geq 1 - 4\delta.$$

■

6. Parameter Tuning

The privacy-preserving learning algorithms presented so far in this paper assume that the regularization constant Λ is provided as an input, and is independent of the data. In actual applications of ERM, Λ is selected based on the data itself. In this section, we address this issue: how to design an ERM algorithm with end-to-end privacy, which selects Λ based on the data itself.

Our solution is to present a privacy-preserving parameter tuning technique that is applicable in general machine learning algorithms, beyond ERM. In practice, one typically tunes parameters (such as the regularization parameter Λ) as follows: using data held out for validation, train predictors $\mathbf{f}(\cdot;\Lambda)$ for multiple values of Λ, and select the one which provides the best empirical performance. However, even though the output of an algorithm preserves ε_p-differential privacy for a fixed Λ (as is the case with Algorithms 1 and 2), by choosing a Λ based on empirical performance on a validation set may violate ε_p-differential privacy guarantees. That is, if the procedure that picks Λ is not private, then an adversary may use the released classifier to infer the value of Λ and therefore something about the values in the database.

We suggest two ways of resolving this issue. First, if we have access to a smaller publicly available data from the same distribution, then we can use this as a holdout set to tune Λ. This Λ can be subsequently used to train a classifier on the private data. Since the value of Λ does not depend on the values in the private data set, this procedure will still preserve the privacy of individuals in the private data.

If no such public data is available, then we need a differentially private tuning procedure. We provide such a procedure below. The main idea is to train for different values of Λ on separate subsets of the training data set, so that the total training procedure still maintains ε_p-differential privacy. We score each of these predictors on a validation set, and choose a Λ (and hence $\mathbf{f}(\cdot;\Lambda)$) using a randomized privacy-preserving comparison procedure (McSherry and Talwar, 2007). The last step is needed to guarantee ε_p-differential privacy for individuals in the validation set. This final algorithm provides an end-to-end guarantee of differential privacy, and renders our privacy-preserving ERM procedure complete. We observe that both these procedures can be used for tuning multiple parameters as well.

6.1 Tuning Algorithm

Algorithm 4 Privacy-preserving parameter tuning

Inputs: Database $\mathcal{D}$, parameters $\{\Lambda_1,\ldots,\Lambda_m\}$, ε_p.

Outputs: Parameter $\mathbf{f}_{\text{priv}}$.

Divide $\mathcal{D}$ into $m+1$ equal portions $\mathcal{D}_1,\ldots,\mathcal{D}_{m+1}$, each of size $\frac{|\mathcal{D}|}{m+1}$.

For each $i = 1,2,\ldots,m$, apply a privacy-preserving learning algorithm (for example Algorithms 1, 2, or 3) on $\mathcal{D}_i$ with parameter Λ_i and ε_p to get output $\mathbf{f}_i$.

Evaluate z_i, the number of mistakes made by $\mathbf{f}_i$ on $\mathcal{D}_{m+1}$. Set $\mathbf{f}_{\text{priv}} = \mathbf{f}_i$ with probability

$$q_i = \frac{e^{-\varepsilon_p z_i/2}}{\sum_{i=1}^{m} e^{-\varepsilon_p z_i/2}}.$$

We note that the list of potential Λ values input to this procedure should not be a function of the private data set. It can be shown that the empirical error on $\mathcal{D}_{m+1}$ of the classifier output by this procedure is close to the empirical error of the best classifier in the set $\{\mathbf{f}_1,\ldots,\mathbf{f}_m\}$ on $\mathcal{D}_{m+1}$, provided $|\mathcal{D}|$ is high enough.

6.2 Privacy and Utility

Theorem 26 *The output of the tuning procedure of Algorithm 4 is ε_p-differentially private.*

Proof To show that Algorithm 4 preserves ε_p-differential privacy, we first consider an alternative procedure $\mathcal{M}$. Let $\mathcal{M}$ be the procedure that releases the values $(\mathbf{f}_1,\ldots,\mathbf{f}_m,i)$ where, $\mathbf{f}_1,\ldots,\mathbf{f}_m$ are the intermediate values computed in the second step of Algorithm 4, and i is the index selected by the exponential mechanism step. We first show that $\mathcal{M}$ preserves ε_p-differential privacy.

Let $\mathcal{D}$ and $\mathcal{D}'$ be two data sets that differ in the value of one individual such that $\mathcal{D}=\mathcal{D}\cup\{(\mathbf{x},y)\}$, and $\mathcal{D}'=\mathcal{D}\cup\{(\mathbf{x}',y')\}$.

Recall that the data sets $\mathcal{D}_1,\ldots,\mathcal{D}_{m+1}$ are disjoint; moreover, the randomness in the privacy mechanisms are independent. Therefore,

$$\begin{aligned}
&\mathbb{P}\left(\mathbf{f}_1\in\mathcal{S}_1,\ldots,\mathbf{f}_m\in\mathcal{S}_m,i=i^*|\mathcal{D}\right)\\
&\qquad=\int_{\mathcal{S}_1\times\ldots\mathcal{S}_m}\mathbb{P}\left(i=i^*|\mathbf{f}_1,\ldots,\mathbf{f}_m,\mathcal{D}_{m+1}\right)\mu(\mathbf{f}_1,\ldots,\mathbf{f}_m|\mathcal{D})d\mathbf{f}_1\cdots d\mathbf{f}_m\\
&\qquad=\int_{\mathcal{S}_1\times\ldots\mathcal{S}_m}\mathbb{P}\left(i=i^*|\mathbf{f}_1,\ldots,\mathbf{f}_m,\mathcal{D}_{m+1}\right)\prod_{j=1}^{m}\mu_j(\mathbf{f}_j|\mathcal{D}_j)d\mathbf{f}_1\cdots d\mathbf{f}_m, \qquad (29)
\end{aligned}$$

where $\mu_j(\mathbf{f})$ is the density at $\mathbf{f}$ induced by the classifier run with parameter Λ_j, and $\mu(\mathbf{f}_1,\ldots,\mathbf{f}_m)$ is the joint density at $\mathbf{f}_1,\ldots,\mathbf{f}_m$, induced by $\mathcal{M}$. Now suppose that $(\mathbf{x},y)\in\mathcal{D}_j$, for $j=m+1$. Then, $\mathcal{D}_k=\mathcal{D}'_k$, and $\mu_j(\mathbf{f}_j|\mathcal{D}_j)=\mu_j(\mathbf{f}_j|\mathcal{D}'_j)$, for $k\in[m]$. Moreover, given any fixed set $\mathbf{f}_1,\ldots,\mathbf{f}_m$,

$$\mathbb{P}\left(i=i^*|\mathcal{D}'_{m+1},\mathbf{f}_1,\ldots,\mathbf{f}_m\right)\leq e^{\varepsilon_p}\mathbb{P}\left(i=i^*|\mathcal{D}_{m+1},\mathbf{f}_1,\ldots,\mathbf{f}_m\right). \qquad (30)$$

Instead, if $(\mathbf{x},y)\in\mathcal{D}_j$, for $j\in[m]$, then, $\mathcal{D}_k=\mathcal{D}'_k$, for $k\in[m+1],k\neq j$. Thus, for a fixed $\mathbf{f}_1,\ldots,\mathbf{f}_m$,

$$\mathbb{P}\left(i=i^*|\mathcal{D}'_{m+1},\mathbf{f}_1,\ldots,\mathbf{f}_m\right)=\mathbb{P}\left(i=i^*|\mathcal{D}_{m+1},\mathbf{f}_1,\ldots,\mathbf{f}_m\right), \qquad (31)$$

$$\mu_k(\mathbf{f}_k|\mathcal{D}_k)\leq e^{\varepsilon_p}\mu_k(\mathbf{f}_k|\mathcal{D}'_k). \qquad (32)$$

The lemma follows by combining (29)-(32).

Now, an adversary who has access to the output of $\mathcal{M}$ can compute the output of Algorithm 4 itself, without any further access to the data set. Therefore, by a simulatibility argument, as in Dwork et al. (2006b), Algorithm 4 also preserves ε_p-differential privacy. ■

In the theorem above, we assume that the individual algorithms for privacy-preserving classification satisfy Definition 2; a similar theorem can also be shown when they satisfy a guarantee as in Corollary 13.

The following theorem shows that the empirical error on $\mathcal{D}_{K+1}$ of the classifier output by the tuning procedure is close to the empirical error of the best classifier in the set $\{\mathbf{f}_1,\ldots,\mathbf{f}_K\}$. The proof of this Theorem follows from Lemma 7 of McSherry and Talwar (2007).

Theorem 27 *Let $z_{\min} = \min_i z_i$, and let z be the number of mistakes made on D_{m+1} by the classifier output by our tuning procedure. Then, with probability $1-\delta$,*

$$z \leq z_{\min} + \frac{2\log(m/\delta)}{\varepsilon_p}.$$

Proof In the notation of McSherry and Talwar (2007), the $z_{\min} = OPT$, the base measure μ is uniform on $[m]$, and $S_t = \{i : z_i < z_{\min} + t\}$. Their Lemma 7 shows that

$$\mathbb{P}\left(S_{2t}\right) \leq \frac{\exp(-\varepsilon_p t)}{\mu(S_t)},$$

where μ is the uniform measure on $[m]$. Using $\min \mu(S_t) = \frac{1}{m}$ to upper bound the right side and setting it equal to δ we obtain

$$t = \frac{1}{\varepsilon_p} \log \frac{m}{\delta}.$$

From this we have

$$\mathbb{P}\left(z \geq z_{min} + \frac{2}{\varepsilon_p} \log \frac{m}{\delta}\right) \leq \delta,$$

and the result follows. ■

7. Experiments

In this section we give experimental results for training linear classifiers with Algorithms 1 and 2 on two real data sets. Imposing privacy requirements necessarily degrades classifier performance. Our experiments show that provided there is sufficient data, objective perturbation (Algorithm 2) typically outperforms the sensitivity method (1) significantly, and achieves error rate close to that of the analogous non-private ERM algorithm. We first demonstrate how the accuracy of the classification algorithms vary with ε_p, the privacy requirement. We then show how the performance of privacy-preserving classification varies with increasing training data size.

The first data set we consider is the `Adult` data set from the UCI Machine Learning Repository (Asuncion and Newman, 2007). This moderately-sized data set contains demographic information about approximately $47,000$ individuals, and the classification task is to predict whether the annual income of an individual is below or above $\$50{,}000$, based on variables such as age, sex, occupation, and education. For our experiments, the average fraction of positive labels is about 0.25; therefore, a trivial classifier that always predicts -1 will achieve this error-rate, and only error-rates below 0.25 are interesting.

The second data set we consider is the `KDDCup99` data set (Hettich and Bay, 1999); the task here is to predict whether a network connection is a denial-of-service attack or not, based on several attributes. The data set includes about 5,000,000 instances. For this data the average fraction of positive labels is 0.20.

In order to implement the convex minimization procedure, we use the convex optimization library provided by Okazaki (2009).

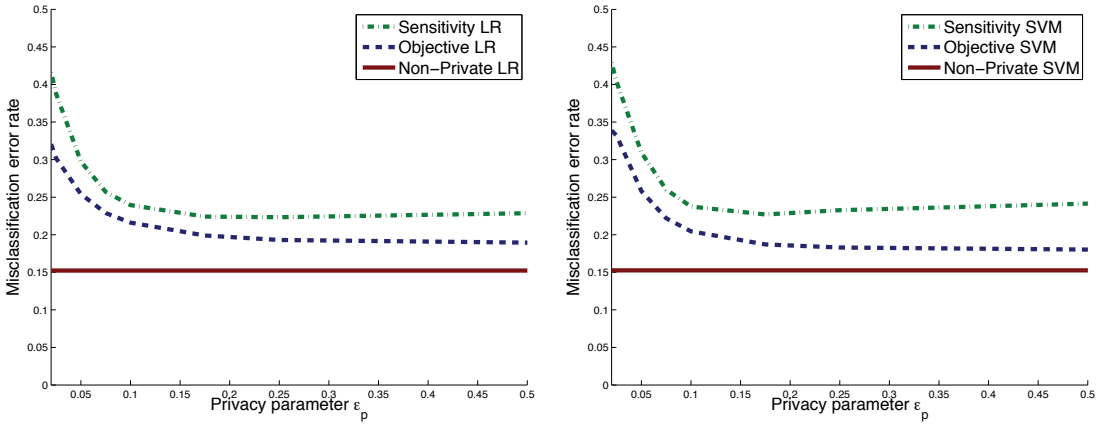
7.1 Preprocessing

In order to process the `Adult` data set into a form amenable for classification, we removed all entries with missing values, and converted each categorial attribute to a binary vector. For example, an attribute such as `(Male,Female)` was converted into 2 binary features. Each column was normalized to ensure that the maximum value is 1, and then each row is normalized to ensure that the norm of any example is at most 1. After preprocessing, each example was represented by a 105-dimensional vector, of norm at most 1.

For the `KDDCup99` data set, the instances were preprocessed by converting each categorial attribute to a binary vector. Each column was normalized to ensure that the maximum value is 1, and finally, each row was normalized, to ensure that the norm of any example is at most 1. After preprocessing, each example was represented by a 119-dimensional vector, of norm at most 1.

7.2 Privacy-Accuracy Tradeoff

For our first set of experiments, we study the tradeoff between the privacy requirement on the classifier, and its classification accuracy, when the classifier is trained on data of a fixed size. The privacy requirement is quantified by the value of ε_p; increasing ε_p implies a higher change in the belief of the adversary when one entry in $\mathcal{D}$ changes, and thus lower privacy. To measure accuracy, we use classification (test) error; namely, the fraction of times the classifier predicts a label with the wrong sign.



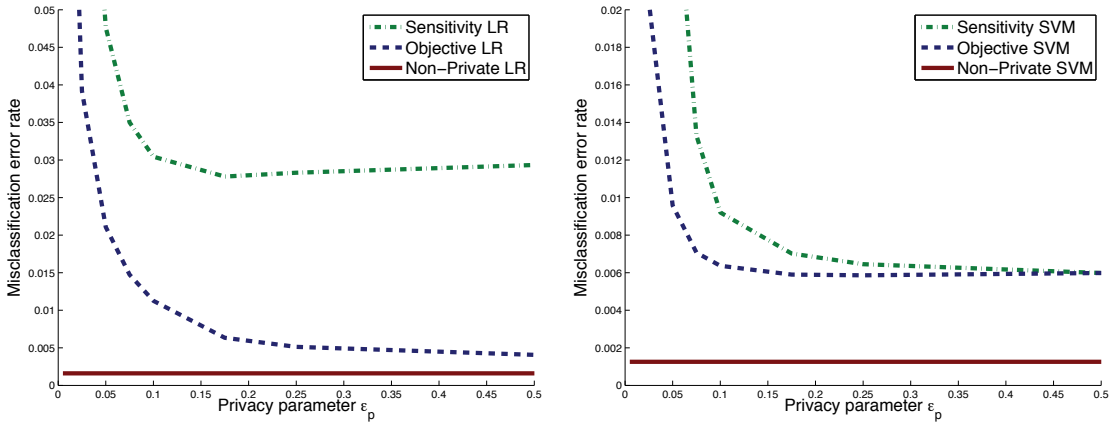
(a) Regularized logistic regression, `Adult`

(b) Regularized SVM, `Adult`

Figure 2: Privacy-Accuracy trade-off for the `Adult` data set

To study the privacy-accuracy tradeoff, we compare objective perturbation with the sensitivity method for logistic regression and Huber SVM. For Huber SVM, we picked the Huber constant $h = 0.5$, a typical value (Chapelle, 2007).[1] For each data set we trained classifiers for a few fixed values of Λ and tested the error of these classifiers. For each algorithm we chose the value of Λ that minimizes the error-rate for $\varepsilon_p = 0.1$.[2] We then plotted the error-rate against ε_p for the chosen value of Λ. The results are shown in Figures 2 and 3 for both logistic regression and support vector

1. Chapelle (2007) recommends using h between 0.01 and 0.5; we use $h = 0.5$ as we found that a higher value typically leads to more numerical stability, as well as better performance for both privacy-preserving methods.
2. For `KDDCup99` the error of the non-private algorithms did not increase with decreasing Λ.



(a) Regularized logistic regression, `KDDCup99`

(b) Regularized SVM, `KDDCup99`

Figure 3: Privacy-Accuracy trade-off for the `KDDCup99` data set

Λ	$10^{-10.0}$	$10^{-7.0}$	$10^{-4.0}$	$10^{-3.5}$	$10^{-3.0}$	$10^{-2.5}$	$10^{-2.0}$	$10^{-1.5}$
Logistic								
Non-Private	0.1540	**0.1533**	0.1654	0.1694	0.1758	0.1895	0.2322	0.2478
Output	0.5318	0.5318	0.5175	0.4928	0.4310	0.3163	**0.2395**	0.2456
Objective	0.8248	0.8248	0.8248	0.2694	0.2369	**0.2161**	0.2305	0.2475
Huber								
Non-Private	0.1527	**0.1521**	0.1632	0.1669	0.1719	0.1793	0.2454	0.2478
Output	0.5318	0.5318	0.5211	0.5011	0.4464	0.3352	**0.2376**	0.2476
Objective	0.2585	0.2585	0.2585	0.2582	0.2559	**0.2046**	0.2319	0.2478

Table 1: Error for different regularization parameters on `Adult` for $\varepsilon_p = 0.1$. The best error per algorithm is in bold.

machines.[3] The optimal values of Λ are shown in Tables 1 and 2. For non-private logistic regression and SVM, each presented error-rate is an average over 10-fold cross-validation; for the sensitivity method as well as objective perturbation, the presented error-rate is an average over 10-fold cross-validation and 50 runs of the randomized training procedure. For `Adult`, the privacy-accuracy tradeoff is computed over the entire data set, which consists of $45,220$ examples; for `KDDCup99` we use a randomly chosen subset of $70,000$ examples.

For the `Adult` data set, the constant classifier that classifies all examples to be negative acheives a classification error of about 0.25. The sensitivity method thus does slightly better than this constant classifier for most values of ε_p for both logistic regression and support vector machines. Objective perturbation outperforms sensitivity, and objective perturbation for support vector machines achieves lower classification error than objective perturbation for logistic regression. Non-private logistic regression and support vector machines both have classification error about 0.15.

3. The slight kink in the SVM curve on `Adult` is due to a switch to the second phase of the algorithm.

Λ	$10^{-9.0}$	$10^{-7.0}$	$10^{-5.0}$	$10^{-3.5}$	$10^{-3.0}$	$10^{-2.5}$	$10^{-2.0}$	$10^{-1.5}$
Logistic								
Non-Private	0.0016	**0.0016**	0.0021	0.0038	0.0037	0.0037	0.0325	0.0594
Output	0.5245	0.5245	0.5093	0.3518	0.1114	0.0359	**0.0304**	0.0678
Objective	0.2084	0.2084	0.2084	0.0196	0.0118	**0.0113**	0.0285	0.0591
Huber								
Non-Private	**0.0013**	0.0013	0.0013	0.0029	0.0051	0.0056	0.0061	0.0163
Output	0.5245	0.5245	0.5229	0.4611	0.3353	0.0590	**0.0092**	0.0179
Objective	0.0191	0.0191	0.0191	0.1827	0.0123	0.0066	**0.0064**	0.0157

Table 2: Error for different regularization parameters on KDDCup99 for $\varepsilon_p = 0.1$. The best error per algorithm is in bold.

For the KDDCup99 data set, the constant classifier that classifies all examples as negative, has error 0.19. Again, objective perturbation outperforms sensitivity for both logistic regression and support vector machines; however, for SVM and high values of ε_p (low privacy), the sensitivity method performs almost as well as objective perturbation. In the low privacy regime, logistic regression under objective perturbation is better than support vector machines. In contrast, in the high privacy regime (low ε_p), support vector machines with objective perturbation outperform logistic regression. For this data set, non-private logistic regression and support vector machines both have a classification error of about 0.001.

For SVMs on both Adult and KDDCup99, for large ε_p (0.25 onwards), the error of either of the private methods can increase slightly with increasing ε_p. This seems counterintuitive, but appears to be due the imbalance in fraction of the two labels. As the labels are imbalanced, the optimal classifier is trained to perform better on the negative labels than the positives. As ε_p increases, for a fixed training data size, so does the perturbation from the optimal classifier, induced by either of the private methods. Thus, as the perturbation increases, the number of false positives increases, whereas the number of false negatives decreases (as we verified by measuring the average false positive and false negative rates of the private classifiers). Therefore, the total error may increase slightly with decreasing privacy.

7.3 Accuracy vs. Training Data Size Tradeoffs

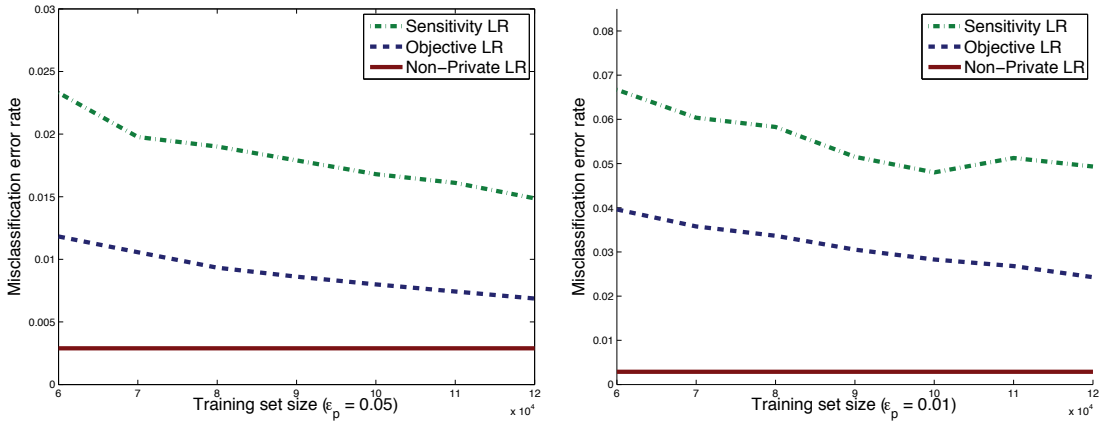
Next we examine how classification accuracy varies as we increase the size of the training set. We measure classification accuracy as the accuracy of the classifier produced by the tuning procedure in Section 6. As the Adult data set is not sufficiently large to allow us to do privacy-preserving tuning, for these experiments, we restrict our attention to the KDDCup99 data set.

Figures 4 and 5 present the learning curves for objective perturbation, non-private ERM and the sensitivity method for logistic loss and Huber loss, respectively. Experiments are shown for $\varepsilon_p = 0.01$ and $\varepsilon_p = 0.05$ for both loss functions. The training sets (for each of 5 values of Λ) are chosen to be of size $n = 60{,}000$ to $n = 120{,}000$, and the validation and test sets each are of size $25{,}000$. Each presented value is an average over 5 random permutations of the data, and 50 runs

of the randomized classification procedure. For objective perturbation we performed experiment in the regime when $\varepsilon'_p > 0$, so $\Delta = 0$ in Algorithm 2.[4]

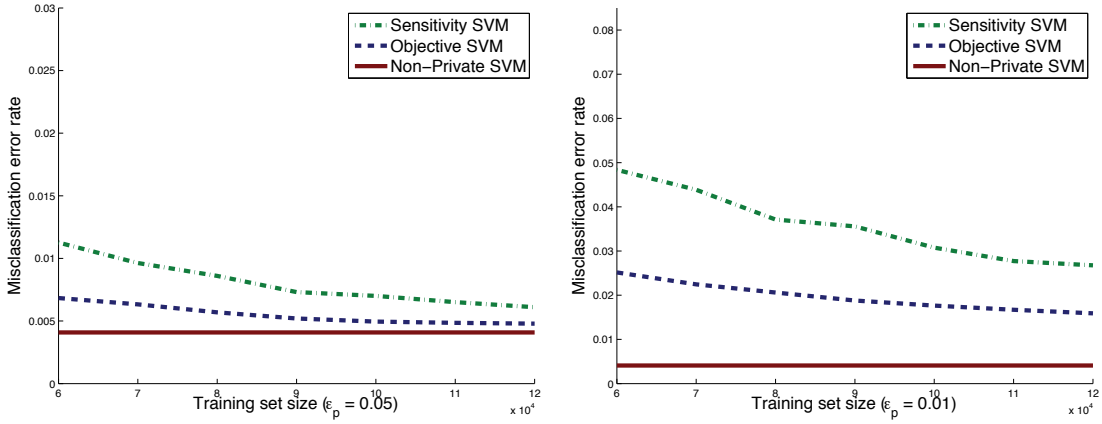
For non-private ERM, we present result for training sets from $n = 300,000$ to $n = 600,000$. The non-private algorithms are tuned by comparing 5 values of Λ on the same training set, and the test set is of size $25,000$. Each reported value is an average over 5 random permutations of the data.

We see from the figures that for non-private logistic regression and support vector machines, the error remains constant with increasing data size. For the private methods, the error usually decreases as the data size increases. In all cases, objective perturbation outperforms the sensitivity method, and support vector machines generally outperform logistic regression.



(a) $\varepsilon_p = 0.05$ (b) $\varepsilon_p = 0.01$

Figure 4: Learning curves for logistic regression on the KDDCup99 data set



(a) $\varepsilon_p = 0.05$ (b) $\varepsilon_p = 0.01$

Figure 5: Learning curves for SVM on the KDDCup99 data set

4. This was chosen for a fair comparison with non-private as well as the output perturbation method, both of which had access to only 5 values of Λ.

8. Discussions and Conclusions

In this paper we study the problem of learning classifiers with regularized empirical risk minimization in a privacy-preserving manner. We consider privacy in the ε_p-differential privacy model of Dwork et al. (2006b) and provide two algorithms for privacy-preserving ERM. The first one is based on the sensitivity method due to Dwork et al. (2006b), in which the output of the non-private algorithm is perturbed by adding noise. We introduce a second algorithm based on the new paradigm of objective perturbation. We provide bounds on the sample requirement of these algorithms for achieving generalization error ε_g. We show how to apply these algorithms with kernels, and finally, we provide experiments with both algorithms on two real data sets. Our work is, to our knowledge, the first to propose computationally efficient classification algorithms satisfying differential privacy, together with validation on standard data sets.

In general, for classification, the error rate increases as the privacy requirements are made more stringent. Our generalization guarantees formalize this "price of privacy." Our experiments, as well as theoretical results, indicate that objective perturbation usually outperforms the sensitivity methods at managing the tradeoff between privacy and learning performance. Both algorithms perform better with more training data, and when abundant training data is available, the performance of both algorithms can be close to non-private classification.

The conditions on the loss function and regularizer required by output perturbation and objective perturbation are somewhat different. As Theorem 6 shows, output perturbation requires strong convexity in the regularizer and convexity as well as a bounded derivative condition in the loss function. The last condition can be replaced by a Lipschitz condition instead. However, the other two conditions appear to be required, unless we impose some further restrictions on the loss and regularizer. Objective perturbation on the other hand, requires strong convexity of the regularizer, convexity, differentiability, and bounded double derivatives in the loss function. Sometimes, it is possible to construct a differentiable approximation to the loss function, even if the loss function is not itself differentiable, as shown in Section 3.4.2.

Our experimental as well as theoretical results indicate that in general, objective perturbation provides more accurate solutions than output perturbation. Thus, if the loss function satisfies the conditions of Theorem 9, we recommend using objective perturbation. In some situations, such as for SVMs, it is possible that objective perturbation does not directly apply, but applies to an approximation of the target loss function. In our experiments, the loss of statistical efficiency due to such approximation has been small compared to the loss of efficiency due to privacy, and we suspect that this is the case for many practical situations as well.

Finally, our work does not address the question of finding private solutions to regularized ERM when the regularizer is not strongly convex. For example, neither the output perturbation, nor the objective perturbation method work for L_1-regularized ERM. However, in L_1-regularized ERM, one can find a data set in which a change in one training point can significantly change the solution. As a result, it is possible that such problems are inherently difficult to solve privately.

An open question in this work is to extend objective perturbation methods to more general convex optimization problems. Currently, the objective perturbation method applies to strongly convex regularization functions and differentiable losses. Convex optimization problems appear in many contexts within and without machine learning: density estimation, resource allocation for communication systems and networking, social welfare optimization in economics, and elsewhere.

In some cases these algorithms will also operate on sensitive or private data. Extending the ideas and analysis here to those settings would provide a rigorous foundation for privacy analysis.

A second open question is to find a better solution for privacy-preserving classification with kernels. Our current method is based on a reduction to the linear case, using the algorithm of Rahimi and Recht (2008b); however, this method can be statistically inefficient, and require a lot of training data, particularly when coupled with our privacy mechanism. The reason is that the algorithm of Rahimi and Recht (2008b) requires the dimension D of the projected space to be very high for good performance. However, most differentially-private algorithms perform worse as the dimensionality of the data grows. Is there a better linearization method, which is possibly data-dependent, that will provide a more statistically efficient solution to privacy-preserving learning with kernels?

A final question is to provide better upper and lower bounds on the sample requirement of privacy-preserving linear classification. The main open question here is to provide a computationally efficient algorithm for linear classification which has better statistical efficiency.

Privacy-preserving machine learning is the endeavor of designing private analogues of widely used machine learning algorithms. We believe the present study is a starting point for further study of the differential privacy model in this relatively new subfield of machine learning. The work of Dwork et al. (2006b) set up a framework for assessing the privacy risks associated with publishing the results of data analyses. Demanding high privacy requires sacrificing utility, which in the context of classification and prediction is excess loss or regret. In this paper we demonstrate the privacy-utility tradeoff for ERM, which is but one corner of the machine learning world. Applying these privacy concepts to other machine learning problems will lead to new and interesting tradeoffs and towards a set of tools for practical privacy-preserving learning and inference. We hope that our work provides a benchmark of the current price of privacy, and inspires improvements in future work.

Acknowledgments

The authors would like to thank Sanjoy Dasgupta and Daniel Hsu for several pointers, and to acknowledge Adam Smith, Dan Kifer, and Abhradeep Guha Thakurta, who helped point out an error in the previous version of the paper. The work of K. Chaudhuri and A.D. Sarwate was supported in part by the California Institute for Telecommunications and Information Technologies (CALIT2) at UC San Diego. K. Chaudhuri was also supported by National Science Foundation IIS-0713540. Part of this work was done while C. Monteleoni was at UC San Diego, with support from National Science Foundation IIS-0713540. The experimental results were made possibly by support from the UCSD FWGrid Project, NSF Research Infrastructure Grant Number EIA-0303622.

References

R. Agrawal and R. Srikant. Privacy-preserving data mining. *SIGMOD Record*, 29(2):439–450, 2000.

A. Asuncion and D.J. Newman. UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2007. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou R3579X? Anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International World Wide Web Conference*, 2007.

K. Ball. An elementary introduction to modern convex geometry. In S. Levy, editor, *Flavors of Geometry*, volume 31 of *Mathematical Sciences Research Institute Publications*, pages 1–58. Cambridge University Press, 1997.

B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the 26th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 273–282, 2007.

A. Beimel, S. P. Kasiviswanathan, and K. Nissim. Bounds on the sample complexity for private learning and private data release. In *Proceedings of the 7th IACR Theory of Cryptography Conference (TCC)*, pages 437–454, 2010.

P Billingsley. *Probability and measure*. A Wiley-Interscience publication. Wiley, New York [u.a.], 3. ed edition, 1995. ISBN 0471007102.

A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In R. E. Ladner and C. Dwork, editors, *Proceedings of the 40th ACM Symposium on Theory of Computing (STOC)*, pages 609–618. ACM, 2008. ISBN 978-1-60558-047-0.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5):1155–1178, May 2007.

K. Chaudhuri and N. Mishra. When random sampling preserves privacy. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 198–213. Springer, 2006. ISBN 3-540-37432-9.

K. Chaudhuri and C. Monteleoni. Privacy-preserving logistic regression. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2008.

C. Dwork. Differential privacy. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006. ISBN 3-540-35907-9.

C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the 41st ACM Symposium on Theory of Computing (STOC)*, 2009.

C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2006a.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *3rd IACR Theory of Cryptography Conference,*, pages 265–284, 2006b.

A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 211–222, 2003.

S.R. Ganta, S.P. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 265–273, 2008.

A. Gupta, K. Ligett, F. McSherry, A. Roth, and K. Talwar. Differentially private approximation algorithms. In *Proceedings of the 2010 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010.

T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391–1415, 2004.

S. Hettich and S.D. Bay. The UCI KDD Archive. University of California, Irvine, Department of Information and Computer Science, 1999. URL `http://kdd.ics.uci.edu`.

N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genetics*, 4 (8):e1000167, 08 2008.

R. Jones, R. Kumar, B. Pang, and A. Tomkins. "i know what you did last summer": query logs and user privacy. In *CIKM '07: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, pages 909–914, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-803-9.

S.P. Kasivishwanathan, M. Rudelson, A. Smith, and J. Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, 2010.

S. A. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *Proc. of FOCS*, 2008.

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

S. Laur, H. Lipmaa, and T. Mielikäinen. Cryptographically private support vector machines. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 618–624, 2006.

A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*, 2006.

A. Machanavajjhala, D. Kifer, J. M. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *Proceedings of the 24th International Conference on Data Engineering (ICDE)*, pages 277–286, 2008.

O. L. Mangasarian, E. W. Wild, and G. Fung. Privacy-preserving classification of vertically partitioned data via random kernels. *ACM Transactions on Knowledge Discovery from Data*, 2(3), 2008.

F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 94–103, 2007.

A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets (how to break anonymity of the netflix prize dataset). In *Proceedings of 29th IEEE Symposium on Security and Privacy*, pages 111–125, 2008.

K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th ACM Symposium on the Theory of Computing (STOC)*, pages 75–84. ACM, 2007. ISBN 978-1-59593-631-8.

N. Okazaki. liblbfgs: a library of limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS). 2009. URL `http://www.chokkan.org/software/liblbfgs/index.html`.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.

A. Rahimi and B. Recht. Uniform approximation of functions with random bases. In *Proceedings of the 46th Allerton Conference on Communication, Control, and Computing*, 2008a.

A. Rahimi and B. Recht. Weighted sums of random kitchen sinks : Replacing minimization with randomization in learning. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2008b.

R.T. Rockafellar and R J-B. Wets. *Variational Analysis*. Springer, Berlin, 1998.

B. I. P. Rubinstein, P. L. Bartlett, L. Huang, and N. Taft. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. In *http://arxiv.org/abs/0911.5708*, 2009.

S. Shalev-Shwartz. *Online Learning: Theory, Algorithms, and Applications*. PhD thesis, The Hebrew University of Jerusalem, July 2007.

S. Shalev-Shwartz and N. Srebro. SVM optimization : Inverse dependence on training set size. In *The 25th International Conference on Machine Learning (ICML)*, 2008.

K. Sridharan, N. Srebro, and S. Shalev-Shwartz. Fast rates for regularized objectives. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2008.

L. Sweeney. Weaving technology and policy together to maintain confidentiality. *Journal of Law, Medicine and Ethics*, 25:98–110, 1997.

L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems*, 2002.

V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, 1998.

R. Wang, Y. F. Li, X. Wang, H. Tang, and X.-Y. Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *ACM Conference on Computer and Communications Security*, pages 534–544, 2009.

L. Wasserman and S. Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010.

A. C.-C. Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 160–164, 1982.

J. Z. Zhan and S. Matwin. Privacy-preserving support vector machine classification. *International Journal of Intelligent Information and Database Systems*, 1(3/4):356–385, 2007.

S. Zhou, K. Ligett, and L. Wasserman. Differential privacy with compression. In *Proceedings of the 2009 International Symposium on Information Theory*, Seoul, South Korea, 2009.

Journal of Machine Learning Research 12 (2011) 1111-1148 Submitted 11/08; Revised 8/10; Published 3/11

Anechoic Blind Source Separation Using Wigner Marginals

Lars Omlor LARS.OMLOR@MEDIZIN.UNI-TUEBINGEN.DE
Martin A. Giese* MARTIN.GIESE@UNI-TUEBINGEN.DE
Section for Computational Sensomotorics, Department of Cognitive Neurology
Hertie Institute for Clinical Brain Research & Center for Integrative Neuroscience
University Clinic Tübingen
Frondsbergstrasse 23
72070 Tübingen, Germany

Editor: Daniel Lee

Abstract

Blind source separation problems emerge in many applications, where signals can be modeled as superpositions of multiple sources. Many popular applications of blind source separation are based on linear instantaneous mixture models. If specific invariance properties are known about the sources, for example, translation or rotation invariance, the simple linear model can be extended by inclusion of the corresponding transformations. When the sources are invariant against translations (spatial displacements or time shifts) the resulting model is called an anechoic mixing model. We present a new algorithmic framework for the solution of anechoic problems in arbitrary dimensions. This framework is derived from stochastic time-frequency analysis in general, and the marginal properties of the Wigner-Ville spectrum in particular. The method reduces the general anechoic problem to a set of anechoic problems with non-negativity constraints and a phase retrieval problem. The first type of subproblem can be solved by existing algorithms, for example by an appropriate modification of non-negative matrix factorization (NMF). The second subproblem is solved by established phase retrieval methods. We discuss and compare implementations of this new algorithmic framework for several example problems with synthetic and real-world data, including music streams, natural 2D images, human motion trajectories and two-dimensional shapes.

Keywords: blind source separation, anechoic mixtures, time-frequency transformations, linear canonical transform, Wigner-Ville spectrum

1. Introduction

Blind source separation is an important approach for the modeling of data by unsupervised learning (Choi et al., 2005; Cichocki and Amari, 2002; Ogrady et al., 2005; Comon and Jutten, 2010). The most elementary class of such methods is based on linear mixture models that combine source signals or mixture components as weighted linear sum. Such linear blind source separation methods have a wide spectrum of applications. Examples include speech processing (Anthony and Sejnowski, 1995; Torkkola, 1996a; Smaragdis et al., 2007; Ogrady et al., 2005), spectral analysis (Nuzillard and Bijaoui, 2000; Chen and Wang, 2001), and the interpretation of biomedical and geophysical data (Hu and Collins, 2004; Aires et al., 2000). Popular approaches exploit typically two classes of generative models: instantaneous and convolutive mixtures (Choi et al., 2005; Ogrady

*. Martin A. Giese is the corresponding author.

©2011 Lars Omlor and Martin A. Giese.

et al., 2005; Pedersen et al., 2007). The *instantaneous mixture model* is defined by the equations:

$$x_i(t) = \sum_{j=1}^{n} \alpha_{ij} s_j(t) \quad i = 1, \cdots, m.$$

The time-dependent signals $x_i(t)$ are approximated by the linear superposition of a number of hidden source signals $s_j(t)$. This superposition is computed separately, time-point by time-point. Contrasting with this model, *convolutive mixtures* assume that the source signals are filtered assuming the filter kernels $\alpha_{ij}(t)$ prior to the superposition, resulting in the mixing model:

$$x_i(t) = \sum_{j=1}^{n} \left(\int_{-\infty}^{\infty} \alpha_{ij}(\tau) s_j(t-\tau) d\tau \right) \quad i = 1, \cdots, m. \tag{1}$$

It is obvious that the instantaneous mixture model is a special case of the convolutive model, where the filter kernels are constrained to be proportional to delta functions $\alpha_{ij}(t) = \alpha_{ij}\delta(t)$. In between these two model classes are *anechoic mixture models* which linearly combine time-shifted and scaled versions of the sources, without permitting multiple occurrences of the same source in the mixture. These models are equivalent to convolutive models for which the filter kernels are constrained to the form $\alpha_{ij}(t) = \alpha_{ij}\delta(t-\tau_{ij})$, resulting in the equation:

$$x_i(t) = \sum_{j=1}^{n} \alpha_{ij} s_j(t-\tau_{ij}) \quad i = 1, \cdots, m. \tag{2}$$

Compared to full covolutive models, anechoic models constrain substantially the space of admissible filter functions. This reduces dramatically the amount of data that is necessary for the estimation of the model parameters. In addition, this restriction of the parameter space often results in model parameters that are easier to interpret, which is critical for many applications that use mixture models for data analysis.

Apart from the problem of robust parameter estimation with limited amounts of data, all blind source separation methods suffer from intrinsic ambiguities. For all discussed models the ordering of the recovered sources is arbitrary. For fully convolutive models the sources can be recovered only up to an unknown filter function (*filter ambiguity*). The distortion of the source shape by this arbitrary filter hampers interpretability of the source signals. This makes it necessary to further constrain the estimation of the sources by introduction of additional auxiliary assumptions, such as minimal distortion (see Matsuoka, 2002 for details). In contrast, for anechoic mixtures (see Equation (2)) the filter ambiguity is limited to an unknown scaling and arbitrary additive shift that can be applied to all time delays τ_{ij}, while the form of the source functions is uniquely defined, except for absolute position. This implies that anechoic mixtures can be advantageous for the modeling of data that are consistent with the corresponding generative model. This advantage is particularly strong if only small amounts of data are available or if the models are employed to interpret the statistical structure of the data.

The presence of shifts (translations) is a common problem in many scientific or technical applications (e.g., spectral displacements due to doppler-shifts in astronomy, asynchronous signal transmission in electrical engineering, or spatial displacements of features in images). The anechoic model provides thus an attractive alternative for the common instantaneous model, which can model shifts only by an introduction of additional sources, typically degrading the stability of the estimates

and the interpretability of the model parameters. Also the assumption of the single occurrence of the sources in the individual components of the mixture is reasonable for many applications. Examples from biology include human motion analysis (Barliya et al., 2009; d'Avella et al., 2008), where the same control signal might influence several muscles or joints with different delays, or in functional magnetic resonance imaging, where time shift occur naturally due to hemodynamic delays (Mørup et al., 2008).

The major part of previous work on anechoic mixtures has considered under-determined (over-complete) source separation problems ($m \leq n$), where the sources outnumber the dimensions of the available data. This is typically the case for acoustic data, for example in the case of the 'cocktail party problem', where the signals of many speakers have to be recovered from a small number of microphones. Such under-determined problems require additional assumptions about the sources (for example sparseness Georgiev et al., 2005; Bofill, 2003; Yilmaz and Rickard, 2004), in order to obtain unique and stable solutions. Since most of existing algorithms for the under-determined case rely on such additional constraints for the estimation of the sources, they cannot be easily generalized for the undercomplete case. This case, where the number of sources is smaller than the dimensionality of he data, is typical for data reduction problems. The application of algorithms developed for the over-determined problem may lead to erroneous results for this case. For example strong sparseness assumptions (like W-disjoint orthogonality Yilmaz and Rickard, 2004) lead to an overestimation of the number of the sources (in the overdetermined case), compared to methods that only assume statistical independence of the sources.
In this paper we present a present a new algorithmic framework for the solution of arbitrary anechoic mixture problems, which is independent of the number of sources and the dimensionality of the data. Contrasting with most previous approaches addressing the model (2), our method is suitable for dimension reduction since it is applicable for the solution of over-determined problems ($m \geq n$). The key idea of the novel framework is to transform the original mixture problem into the time-frequency domain, exploiting the Wigner-Ville transformation (WVT). The resulting transformed problem is completely equivalent to the original problem, but more suitable for an efficient algorithmic solution. Exploiting the fact that for the WVT the knowledge of a limited number of marginals allows the complete and unambiguous reconstruction of the original signal, we devise an algorithm that replaces then original problem by a set lower-dimensional anechoic demixing problems with positivity constraints and a phase retrieval problem. The positive demixing problems are solved by approximative methods, such as nonnegative matrix factorization (NMF) or positive ICA. The projection onto lower-dimensional problems leads to an efficient solution even of higher-dimensional problems with multi-dimensional translations. The obtained solution in time-frequency space is then transformed back into signal space, where in a second step the full solution of the original problem is determined by solution of a phase retrieval problem.

Our method exploits specifically the advantageous mathematical properties of the Wigner-Ville transform. A particular role in this context plays the relationship between the Wigner time-frequency representation and the linear canonical transform (LCT) (also called special affine Fourier transform or ABCD transform). This class of linear integral transformations generalizes classical transformations, like the Fourier or the Gauss-Weierstrass integral transform. It is crucial for the transformation of higher-dimensional problems into a (coupled) set of lower-dimensional problems and provides a theoretical basis for the phase retrieval in the second step of the algorithm. In addition, the choice of appropriate Linear Canonincal Transformations can improve the separability of the source signals.

The paper is structured as follows: After a discussion of related approaches and an introduction of the notation, Section 2 gives a short introduction into the theory of the stochastic Wigner-Ville distribution and its expected value, the Wigner-Ville spectrum. Though most of these results are well established in mathematics they are not so well-known in the machine learning community. The second part of this section introduces the linear canonical transform (LCT) and its properties that are fundamental for our algorithm. For the solution of the non-negative anechoic mixture problem a modification of non-negative matrix factorization (NMF) (Lee and Seung, 1999) is presented in 3.2. Finally, we discuss several phase retrieval methods in 3.3. Section 4 presents in detail three concrete implementations of the novel algorithmic framework. Section 5 presents a validation of the developed method on several data sets, including music streams, natural 2D images, human motion trajectories, and two-dimensional shapes. Finally, conclusions are presented in Section 6.

A preliminary version of the algorithm and some applications have been previously published as conference papers (Omlor and Giese, 2007a,b). However, the complete theoretical framework with a comparison between different implementations has never been published before.

1.1 Related Approaches

Anechoic mixtures have frequently been used in acoustics to model reverberation-free environments. Such models have been treated in several papers focusing on the under-determined case, often in the context of the 'cocktail party problem'. The work in Torkkola (1996a,b) extended the information maximization approach by Anthony and Sejnowski (1995), using the adaptive delay architecture described in Platt and Faggin (1992) in order to unmix anechoic 2×2 mixtures. Another approach by Emile and Comon (1998) is to estimate the unknown parameters directly in the time domain, with the additional assumption of predefined constant mixing weights ($\alpha_{ij} = 1$). Frequency or time-frequency methods, like the DUET algorithm (Yilmaz and Rickard, 2004) or the scatter plot method by Bofill (2003), exploit sparsity properties of the sources in these domains. For the even-determined case ($n = m$) the weights and delays can be estimated by joint diagonalization of specific spectral matrices, as demonstrated in Yeredor (2003). Also a two-dimensional version of the AC-DC joint diagonalization algorithm has been successfully applied for the separation of images that appeared with unknown spatial-shifts (Be'ery and Yeredor, 2008). Other work on the under-determined case is summarized in Ogrady et al. (2005), Arberet et al. (2007) and Namgook and Jay Kuo (2009).
The over-determined case, which is most important for dimension reduction applications has been treated only very rarely so far. In Harshman et al. (2003) this problem has been addressed using an alternating least squares (ALS) algorithm (Shifted Factor Analysis). This algorithm has been revised and improved in Mørup et al. (2007), exploiting the Fourier shift theorem and information maximization in the complex domain (SICA, Shifted Independent Component Analysis).

The performance of independent component analysis and blind source separation methods is critically dependent on the non-Gaussianity of the source distributions (Cardoso, 1998), and the possibility of a sparse representation of the data, which in turn is related to the super-gaussianity of the distribution of the sources. This implies that preprocessing of the signals, for example, by application of bilinear time-frequency transformations before source separation can be essential. In this context time-frequency representations have been used quite frequently in the context of blind source separation (Karako-Eilon et al., 2003; Leung and Siu, 2007; Li et al., 2004; Seki et al., 1998), and specifically for anechoic demixing (Yilmaz and Rickard, 2004). Besides, the application

of the superposition law of the Wigner-Ville spectrum (WVS) in Belouchrani and Amin (1998), such distributions have been mainly applied for pre-processing purposes. Contrasting with this work, our approach relies on further properties of the stochastic Wigner-Ville spectrum (WVS), which to our knowledge, never have been exploited for source separation previously.

1.2 Notations

Throughout the paper the following well-established mathematical notations will be used:

- The notation $:=$ indicates that the left hand side is defined by the right hand side of the equation.
- $\mathbb{i}$ denotes the complex unit.
- E denotes the expectation operator.
- For a scalar, or a function x , x^* denotes the complex conjugate.
- The operators $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the Fourier and inverse Fourier transform respectively, defined by:

$$\mathcal{F}x(f) := \int x(t)e^{-2\pi\mathbb{i}tf}dt = X(f),$$
$$\mathcal{F}^{-1}X(t) = \int X(f)e^{2\pi\mathbb{i}tf}df.$$

  In the case of discrete time variables the same symbols signify the Discrete Fourier Transform (DFT).
- The notation $T_{ij}x$ indicates the shift operator $T_{ij}x(t) = x(t-\tau_{ij})$ with $\tau_{ij} \in \mathbb{R}$.
- If not noted otherwise, $t \in \mathbb{R}^d$ and $x : \mathbb{R}^d \to \mathbb{R}, t \mapsto x(t)$, that is, in general x denotes a multivariate function. In order to distinguish the variable of integration from the functional variable, in addition to the variable t, the variable t' is used. (Thus $x(t')$ denotes the function x at point t'.)
- The symbol $*$ marks the convolution.
- If A, B are two matrices then $\frac{A}{B}$ denotes the entrywise fraction $(\frac{A}{B}) = (\frac{a_{ij}}{bij})_{ij}$.
- $x'(t)$ is short for the derivative $\frac{dx}{dt}(t)$.
- $x \leftarrow y$ implies that x is replaced by y in the current iteration of an algorithm.

2. The Wigner-Ville Distribution and Its Stochastic Generalizations

The Wigner-Ville distribution was originally defined by Wigner (1932) in the context of quantum mechanics. It was later reintroduced in signal analysis by Ville (1948), with the basic idea of defining a joint distribution of the signal energy simultaneously in time and frequency (in physics corresponding to coordinates and momentum). For a continuous scalar real or complex signal (wave function) $x(t)$ the Wigner-Ville distribution is defined as the bilinear transformation:

$$W_x(t,f) := \int x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right) e^{-2\pi i \tau f} d\tau. \tag{3}$$

Unfortunately, this expression cannot be interpreted as a true probability density, since it can become negative. A variety of mathematical properties have been proven for the Wigner-Ville distribution (Mecklenbruker and Hlawatsch, 1997), making it a widely used tool in signal analysis, with generalizations to linear signal spaces, linear time-varying systems or frames (see, e.g., Matz and Hlawatsch, 2003 for review). While the Wigner distribution was developed in the probabilistic framework of quantum mechanics, definition Equation (3) applies to deterministic functions x. In the works of Janssen (1979), Martin (1982) as well as Martin and Flandrin (1985) the deterministic definition Equation (3) has been extended to the very general class of harmonizable stochastic processes. The only requirement for a zero mean random signal to be harmonizable is the existence of a Fourier representation Φ of its autocovariance function $r_x(t,t')$ that is defined by:

$$r_x(t,t') := \mathrm{E}\{x(t)x^*(t')\} = \int\int e^{2\pi i(\lambda t-\mu t')}\Phi(\lambda,\mu)d\lambda d\mu.$$

The probabilistic analogue to the deterministic Wigner distribution for a stochastic process x is given by the stochastic integral:

$$w_x(t,f) := \int x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right) e^{-2\pi i \tau f} dP(\tau).$$

In this formula dP signifies a probability measure, defining a stochastic integral. (See Papoulis et al., 2001 for further details.) In this case, the distribution $w_x(t,f)$ is again a stochastic process. This stochastic integral exists in quadratic mean if the absolute forth order moments $E\{|x|^4\}$ exist (Martin, 1982). Furthermore, the existence of these moments guarantees that the expected value and the integration can be exchanged:

$$\begin{aligned} E\{w_x(t,f)\} &= E\left\{\int x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right) e^{-2\pi i \tau f} dP(\tau)\right\} \\ &= \int E\left\{x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right)\right\} e^{-2\pi i \tau f} d\tau = \int r_x\left(t+\frac{\tau}{2},t-\frac{\tau}{2}\right) e^{-2\pi i \tau f} d\tau \\ &=: \overline{W_x(t,f)}. \end{aligned} \tag{4}$$

The last expression $\overline{W_x(t,f)}$, which can be defined for a more general class of random processes, is called the Wigner-Ville spectrum (WVS). The invertibility of the Fourier integral Equation (4) assures that the WVS is equivalent to the covariance function $r_x(t,t')$. Therefore, it contains full information about the second-order statistics of x. The WVS represents a time-dependent spectrum that is commonly used to study the local and global non-stationarity of random processes.

2.1 Basic Examples for the WVS

White noise process

The white noise process x is defined by:

$$E\{x(t)\} = 0$$
$$r_x(t,t') = \delta(t-t').$$

Then it is easy to see that the WVS is given by:

$$\overline{W_x(t,f)} = \int \delta(\tau) e^{-2\pi i \tau f} d\tau = 1.$$

Signal plus noise

Defining a process $x(t) = y(t) + n(t)$ by the superposition of a deterministic signal $y(t)$ and a zero-mean noise component $n(t)$, the WVS is given by:

$$\begin{aligned}\overline{W_x(t,f)} &= \int E\left\{x\left(t+\frac{\tau}{2}\right)x^*\left(t-\frac{\tau}{2}\right)\right\} e^{-2\pi i \tau f} d\tau \\ &= \int \left[E\{yy^*\} + yE\{n^*\} + y^*E\{n\} + E\{nn^*\}\right] e^{-2\pi i \tau f} d\tau = \int [yy^* + E\{nn^*\}] e^{-2\pi i \tau f} d\tau \\ &= \underbrace{\int yy^* e^{-2\pi i \tau f} d\tau}_{\text{deterministic}} + \int E\{nn^*\} e^{-2\pi i \tau f} d\tau = W_y(t,f) + \overline{W_n(t,f)}.\end{aligned}$$

This shows that for signal plus noise the WVS is given by the sum of the deterministic Wigner distribution and the WVS of the noise.

2.2 Properties of the WVS and the Wigner Distribution

The stochastic as well as the deterministic definition share many properties. Consequently, in the following only the properties of the WVS will be listed, unless the properties of the deterministic WV transformation are different. (Proofs for the properties can be found, for example, in Cohen 1989).

– `Real:` The WVS is a real function of time and frequency:

$$\begin{aligned}\overline{W_x(t,f)}^* &= \int E\left\{x^*\left(t+\frac{\tau}{2}\right)x\left(t-\frac{\tau}{2}\right)\right\} e^{2\pi i \tau f} d\tau \\ &= \int E\left\{x^*\left(t-\frac{\tau}{2}\right)x\left(t+\frac{\tau}{2}\right)\right\} e^{-2\pi i \tau f} d\tau = \overline{W_x(t,f)}.\end{aligned}$$

– `Time-frequency shift covariant:` The Wigner-Ville spectrum of a time frequency shifted signal $\tilde{x}(t) = x(t-t_0)e^{2\pi i f_0(t-t_0)}$ is the shifted WVS of the original signal:

$$\overline{W_{\tilde{x}}(t,f)} = \overline{W_x(t-t_0, f-f_0)}. \tag{5}$$

– `Correct marginals:` The marginals in time and frequency of the WVS reflect the second order properties of the process:

$$\int \overline{W_x(t,f)}df = r_x(t,t) = E\{|x(t)|^2\}, \tag{6}$$

$$\int \overline{W_x(t,f)}dt = r_{\mathcal{F}x}(f,f) = E\{|\mathcal{F}x(f)|^2\}. \tag{7}$$

– `Cross terms:` Due to its quadratic nature, the deterministic Wigner-distribution of a multi-component signal $x = s_1 + s_2$ always contains cross terms of the form:

$$W_{s_1,s_2}(t,f) := \int s_1\left(t+\frac{\tau}{2}\right) s_2^*\left(t-\frac{\tau}{2}\right) e^{-2\pi i\tau f} d\tau.$$

Geometrically, these terms will always occur in the time-frequency plane midway between two (auto-)components $W_{x_{1,2}}$ of the deterministic Wigner distribution (Mecklenbruker and Hlawatsch, 1997). For the stochastic WVS existence of cross terms is determined by the time-frequency correlations of the processes. Is for example $x(t) = \sum_{j=1}^{n} \alpha_j \cdot s_j(t)$, the sum of n uncorrelated zero mean random processes $s_j(t)$, then it is obvious from $r_{s_k,s_l}(t,t') = E\{s_k(t)s_l(t')\} = 0\ \forall k \neq l$ that $\overline{W_{s_k,s_l}(t,f)} = 0\ \forall k \neq l$ and thus:

$$\overline{W_x(t,f)} = \sum_j^n |\alpha_j|^2 \overline{W_{s_j}(t,f)}. \tag{8}$$

A similar superposition law also holds for the sum $x(t) = \sum_{j=1}^{n} \alpha_j \cdot s_j(t)$ of deterministic signals $s_j(t)$ if the weights α_j are uncorrelated zero mean random factors:

$$\overline{W_x(t,f)} = \sum_j^n E\{|\alpha_j|^2\} \overline{W_{s_j}(t,f)}.$$

– `Instantaneous frequency:` For a univariate random signal x which is square mean differentiable, the instantaneous frequency can be defined as:

$$f_x(t) = \frac{\mathrm{Im}(x'(t)x(t)^*)}{2\pi\,\mathrm{var}(x(t))}.$$

The expectation is given by the following relation:

$$E\{f_x(t)\} = \frac{\int f\overline{W_x(t,f)}df}{\mathrm{var}(x(t))} = \frac{E\{\mathrm{Im}(x'(t)x(t)^*)\}}{2\pi\,\mathrm{var}(x(t))}. \tag{9}$$

A similar property also holds for the group delay (Cohen, 1989).

– `Symplectic covariance:` For our algorithm it will be central that signals are related that correspond to Wigner Ville Spectra resulting from each other by linear transformations in the time-frequency plane. More specifically, the WVS is covariant against area-preserving linear

transformations M of the time-frequency plain. Such transformations $M \in \mathbb{R}^{2d\times 2d}$ belong to the symplectic group, that is, M has the form:

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \text{ with } \det(M) = 1 \ , \ A^T C = C^T A \ , \ D^T B = B^T D.$$

The transformed WVS is again the WVS of another signal $\tilde{x}(t)$, which is given by the relationship:

$$\begin{aligned} \overline{W_x(t,f)} &= \overline{W_{\tilde{x}}((t,f)M^T)} \text{ or equivalently} \\ \overline{W_x((t,f)(M^T)^{-1})} &= \overline{W_{\tilde{x}}(t,f)}. \end{aligned} \tag{10}$$

The processes x and $\tilde{x}$ are related by the so called linear canonical transform (LCT), discussed in the following section.

2.3 WV Spectrum Estimation for Non-stationary Signals

In most practical applications, it is necessary to estimate the WVS from a single realization of a process. For this reason ensemble averages are often unavailable. As the WVS is the Fourier transform of the autocovariance function, any estimator for $r_x(t,t')$ is sufficient in the sense of statistics. All such estimators have a fundamental bias-variance tradeoff. Smoothing reduces the variance of the estimate but introduces a bias. One way to derive an estimator for the WVS is to assume the process $x(t)$ is semi-stationary, implying that its characteristics are changing slowly with time. This so called quasi-stationary assumption allows the approximation of the autocovariance function $r_x(t,t')$ with a local average:

$$r_x\left(t+\frac{\tau}{2},t-\frac{\tau}{2}\right) \approx \int F(s-t,\tau)x\left(s+\frac{\tau}{2}\right)x^*\left(s-\frac{\tau}{2}\right)ds.$$

Here F is an arbitrary window function that determines the local time-averaging used to estimate the autocovariance function $r_x(t,t')$. If this approximation is used to form an estimate of the WVS (replacing r_x in Equation (4)) this leads to the equation:

$$\begin{aligned} \overline{W_x(t,f)} &\approx \int W_x(t',f')\vartheta(t-t',f-f')dt'df' \text{ with} \\ F(t,\tau) &= \int \vartheta(t,\nu)e^{-2\pi i\nu\tau}d\nu. \end{aligned} \tag{11}$$

Therefore, all distributions belonging to Cohen's class (Cohen, 1989) form estimators for the WVS (Martin and Flandrin, 1985). An alternative way to justify this particular class of estimators can be found in Akbar and Douglas (1995). Other ways to estimate the WVS include multitaper reassignment (Xiao and Flandrin, 2007) or soft wavelet thresholding (Baraniuk, 1994).

The marginal properties (Equation 6, Equation 7) impose certain conditions on the kernel. This implies that not all members of Cohen's class share all properties of the WVS. From this point of view the choice $\vartheta \equiv \delta(t)$, which approximates the WVS by the deterministic Wigner distribution of the samples, seems a natural choice for the estimator. It should also be noted that, as long as certain properties preserved by the chosen kernels, such as the correct marginals (Equation 6, Equation 7), the derivations derived in the following sections remain valid even if the estimator does not share

all properties of the WVS. The choice of the deterministic Wigner Ville Distribution as estimator is thus less restrictive as it might originally appear.

While the algorithms derived in the following sections do not require at any point the explicit numerical computation of a WVS, numerically estimators have been proposed that allow to implement the computation of the transformation on a computer, for example Martin and Flandrin (1985):

$$\overline{W_x(t,f)} \approx 2 \sum_{\tau=-\infty}^{\infty} \sum_{\tau'=-\infty}^{\infty} F(\tau, 2\tau') x(t+\tau+\tau') x^*(t-\tau+\tau') e^{-2\mathbb{i}\omega\tau}.$$

2.4 The Linear Canonical Transform (LCT)

At several points of the algorithms that will be derived in the following we exploit the symplectic covariance property (Equation 10) of the WVS. This properties links signals that correspond to WVS that are related by a symplectic trasformation in the time-frequency domain. The relationship between x and $\tilde{x}$ as specified by the symplectic covariance property (Equation 10) can be derived explicitly in the case of a nonsingular submatrix B (i.e., $\det(B) \neq 0$). It is given by the following integral transformation, which is called linear canonical transform (LCT) or ABCD-transform (Bultheel and Martínez-Sulbaran, 2006):

$$\mathcal{L}(M)[x](f) := (\det \mathbb{i}B)^{1/2} \int_{-\infty}^{\infty} x(t) \exp\left(\mathbb{i}\pi(t^T B^{-1} A t - 2t^T B^{-1} f + f^T D B^{-1} f)\right) dt. \tag{12}$$

In the case of singular matrices B the general description is more involved and can be found in Alieva and Bastiaans (2007).

Integral transformations of this type play an important role in optics and information processing, as they specify affine transformations in phase space. For example Equation (12) describes the behavior of a wave function x by a propagation through a system of thin lenses, free space or if focused with a satellite dish. For special cases of matrices M many classical integral transformations used in signal processing can be realized. Some examples are, considering a function $x(t)$ with one-dimensional argument t:

– Fourier and fractional Fourier transforms are special cases given by the one-parameter subgroup $M(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$. The fractional Fourier transform with parameter θ is an operator power of the normal Fourier transform with the exponent $\frac{2\theta}{\pi}$. The Integral representation (Ozaktas et al., 2001) is given by:

$$\mathcal{F}^{\theta} x(f) := \sqrt{\frac{1 - \mathbb{i}\cot(\theta)}{2\pi}} e^{\mathbb{i}\cot(\theta) f^2/2} \int_{-\infty}^{\infty} e^{-\mathbb{i}\csc(\theta) f t + \mathbb{i}\cot(\theta)\frac{t^2}{2}} x(t) dt. \tag{13}$$

The case $\theta = \frac{\pi}{2}$ corresponds to the classical Fourier transform. This form was used in the context of one implementation of the proposed algortihmic framework (see Section 4.2).

– The Fresnel transform is defined by the expression:

$$[\mathrm{Fresnel}(z,l)x](\xi) = \frac{e^{\mathbb{i}\pi z/l}}{\sqrt{\mathbb{i}lz}} \int_{-\infty}^{\infty} e^{\mathbb{i}\pi/zl(\xi - t)^2} x(t) dt.$$

is obtained in the case $M = \begin{pmatrix} 1 & b \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & \frac{zl}{2\pi} \\ 0 & 1 \end{pmatrix}$.

– Chirp multiplication is given by $M = \begin{pmatrix} 1 & 0 \\ c & 1 \end{pmatrix}$.

– Scaling can be achieved by $M = \begin{pmatrix} a & 0 \\ 0 & \frac{1}{a} \end{pmatrix}$.

These relationships make it possible to adopt the developed framework to different common representations of data in the frequency space, in order to optimize the performance of the algorithm.

2.5 Main Properties of the Linear Canonical Transform

Independent of the matrix M, all linear canonical transforms share the following properties:

– `Linearity:` Obviously the linear canonical transform is a linear transformation, for example:

$$\mathcal{L}(M)[\lambda x + \mu y](f) = \lambda \mathcal{L}(M)[x](f) + \mu \mathcal{L}(M)[y](f) \; \forall \, \lambda, \mu \in \mathbb{R}.$$

– `Unitarity:` The LCT is a unitary operation. Assuming $(.)^*$ denotes the adjoint operator then:

$$(\mathcal{L}(M))^{-1} = (\mathcal{L}(M))^* = \left(\mathcal{L}(M^{-1})\right).$$

– `Group structure:` The product of two LCT operators with matrices M_1 and M_2 is again a LCT with matrix $M_3 = M_1 M_2$:

$$(\mathcal{L}(M_1)\mathcal{L}(M_2)) = \mathcal{L}(M_1 M_2).$$

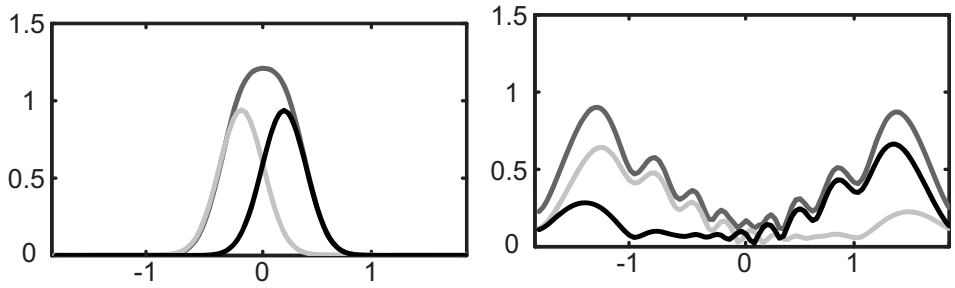
– `Shift Theorem:` Of particular interest for the anechoic mixing problem is the behavior of the linear canonical transform under shifts of the input signal:

$$\mathcal{L}(M)[x(t-\tau)](f) = \exp\left(\mathbb{i}\pi(2f - A\tau)^T C\tau\right) \mathcal{L}(M)[x](f - A\tau). \tag{14}$$

Since the output signal is translated by the term $A\tau$ a good choice of the matrix A can optimize the separability of the the signal x from it is shifted counterparts in particular LCT domains (Sharmaa and Joshi, 2006). To illustrate this point, consider the example of two gaussian signals with a small difference in means (see Figure 1(a)). Linear canonical transformation with $A = 3$ and $B = 0$ increases the mean difference, and thus leads to an improved separability, as shown in Figure 1(b).

3. Algorithmic Framework: Anechoic Demixing Using Wigner Marginals

In the following, we will first derive the basic algorithmic framework by application of the mathematical properties of the WVS, as discussed in the last section, to the anechoic mixing model. The resulting algorithm consists of two major steps: the solution of positive anechoic demixing problems and phase retrieval. In the following, we will discuss specific methods that implement these two main steps of our algorithmic framework. Section 3.2 discusses a special form of Non-negative Matrix Factorization, which is particularly suited for the algorithm. Finally, Section 3.3 discusses different methods for the implementation of the phase retrieval step.



(a) Two gaussian signals with a small difference in mean and their sum (dark grey).

(b) Absolute value of the LCT transformation of the two gaussians with $A = 3, B = 0$.

Figure 1: Example for increased separability of signals in appropriate LCT-domain.

3.1 Application of the WVS to the Mixture Model

We assume that the observed random signals (processes) $x_i(t)$, $i = 1, \ldots, m$ are delayed superpositions of uncorrelated source signals $s_j(t)$, that is:

$$x_i(t) = \sum_{j=1}^{n} \alpha_{ij} \cdot s_j(t - \tau_{ij}) \quad i = 1, \cdots, m \tag{15}$$

$$\text{with } r_{s_i,s_j}(t,t') = E\{s_i(t)s_j^*(t')\} = 0 \text{ for } i \neq j.$$

The shift covariance (5) of the WVS and the superposition law (8) imply the following relation between the spectra of the sources and observations:

$$\overline{W_{x_i}(t,f)} = \sum_{j=1}^{n} |\alpha_{ij}|^2 \overline{W_{s_j}(t - \tau_{ij}, f)} = \sum_{j=1}^{n} |\alpha_{ij}|^2 \overline{W_{T_{ij}s_j}(t,f)}. \tag{16}$$

Since the relationship (16) holds pointwise for all points of the time-frequency plain (t,f) it is also fulfilled after application of an area preserving symplectic transformation M^{-1}:

$$\begin{aligned} \overline{W_{x_i}\big((t,f)(M^T)^{-1}\big)} &= \sum_{j}^{n} |\alpha_{ij}|^2 \overline{W_{T_{ij}s_j}\big((t,f)(M^T)^{-1}\big)} \\ \overset{(10)}{\Longrightarrow} \overline{W_{\mathcal{L}[M](x_i)}(t,f)} &= \sum_{j}^{n} |\alpha_{ij}|^2 \overline{W_{\mathcal{L}[M](T_{ij}s_j)}(t,f)}. \end{aligned} \tag{17}$$

In order to make use of relation (17) an estimator for the WVS is required (cf. Section 2.3). The simplest bias-free estimator for the WVS is just the empirical Wigner transformation. Alternative estimators belong to Cohen's class of time-frequency distributions. In the following only the deterministic Wigner transformation is applied, since it has the advantage of preserving the properties of the WVS. Depending on the smoothing kernel ϑ in Equation (11), these properties are shared by large number of time-frequency-distributions in Cohen's class, consequently leading to the same expression (20) (see below). Replacing the WVS by its deterministic counterpart, Equation (17) transforms into:

$$W_{\mathcal{L}[M](x_i)}(t,f) = \sum_{j=1}^{n} |\alpha_{ij}|^2 W_{\mathcal{L}[M](T_{ij}s_j)}(t,f). \tag{18}$$

This equation could directly serve as basis for the separation of the source signals s_j. However, for higher-dimensional cases the computational costs make a direct solution prohibitive. Since the joint description of the involved function in time and frequency is highly redundant, an equivalent description can be derived by computing the marginals of Equation (18):

$$\begin{aligned}|\mathcal{L}(M)[x_j]|^2(t) &= \int W_{\mathcal{L}(M)[x_j]}(t,f)df = \sum_j^n |\alpha_{ij}|^2 \int W_{\mathcal{L}(M)[T_{ij}s_j]}(t,f)df \\ &= \sum_j^n |\alpha_{ij}|^2 |\mathcal{L}(M)[T_{ij}s_j]|^2(t) \overset{(14)}{=} \sum_j^n |\alpha_{ij}|^2 |\mathcal{L}(M)[s_j](t-A\tau_{ij})|^2. \end{aligned} \tag{19}$$

Note that by introduction of the general matrix M into Equation (17) time and frequency marginals lead to equivalent expressions Equation (19), but with different symplectic transformations $\tilde{M}$. The two types of marginal are are related to each other by the product of M with a $\frac{\pi}{2}$-rotation in the time-frequency plain, resulting in an interchange of the time and the frequency axis. The individual marginals have the form of an anechoic mixture problem with an additional nonnegativity constraint:

$$|\mathcal{L}(M)[x_i]|^2(t) = \sum_{j=1}^n |\alpha_{ij}|^2 |\mathcal{L}(M)[s_j](t-A\tau_{ij})|^2 + n(t). \tag{20}$$

The additional noise term $n(t)$ emphasizes the fact that Equation (20) is an approximation, which only holds precisely under the asymptotic condition $r_{s_i,s_j}(t,t') = 0$. In practice deviation from exact time-frequency disjointness (e.g., due to the finite sample size) result non-zero noise terms $n(t)$. However, the (approximate) solution of Equation (20) gives an estimate for the power spectra of the sources in LCT domain $|\mathcal{L}(M)[s_j]|^2$, the scaled delays $A\tau_{ij}$ and the absolute value of the weights $|\alpha_{ij}|$. Methods for the solution of the positive shifted mixture problem will be discussed in Section 3.2. The missing phase information for $\mathcal{L}(M)[s_j]$ can be recovered by computing multiple marginals depending on different matrices M. Phase retrieval methods for this purpose are discussed in the following Section 3.3. This operation exploits for a second time the special mathematical properties of the WVS with respect to the symplectic covariance.

Since the projections onto LCT domains (19) are also anechoic mixtures, at first glance, the positive system of Equations (20) seems not to provide any advantage compared to the direct solution of Equations (15) or (18). However, a more thorough analysis shows that the transformation into the coupled set of equations for the marginals has the following advantages:

(i) A single marginal described by Equation (19) is not sufficient to reconstruct the signal or its WVS. It is thus necessary to compute a set of different marginals, and the choice of the matrix M in the LCT allows the selection of different marginals that correspond, for example, to different slanted lines in the time-frequency space in the case of fractional Fourier transform. This procedure is different from a source separation directly in the time-frequency plane (Belouchrani and Amin, 1998). In this case (separation in the TF plane), application of the LCT would not result in a facilitation of source separation because it corresponds to an area preserving deformation M (Healy et al., 2008). The chosen approach to work on the marginals however, has the immediate advantage of a much higher computational efficiency since it avoids the high computational cost of the WVS and allows the reduction of multi-variate problems into an uni-variate problems.

(ii) For each single marginal (19) the linear canonical transform, that is, the choice of M, determines the distribution of the estimated sources, and especially their sparseness. This can be exploited to improve the interpretability and compactness of recovered source models. Since the LCT includes a lot of different transformations, such as the Fourier or the fractional Fourier transformation, it provides much more flexibility for an optimization of the representation space of the data than methods relying on a single representation (e.g., Healy et al., 2008). A nice example of this advantage is data that can be represented efficiently by a superposition of chirps, given by functions of the form $e^{2\alpha\pi\mathbb{i}t^2}$. These functions are neither sparse in time nor frequency. However, they have compact support in a correctly chosen LCT domain. A particular useful approach is to chose the matrix M in a way that maximizes the sparsity of the power spectral density within the signal space that is defined by the corresponding LCT. (The WVS in different linear spaces is discussed in Hlawatsch and Kozek (1993).) Formally, one might thus determine M for a given data set that spans the linear space $\mathcal{X} = \text{span}(x_1, \ldots, x_n)$ by maximizing the sparseness of the approximation, formally:

$$\max_M \text{sparsity}\left(|\mathcal{L}(M)[\mathcal{X}]|^2\right).$$

(iii) The magnitude of the effective (scaled) delays of the mixture components in Equation (19) τ_{ij} directly depends on the parameter A of the LCT. An appropriate choice of A results thus in improved separation of the signals (Sharmaa and Joshi, 2006). Furthermore the choice $A = 0$ simplifies the mixture by transforming it into an instantaneous mixture. As the multivariate LCT can be the tensorial product of several one-dimensional LCTs, A can be adjusted for each dimension individually, making it possible to transform a multivariate anechoic mixture into several coupled one-dimensional mixtures (Omlor and Giese, 2007a). This is the core idea behind the implementation of our algorithm for the multi-variate case, which is discussed in Section 4.2.

Additional remarks about the proposed framework:

1. Since both the deterministic and the stochastic WVS share the property of correct marginals, Equation (20) holds also for the exact power-spectra $E\{|\mathcal{L}(M)s_i|^2\}$. The positivity in (20) is thus a direct consequence of the assumption of uncorrelated sources and does not result from an additional approximation. Similarly, the fact that the cross-terms vanish in Equation (18) represent a consequence of this assumption. With respect to these points, the derivation of Equation (18) exploits the same properties as joint diagonalization approaches (e.g., Belouchrani and Amin, 1998).

2. The uncorrelatedness assumption for the sources leads to vanishing cross terms in the full time-frequency plane. However, a single projection (20) requires only that the cross terms in that particular LCT domain vanish. In practice, this simplifies the choice of appropriate LCT domains in order to improve the separability of the sources.

3. The linear weights α_{ij} and the delays τ_{ij} can be estimated exploiting multiple marginals simultaneously. This makes the estimation more robust against the influence of noise.

4. For the special case $\mathcal{L}[M] = \mathcal{F}$ (Fourier transformation), Equation (20) can be derived without the theoretical background of time-frequency analysis. However, the obtained single equation

is not sufficient for the reconstruction of the sources, and without the mathematical theory of the WVS it is not clear how the missing phase information can be recovered.

3.2 Nonnegative Matrix Factorization

In the following we will introduce a modified version of convolutive non-negative matrix factorization as one method that is particularly suitable for the numerically efficient solution of the positive mixture problem (19). Opposed to standard non-negative matrix factorization (NMF), convolutive NMF assumes a non-negative convolutive model (see Equation 1).

3.2.1 CONVOLUTIVE NON-NEGATIVE MATRIX FACTORIZATION

The simple standard non-negative matrix factorization model

$$\mathbf{X} \approx \mathbf{AS} \text{ with } \mathbf{A} \geq 0 \ , \ \mathbf{S} \geq 0 \tag{21}$$

assumes that the data $\mathbf{X}$ arises from a linear superposition of positive vectors. For many applications, such as spectral analysis or blind image deblurring, the data can be described more exactly as a (discrete) positive convolution of the form:

$$x_i(t) = \sum_{j=1}^{n} (a_{ij} * s_j)[t] = \sum_{j=1}^{n} \sum_{u} a_{ij}[u] s_j[t-u] \tag{22}$$
$$a_{ij} \geq 0 \ , \ s_j \geq 0 \ \forall i \in 1, \cdots, m.$$

Depending on the definition of the convolution operator $*$ in Equation (22), the weights a_{ij} are either periodic functions or are characterized by a compact support, referred to as circular or linear convolution. It is always possible to interpret a linear convolution with compact support as a circular convolution with additional zero padding (Rabiner and Gold, 1975). We thus present here the convolutive NMF algorithm in the following for the circular case.

If both, the filters a_{ij} and the sources s_j are unknown, Equation (22) implies the estimation problem:

$$\min_{a_{ij}, b_j} \| x_i(t) - \sum_{j=1}^{n} (a_{ij} * s_j)(t) \| \tag{23}$$
$$\text{subject to } s_j \geq 0 \ , \ a_{ij} \geq 0 \ \forall i, j.$$

Since (22) is both finite-dimensional and linear, it can be expressed in matrix form. In the simple case of one-dimensional (vectorial) data the convolution $a_{ij} * s_j$ is equal to the matrix products:

$$(a_{ij} * s_j) = \begin{bmatrix} a_{ij}(0) & a_{ij}(N) & a_{ij}(N-1) & \cdots \\ a_{ij}(1) & a_{ij}(0) & a_{ij}(N) & \dots \\ a_{ij}(2) & a_{ij}(1) & \ddots & \ddots \\ \vdots & \vdots & \vdots & \vdots \\ a_{ij}(N) & a_{ij}(N-1) & \dots & \dots \end{bmatrix} \cdot \begin{bmatrix} s_j(0) \\ s_j(1) \\ \vdots \\ \\ s_j(N) \end{bmatrix} =: \mathbf{A}^{ij}\mathbf{S}^j$$

$$= \begin{bmatrix} s_j(0) & s_j(N) & s_j(N-1) & \cdots \\ s_j(1) & s_j(0) & s_j(N) & \dots \\ s_j(2) & s_j(1) & \ddots & \ddots \\ \vdots & \vdots & \vdots & \vdots \\ s_j(N) & s_j(N-1) & \dots & \dots \end{bmatrix} \cdot \begin{bmatrix} a_{ij}(0) \\ a_{ij}(1) \\ \vdots \\ \\ a_{ij}(N) \end{bmatrix} =: \mathbf{S}_j\mathbf{A}_{ij}. \tag{24}$$

For higher dimensional data the vectors $\mathbf{S}_j, \mathbf{S}^j$ and the Toeplitz matrices $\mathbf{A}_{ij}, \mathbf{A}^{ij}$ are replaced with column vectors and block circular matrices respectively. Adopting this matrix notation, one can reformulate the model (22):

$$\mathbf{X} := \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = (x_i)_i = \left(\sum_{j=1}^{n} a_{ij} * s_j \right)_i \quad = \left(\sum_{j=1}^{n} \mathbf{A}^{ij}\mathbf{S}^j \right)_i = \begin{bmatrix} \mathbf{A}^{11} & \mathbf{A}^{12} & \dots & \mathbf{A}^{1n} \\ \mathbf{A}^{21} & \mathbf{A}^{22} & \dots & \mathbf{A}^{2n} \\ \vdots & \vdots & \ddots & \end{bmatrix} \cdot \begin{bmatrix} \mathbf{S}^1 \\ \vdots \\ \mathbf{S}^n \end{bmatrix}$$

$$=: \mathbf{A} \cdot \mathbf{S}. \tag{25}$$

and the optimization problem (23):

$$\min_{\mathbf{A},\mathbf{S}} \|\mathbf{X} - \mathbf{A}\mathbf{S}\|$$
$$\text{subject to } \mathbf{A}, \mathbf{S} \geq 0 \; \forall i, j.$$

Writing the deconvolution problem in this form immediately shows the connection to nonnegative matrix factorization. Indeed Equation (25) is just a special case of Equation (21) with a specific structure of the matrix $\mathbf{A}$. Thus according to the multiplicative updates derived by Lee and Seung (1999) Equation (23) can be optimized by repeated iteration of the two steps:

$$\mathbf{A}_{ij} \leftarrow \mathbf{A}_{ij} \frac{(\mathbf{X}\mathbf{S}^T)_{ij}}{(\mathbf{A}\mathbf{S}\mathbf{S}^T)_{ij}}, \tag{26}$$

$$\mathbf{S}_j \leftarrow \mathbf{S}_j \frac{(\mathbf{A}^T\mathbf{X})_j}{(\mathbf{A}^T\mathbf{A}\mathbf{S})_j}. \tag{27}$$

In Equations (26,27) it is important to note that none of the operators $\mathbf{A}, \mathbf{A}^T, \mathbf{S}$ and $\mathbf{S}^T$ appears alone, but always in combination with its influence on another operator. For example, in order to compute

$$\mathbf{A}\mathbf{S} = \left(\sum_{j=1}^{n} a_{ij} * s_j \right)_i$$

it is not necessary to represent either $\mathbf{A}$ or $\mathbf{S}$ as high-dimensional matrix. Additionally the convolution can be implemented very efficiently exploiting Fast Fourier Transform (FFT), instead of an expensive matrix multiplication. From relations (24) and (25) it is easy to derive that the effect of the transposed operators $\mathbf{A}^T, \mathbf{S}^T$ can be expressed as:

$$\mathbf{A}^T\mathbf{X} = \left(\sum_{i=1}^{n} a_{ij}^T * x_i\right)_j = \left(\sum_{i=1}^{n} \mathcal{F}^{-1}\left((\mathcal{F}a_{ij})^* \mathcal{F}x_i\right)\right)_j,$$
$$\mathbf{X}\mathbf{S}^T = \left(x_i * s_j^T\right)_{ij} = \left(\mathcal{F}^{-1}\left((\mathcal{F}s_j)^* \mathcal{F}x_i\right)\right)_{ij}.$$

Here we exploited the fact that the impulse response of the adjoint filter is the complex conjugate of the original signal. Replacing the matrices in Equation (26,27) by their corresponding representations in the discrete Fourier domain, results in the following fast *update rules for convolutive NMF*:

$$a_{ij} \leftarrow a_{ij} \frac{\left(\mathcal{F}^{-1}\left((\mathcal{F}s_j)^* \cdot \mathcal{F}x_i\right)\right)}{\left(\mathcal{F}^{-1}\left(\sum_k (\mathcal{F}s_j)^* \cdot \mathcal{F}a_{ik} \cdot \mathcal{F}s_k\right)\right)} \forall i, j,$$
$$s_j \leftarrow s_j \frac{\mathcal{F}^{-1}\left(\sum_k (\mathcal{F}a_{kj})^* \cdot \mathcal{F}x_k\right)}{\mathcal{F}^{-1}\left(\sum_{p,l} (\mathcal{F}a_{pj})^* \cdot \mathcal{F}a_{pl} \cdot \mathcal{F}s_l\right)} \forall j.$$

Though the non-negative deconvolution problem (22) is formally equivalent to the regular NMF the matrix product $\mathbf{AB}$ does not define a low rank decomposition. As a consequence Equation (22) has, in contrast to Equation (21), no prospect of uniqueness. For example, every member of the family $a_{ij} = x_i(t - \tau_j)$ and $s_j = \delta_{-\tau_j}$ is a (trivial) possible solution. In order to find unique solutions for the optimization problem (22) it is thus necessary to introduce some form of regularization, such as sparseness assumptions about the filters (e.g., Chen and Cichocki, 2005).

3.2.2 CONVOLUTIVE NMF FOR ANECHOIC DEMIXING (ANECHOIC NMF (ANMF))

The positive anechoic demixing problem

$$x_i(t) = \sum_{j=1}^{n} \alpha_{ij} s_j(t - \tau_{ij}) \text{ with } x_i, \alpha_{ij}, s_j \geq 0 \; \forall \, i, j$$

can be written as a special case of a convolution, by defining the filters as $\nu_{ij} = \alpha_{ij}\delta(t - \tau_{ij})$:

$$x_i = \sum_{j=1}^{n} \nu_{ij} * s_j \text{ with } x_i, \nu_{ij}, s_j \geq 0 \; \forall \, i, j. \tag{28}$$

The anechoic problem (28) can thus be solved using the convolutive NMF algorithm for the solution of the problem (22), if sparseness of the filters is enforced. Replacing the update rules for the squared euclidian distance with rules which minimize Amari's alpha-divergence (Cichocki et al., 2008) results in the update rules:

$$\nu_{ij} \leftarrow \left[\nu_{ij}\left(\mathcal{F}^{-1}\left((\mathcal{F}s_j)^* \mathcal{F}\left[\frac{y_i}{\mathcal{F}^{-1}\left(\sum_k \mathcal{F}\nu_{ik}\mathcal{F}s_k\right)}\right]^{\beta}\right)\right)^{\frac{\mu}{\beta}}\right]^{1+\lambda_1}, \tag{29}$$

$$s_j \leftarrow \left[s_j\left(\mathcal{F}^{-1}\sum_k (\mathcal{F}\nu_{jk})^* \mathcal{F}\left[\frac{y_k}{\mathcal{F}^{-1}\left(\sum_l \mathcal{F}\nu_{kl}\mathcal{F}s_l\right)}\right]^{\beta}\right)^{\frac{\mu}{\beta}}\right]^{1+\lambda_2} \tag{30}$$

which can be adjusted for sparseness. The specific choice $\mu = 1.9, \beta = 2, \lambda_1 = 0.02, \lambda_2 = 0$ results in sparse features (Cichocki et al., 2008). Alternative approaches for the introduction of sparseness include, for example, bayesian regularization (Lin and Lee, 2005).

One disadvantage of delay estimation by deconvolution, is the difficulty to locate the exact peak of the sparse filter, especially for fractional delays (non-integer delays). This can be avoided if the multiplicative updates for the sparse filter given by Equation (29), are replaced with a constrained standard estimator for the delays (see Appendix).

The positive anechoic mixture problem that is given by Equation (20)

$$|\mathcal{L}(M)[x_i]|^2(t) = \sum_{j=1}^{n} |\alpha_{ij}|^2 |\mathcal{L}(M)[s_j](t - A\tau_{ij})|^2 + n(t).$$

can be solved with this anechoic non-negative matrix factorization approach (ANMF). The derivation of the last equation assumed vanishing cross-terms in the Wigner-Ville spectrum (equivalent to the assumption of uncorrelated sources). Similar as in Vollgraf et al. (2000), this together with the marginal properties of the Wigner-Ville cross spectrum implies vanishing LCT cross-spectra, that is:

$$\begin{aligned} W_{s_i,s_j}(t,f) \overset{i\neq j}{=} 0 \;&\Rightarrow\; \mathcal{L}(M)(s_i)\mathcal{L}(M)(s_j)^* \overset{i\neq j}{=} 0 \\ &\Rightarrow\; |\mathcal{L}(M)(s_i)|^2 |\mathcal{L}(M)(s_j)|^2 \overset{i\neq j}{=} 0. \end{aligned} \tag{31}$$

The fact that the cross-spectra are not present in Equation 31 permits a reformulation as a constraint for the anechoic NMF problem (19) of the form:

$$\mathbf{S}\mathbf{S}^T = I. \tag{32}$$

This orthogonality constraint guarantees a unique solution NMF-factorization problem (Ding et al., 2006). In addition, it is closely linked to positive independent component analysis (Yang and Yi, 2007; Plumbley and Oja, 2004). Equation (32) represents a special property of the estimator, which it is not necessarily fulfilled for the expectations $E\{\mathcal{L}(M)(s_i)\}$ because they are not zero mean. This implies that the condition (32) guaranties source separation, but only if one imposes an additional asymptotic constraint on the sources. This orthogonality constraint (32) guaranties the uniqueness of the solution of the anechoic demixing problem, making the new algorithm a true blind source separation method. In practice, however, it is often useful to relax this restriction for the sources and to replace the orthogonal NMF update rules with more classical updates, for example, based on least squares. The resulting model can capture also sources that are not perfectly uncorrelated (independent), often resulting in better approximation of the data.

3.3 Phase Retrieval

The solution of the positive anechoic mixture problem:

$$|\mathcal{L}(M)[x_i]|^2(t) = \sum_{j=1}^{n} |\alpha_{ij}|^2 |\mathcal{L}(M)[s_j](t - A\tau_{ij})|^2$$

that is given by Equation (20) yields the LCT power spectra $|\mathcal{L}(M)[s_j]|^2$. In order to reconstruct the sources s_j, or equivalently their LCT transforms $\mathcal{L}(M)[s_j]$, the missing phase information needs to be recovered. Depending on the choice of M in (20) several phase recovery techniques can be applied for this purpose.

3.3.1 DECONVOLUTION

In many applications, such as computer tomography or image processing, the linear weights α_{ij} can be assumed to be nonnegative. This implies that the absolute value operation on the weights can be dropped in (20). For $A \neq 0$ in this case the anechoic mixture (2) reduces to a simple deconvolution problem with known filters $\nu_{ij} = a_{ij}\delta(t-\tau_{ij})$. If the weights are real and have arbitrary signs the filter can determined only up to an arbitrary sign, and if the weights are complex a constant complex phase factor of the filter remains undetermined. In all cases (with appropriate reguarization) the filters can be estimated applying applying a standard deconvolution algorithms, such as the Wiener filter (Wiener, 1964), and the sources s_j can be retrieved by least squares estimation.

3.3.2 GERCHBERG-SAXTON (GS) ALGORITHM

In typical phase retrieval applications the parameters known about the signal are the phase and some other properties like the support. A simple procedure to recover the signal from these constraints is based on the projection onto convex sets (POCS). This method iteratively projects random starting values onto constrained sets that are specified by the given data. This approach can been applied for the reconstruction of signals from multiple power-spectra (Zalevsky et al., 1996). For the phase retrieval problem in our algorithm, an estimate of the signal is transformed back and forth between different LCT domains, always replacing the estimated amplitude spectrum by the known amplitude spectrum from the solution of the positive mixture problem and re-eastimating the phase. This fundamental algorithm is called Gerchberg-Saxton (GS) procedure (Gerchberg and Saxton, 1972). Given the spectra for the LCT domains $M_0, \ldots, M_k$, it can be characterized by the following pseudo-code:

Algorithm 1: Simple GS algorithm for the linear canonical transform

Input: Spectra $|\mathcal{L}(M_0)[x]|, \ldots, |\mathcal{L}(M_k)[x]|$
$y = |\mathcal{L}(M_0)[x]|$;
$y_{\text{old}} = -\infty$; *% Initial vector with entries* $-\infty$
while $\|y - y_{old}\| > \varepsilon$ **do**
  for $i = 0 : k$ **do**

$$y_{\text{old}} = y$$

$$y = |\mathcal{L}(M_{\text{mod}(i+1,k+1)})[x]| \frac{\mathcal{L}(M_{\text{mod}(i+1,k+1)} M_i^{-1})[y]}{|\mathcal{L}(M_{\text{mod}(i+1,k+1)} M_i^{-1})[y]|}$$

  end
end

According to the properties of the LCT $\mathcal{L}(M_{i+1}M_i^{-1})$ transforms a signal from the domain M_i into the domain belonging to M_{i+1}. The modulo operation $\text{mod}(i+1, k+1)$ just ensures a cyclic run through all parameters.

3.3.3 HIGHER-ORDER WIGNER MOMENTS

Given a one-dimensional signal $x(t) = |x(t)|\exp(\mathbb{i}\varphi(t))$ the instantenous frequency property of the WVS (9) is given by:

$$\begin{aligned}\int fW_x(t,f)df &= \mathrm{Im}(x'(t)x^*(t)) = \mathrm{Im}\Big(\big(|x(t)|'e^{\mathbb{i}\varphi(t)} + |x(t)|e^{\mathbb{i}\varphi(t)}\mathbb{i}\varphi'(t)\big)|x(t)|e^{-\mathbb{i}\varphi(t)}\Big)\\ &= |x(t)|^2\varphi'(t).\end{aligned}$$

This link between the derivative of the phase of a signal and the first moment, with respect to frequency, of it is Wigner distribution can be applied to Equation (17). It is thus possible to compute the missing phase information from the relation:

$$\begin{aligned}&|\mathcal{L}(M)[x_i(t)]|^2\varphi'_{L^C_T(M)[x_i]}(t) = \int fW_{\mathcal{L}(M)[x_i]}(t,f)df = \sum_{j=1}^{n}|\alpha_{ij}|^2\int fW_{\mathcal{L}[M](T_{ij}s_j)}(t,f)df\\ &= \sum_{j=1}^{n}\underbrace{|\alpha_{ij}|^2|\mathcal{L}(M)[s_j(t-A\tau_{ij})]|^2}_{\text{known from (20)}}\big(\varphi'_{\mathcal{L}(M)[s_j]}(t-A\tau_{ij}) + 2\pi C\tau_{ij}\big). \qquad (33)\end{aligned}$$

Using the estimators from Equation (20) in relation (33), allows the computation of the phase derivatives $\varphi'_{\mathcal{L}(M)[s_j]}$. Thus by integration, the missing phase information can be theoretically recovered up to the natural ambiguity of an arbitrary timeshift of the sources. In practice, this approach has the disadvantage that only the product of the power spectrum and the phase derivative can be computed. Zeros in the power spectrum thus prevent a direct integration of the phase information. However, this approach is feasible if it is combined with the GS algorithm (presented in 3.3.2). For this purpose, not only the correct power-spectra are enforced during each step of the GS algorithm, but in addition also also the constraint given by (33).

3.3.4 PHASE RETRIEVAL FROM TWO SIMILAR LCT SPECTRA

In theory, the Gerchberg-Saxton algorithm is suitable for computing the phase from arbitrary two LCT power spectra. In practice however (Cong et al., 1998), the speed of convergence highly depends on the distance between the power spectra specified by different values of the matrix M. For spectra defined by very similar matrices the GS procedure might fail. In this case, the following procedure permits to recover the phase from two very similar power spectra.

For simplicity consider again the one-dimensional signal case. This makes it possible to reformulate the instantaneous frequency property of the WVS:

$$\begin{aligned}\mathcal{F}|\mathcal{L}(M)[x(t)]|^2(\omega) &= \int|\mathcal{L}(M)[x(t)]|^2e^{-2\pi\mathbb{i}\omega t}dt \overset{(10)}{=} \iint W_x\big((t,f)(M^T)^{-1}\big)dfe^{-2\pi\mathbb{i}\omega t}dt\\ &= \iint W_x(u,v)e^{-2\pi\mathbb{i}\omega(Au+Bv)}dudv =: \mathcal{A}_x(A\omega,B\omega). \qquad (34)\end{aligned}$$

The two-dimensional Fourier transform of the Wigner distribution $\mathcal{A}_x(A\omega,B\omega)$ is called the Ambiguity-function (Mecklenbruker and Hlawatsch, 1997). Due to the properties of the Fourier transform we obtain:

$$t^kf^lW_x(t,f) = \left(\frac{1}{2\pi\mathbb{i}}\right)^{k+l}\mathcal{F}\left[\frac{\partial^{k+l}\mathcal{A}_x(\alpha,\beta)}{\partial^k\alpha\,\partial^l\beta}\right](t,f).$$

Now the instantaneous frequency property can be formulated as follows:

$$\begin{aligned}
|x(r)|^2\varphi_x'(r) &= \int fW_x(r,f)df = \int \frac{1}{2\pi\mathbb{i}}\mathcal{F}\left[\frac{\partial\mathcal{A}_x(\alpha,\beta)}{\partial\beta}\right](r,f)e^{-2\pi\mathbb{i}0f}df \\
&= \frac{1}{2\pi\mathbb{i}}\int\left(\frac{\partial\mathcal{A}_x(\tau,u)}{\partial u}\right)_{u=0}e^{-2\pi\mathbb{i}r\tau}d\tau \\
&\overset{\tau=A\omega}{=} \frac{1}{2\pi A\mathbb{i}}\int\left(\frac{\partial\mathcal{A}_x(A\omega,u)}{\partial u}\right)_{u=0}e^{-2\pi\mathbb{i}rA\omega}d\omega \\
&= \frac{1}{2\pi A\mathbb{i}}\int\left(\frac{\partial\mathcal{A}_x(A\omega,B\omega)}{\partial B}\frac{1}{\omega}\right)_{B=0}e^{-2\pi\mathbb{i}rA\omega}d\omega \\
&\overset{(34)}{=} \frac{1}{2\pi A\mathbb{i}}\iint\left[\frac{\partial|\mathcal{L}(M)[x(t)]|^2}{\partial B}\right]_{B=0}\frac{1}{\omega}e^{-2\pi\mathbb{i}\omega t}dte^{-2\pi\mathbb{i}rA\omega}d\omega \\
&= \int\left[\frac{\partial|\mathcal{L}(M)[x(t)]|^2}{\partial B}\right]_{B=0}\left(\frac{1}{2\pi A\mathbb{i}}\int\frac{1}{\omega}e^{-2\pi\mathbb{i}(\omega t+rA\omega)}d\omega\right)dt \\
&= \frac{1}{2}\int\left[\frac{\partial|\mathcal{L}(M)[x(t)]|^2}{\partial B}\right]_{B=0}\operatorname{sgn}(\frac{t}{A}+r)dt. \qquad (35)
\end{aligned}$$

Equation (35) links the derivative of LCT intensities $|\mathcal{L}(M)[x]|^2$ with the instantaneous frequency of the signal. Given two close intensities the derivative $\left(\partial|\mathcal{L}(M)[x(t)]|^2/\partial B\right)_{B=0}$ can be approximated by a difference quotient, so that the local frequency can be recovered (Bastiaans and Wolf, 2003).

4. Example Algorithms

Summarizing the theoretical considerations from Section 3.1, we conclude that with the assumption of uncorrelated sources general the anechoic mixture problem for mixtures of the form

$$x_i(t) = \sum_{j=1}^{n}\alpha_{ij}s_j(t-\tau_{ij}) \quad i = 1,\cdots,m$$

can be solved by the following two step procedure:

Algorithm 2: Generic algorithm for anechoic demixing based on Wigner-marginals

Input: Observed data x_i , $i = 1,\ldots,m$, LCT domains M_k , $k = 1,\ldots,K$

Step 1: Solve the positive anechoic demixing problems given by:

$$|\mathcal{L}(M_k)[x_j]|^2(t) = \sum_{j}^{n}|\alpha_{ij}|^2|\mathcal{L}(M_k)[s_j](t-A\tau_{ij})|^2$$

for example, using the ANMF algorithm (1) or the bayesian Probabilistic Latent Component Analysis (PLCA) algorithm (Smaragdis et al., 2007).

Step 2: Recover the phase information for s_j by one of the methods discussed in Section 3.3.

The general algorithm 2 is quite flexible, and it can be implemented by many choices for the LCT and the phase retrieval method. To illustrate this generality we discuss in the following three example implementations that cover a wide spectrum of applications.

4.1 Example 1: Time Series Demixing using Higher-order Wigner Moments

The choice $M = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ reduces the LCT to the standard Fourier transform. With $A = 0$ this simplifies the first step in algorithm 2, reducing the positive anechoic mixture to an instantenous mixture problem:

$$|\mathcal{F}x_i|^2(f) = \sum_j^n |\alpha|_{ij}^2 |\mathcal{F}s_j|^2(f).$$

In this case the phase retrieval (step 2) can be implemented by an alternating least squares approach that is given by the following iteration:

---

for *iter=1:maxiter* **do**

Compute the instantaneous phase $\varphi'_{(\mathcal{F}s_j)}$ by solving numerically the following equation:

$$|\mathcal{F}x_i(f)|^2 \varphi'_{(\mathcal{F}x_i)}(f) = \sum_j^n |\alpha|_{ij}^2 |\mathcal{F}s_j(\xi)|^2 \left(\varphi'_{(\mathcal{F}s_j)}(f) - 2\pi\tau_{ij}\right)$$

Update the delays τ_{ij} using an arbitrary delay estimator (e.g., Swindelhurst, 1998.).

end

4.2 Example 2: Multivariate Demixing using Fractional Fourier Transform

In the case of multivariate data (e.g., images) the simplest choice for a linear canonical transform is just the tensorial product of one dimensional LCTs, corresponding to the choice of diagonal matrices for A,B,C,D. This has the advantage that by selection of $\mathrm{diag}(A) = (a_{ii})_i$ the multivariate positive demixing problem simplifies to a set of one-dimensional problems.

As an illustrative example we discuss the demixing of two-dimensional image data $x_i(t_1,t_2)$, where t_1 and t_2 are the pixel coordinates. A simple example for a tensorial linear canonical transform are the fractional Fourier transforms $\mathcal{F}^{(\theta_1,\theta_2)} = \mathcal{F}_1^{\theta_1}\mathcal{F}_2^{\theta_2}$. In this notation $\mathcal{F}_i^{\theta}$ represents the fractional Fourier transform $\mathcal{F}^{\theta}$ (as defined by (13)) in the ith variable (see (13) or Ozaktas et al., 2001). The system of equations (20) then reduces to:

$$|\mathcal{F}^{(\pi/2,\theta)}x_i(\omega_1,\omega_2)|^2 = \sum_{j=1}^n |\alpha|_{ij}^2 |\mathcal{F}^{(\pi/2,\theta)}s_j\big((\omega_1,\omega_2) - (0,\tau_{ij_2}\cos\theta)\big)|^2,$$

$$|\mathcal{F}^{(\theta,\pi/2)}x_i(\omega_1,\omega_2)|^2 = \sum_{j=1}^n |\alpha|_{ij}^2 |\mathcal{F}^{(\theta,\pi/2)}s_j\big((\omega_1,\omega_2) - (\tau_{ij_1}\cos\theta,0)\big)|^2.$$

Note that for multivariate data the shifts τ_{ij} are vectors, denoted here with $\tau_{ij} = (\tau_{ij_1},\tau_{ij_2})$. Since in this context the variables α_{ij} model intensities of basis images s_j, one can assume $\alpha_{ij} \geq 0$. Hence phase retrieval can be accomplished by a simple deconvolution procedure, as described in 3.3.1.

4.3 Example 3: LCT Based Positive Demixing of Time Series

For the univariate case a particularly attractive parameter choice for the linear canonical transform is $A = 1$ and $B = 1$. For $A = 1$ the values of the delays remain unchanged, and $B = 1$ has the

considerable numerical advantage that the integral transform (12) does not involve scaling. Thus this transformation just converts the original anechoic mixture problem into a nonnegative mixture problem that can be solved using the update rules given by (29) and (30).

5. Applications

In its generic form algorithm 2 has a very broad spectrum of applications. The inclusion of shifts into the mixing model makes it suitable for the shift-invariant learning of features or sources. This problem is relevant in a variety of areas. Furthermore, the theoretical results in Section 3.1 are valid independent of the dimensionality of the data and of the number of extracted sources. However, the ANMF algorithm is better suited for the over-determined case. We constrained the testing of our framework to dimension-reduction for 1D and 2D data, since these are the most frequent problems outside of acoustics. A version of the code that implements the basic version of the algorithm can be downloaded from the web page: `http://www.compsens.uni-tuebingen.de/pub/download/software/anmf`.

5.1 Sound Mixtures

Since the formulation of the "cocktail party problem" by Cherry (1953) acoustics has been a major field for the application of blind source separation methods. The cocktail party problem refers to the separation of individual human speakers in a noisy environment from a limited number of signals, recorded by microphones. In this context, many methods for the unmixing of sound signals have been developed over the years (e.g., Anthony and Sejnowski, 1995; Torkkola, 1996a). The most realistic linear models for BSS in acoustics are convolutive. However, anechoic mixtures are relevant for the case of reverberation-free environments (Bofill, 2003; Yilmaz and Rickard, 2004).

To demonstrate that algorithm 2 is applicable to sound mixtures we present three different separation problems for synthetically generated delayed mixtures (model (2)) using speech and sound segments from the ICA benchmark described in Cichocki and Amari (2002). In total the data set consisted of 14 signals with a length of 8000 time points. In order to obtain statistically representative results, data sets were recomputed 20 times with random selection of the source signals, and/or of the mixing and delay matrices. Three types of mixtures were generated:

(I) Mixtures of 2 source segments with random mixing and delay matrices (2×2), each resulting in two simulated signals $x_{1,2}$.

(II) Mixtures of 2 randomly selected segments from the speech data basis using the constant mixing matrix A and the constant delay matrix T:

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 1 \\ 10 & 5 \\ 1 & 2 \\ 1 & 1 \end{pmatrix}, \; T = (\tau_{ij})_{ij} = \begin{pmatrix} 0 & 4000 \\ 2500 & 5000 \\ 100 & 200 \\ 1 & 1 \\ 500 & 333 \end{pmatrix}.$$

Data set (II) with fixed mixing and delay matrices was included since completely random generation sometimes produced degenerated anechoic mixtures (instantaneous mixtures or ill-conditioned mixing matrices).

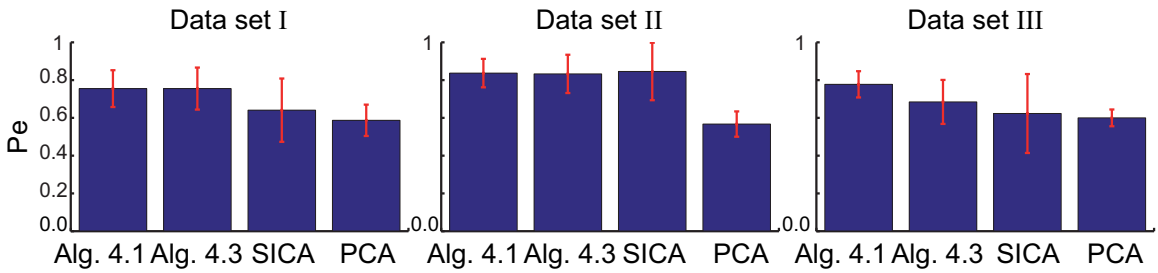


Figure 2: Comparison of different blind source separation algorithms for synthetic mixtures of sound signals with delays (data sets I-III, see text).

(III) A third data set was generated by mixing two randomly selected source segments with random mixture matrices and random delay matrices.

We compared the implementations of our algorithm described in section 4.1 and 4.2 with principal component analysis (as baseline) and shifted independent component analysis (SICA) (Mørup et al., 2007). The performance *Pe* of each algorithm was measured by the maximum of the cross-correlations between extracted sources $s_{\text{extract},j}$ and original sources $s_{\text{orig},j}$ (after appropriate matching of individual sources, since the recovered sources are not ordered in a specific manner):

$$Pe = (1/n) \sum_{j=1}^{n} \max_{\tau} \left| E\{s_{\text{approx.},j}(t) s_{\text{orig},j}(t+\tau)\} \right|.$$

5.1.1 RESULTS FOR THE SOUND DEMIXING

The results for the sound demixing are summarized in Figure 2. The bar plots show the mean and the standard deviation of the performance measure *Pe* determined for the twenty simulations, comparing the methods for the data sets I-III. As expected, the performance *Pe* of the instantaneous mixture model assumed by principal component analysis (PCA) is worse than the performance achieved by the three compared anechoic methods. Overall the classical Fourier domain method 4.1 with $Pe \approx 80\%$ is superior to both the LCT-method 4.3 ($Pe \approx 75\%$) as well as SICA ($Pe \approx 70\%$). This reflects the well-known fact that the Fourier frequency domain is well suited for the separation of sound signals.

5.2 Human Motion Data

Characterizing manifolds that parameterize human motion is an important task for both, the analysis and the synthesis of movement trajectories. In analysis (Flash and Hochner, 2005) a popular idea is that such manifold representations reflects some lower dimensional space that results from the fact that the central nervous system reduces the exploited degree of freedoms by application of appropriate control strategies. A related important interpretation is that the set of source signals, that are appropriate for the reconstruction of human movement trajectories, or electromyography (EMG) signals, might reflect control units, synergies or movement primitives that are exploited by the central nervous system to simplify the underlying control problem by avoiding the curse of

dimensionality (Bellman, 1957). An overview of BSS methods in this field can be found in Tresch et al. (2006) or Chau (2001).

In synthesis the main purpose of an approximation of such a manifold is to represent a set of natural movements, in order to guarantee realistic looking animations. For this purpose many different methods have been suggested, ranging from linear or multilinear methods like PCA (Safonova et al., 2004) to nonlinear methods like isomap (Jenkins and Mataric, 2002), locally linear embedding (LLE) (Roweis and Saul, 2000), or methods based on space-time correspondence (Ilg et al., 2004).

Our second data set consists of human motion data. We show that anechoic mixtures have the potential for a strong dimensionality reduction. A first data set consisted of joint angle trajectories (Euler angles) that were computed from motion capture data (mocap), recorded from twenty-five lay-actors who executed walking with five basic emotional styles (neutral, happy, angry, sad and fear). A second data set consisted of the shoulder and elbow trajectories of five actors executing various arm movements (right-handed throwing, golf swing and tennis swing). All mocap data was recorded using a VICON 612 motion capture system with seven, respectively nine cameras. The system has a sampling frequency of 120 Hz and determines the three-dimensional positions of reflective markers (diameter: 1.25 cm) with spatial error below 1.5 mm. The markers were attached to skin or tight clothing with double-sided adhesive tape, according to the positions of VICON's Plug-In-Gait marker set. Commercial VICON software was used to reconstruct and label the markers, and to interpolate short missing parts of the trajectories.

5.2.1 RESULTS FOR THE MOTION DATA

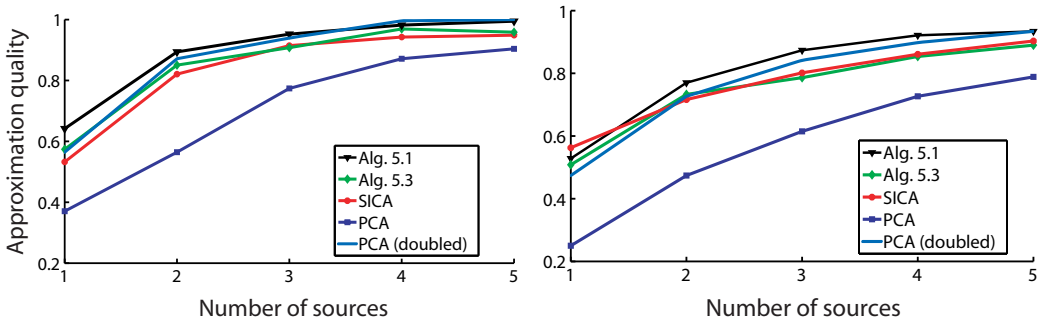
To quantify the performance of different methods for dimension reduction, we measured the quality of approximation as a function of the number of sources. The quality of an approximation F to the data matrix X can be quantified by the quotient:

$$Q := 1 - \frac{\|X - F\|}{\|X\|}$$

where $\|.\|$ is the Frobenius norm. This measure is related to explained variance, which is defined by $1 - \left(\|X-F\|/\|X\|\right)^2$, but it has the advantage of linear scaling with the residual-norm. For small residuals explained variance is hard to interpret since, due to the fact that it is quadratic in the residual norm it results in values close to one even for mediocre approximations. Figure 3 shows this measure for approximation quality as a function of the number of sources comparing principal component analysis (as baseline), shifted independent component analysis (SICA) (Mørup et al., 2007) and the example algorithms described in Sections 4.1 and 4.3.

The results of the comparison are shown in Figure 3. Overall, the best approximation is achieved by the Fourier-domain algorithm 4.1, exceeding the quality of PCA with the double number of sources (light blue line in Figure 3(a)). Slightly worse performance is obtained with algorithm 4.3, likely explained by the additional positivity constraint $\alpha_{ij} \geq 0$. Qualitatively the same results are obtained for the non periodic arm movements (figure 3(b)). The total approximation quality is lower in the second case, reflecting the higher variability of this data set.

In general, for both classes of movements a very accurate approximation of the trajectory sets can be achieved with very few (≤ 5) sources. This makes the method very interesting for the classification of movements (Omlor and Giese, 2007b), but also for the synthesis of realistic looking human movement data in computer graphics (Park et al., 2008; Mukovskiy et al., 2008).



(a) Approximation quality as a function of the number of extracted sources for periodic gait data.

(b) Approximation quality as a function of the number of extracted sources for non periodic arm movements.

Figure 3: Comparison of different blind source separation algorithms. PCA doubled refers to principle component analysis using twice as many sources.

5.3 Image Processing

Blind source separation is interesting for a wide range of applications in image processing. This includes watermarking (Bounkong et al., 2004), denoising (Hoyer and Oja, 2000), deblurring (Bar et al., 2005), or the extraction of image features (Lee and Seung, 1999; Draper et al., 2003). In most of these applications only instantenous models and algorithms have been applied.

With anechoic mixtures spatial displacements can be explicitly modeled (Omlor and Giese, 2007a; Be'ery and Yeredor, 2008). To demonstrate this, a first set of test images (Figure 4) was generated by pasting two objects at random positions in images with a resolution of 150×150 pixels. Ideally, algorithm 4.2 should be able to extract the original pictures of the two objects as features. The result of the feature extraction are shown in Figure 4 for both, the Gerchberg-Saxton phase retrieval method 3.3.2 and deconvolution 3.3.1.

Clearly, deconvolution is superior to the phase retrieval. This is partially due to the very slow convergence of the Gerchberg-Saxton algorithm, especially for small fractional powers of the fractional Fourier transform and for inaccurate estimates of the power spectra. In addition, the deconvolution method exploits, for a second time, the specific structure of the mixture model. A quantitative comparison shows that images predicted from the extracted components predict 95% of the variance of the original images for the deconvolution method, but only 72% for the phase retrieval method.

More interesting is the application of this feature extracting method to real images. Our second image data set consisted of four gray-scale images taken with a digital camera and resampled with a resolution of 200 $\times$ 200 pixels (cf. Figure 5). The photographs show two objects (scissors and a cup) that were placed at different positions on a wooden surface. Before the application of the algorithms the images were whitened (Gluckman, 2005) in order to compensate for the correlation statistics of natural images. This procedure removes strong correlations between features on small spatial scales. In this case only the deconvolution method was implemented. The reconstruction explains 85% of the of the pre-whitened training images and recovers the original objects with reasonable accuracy (see Figure 5).

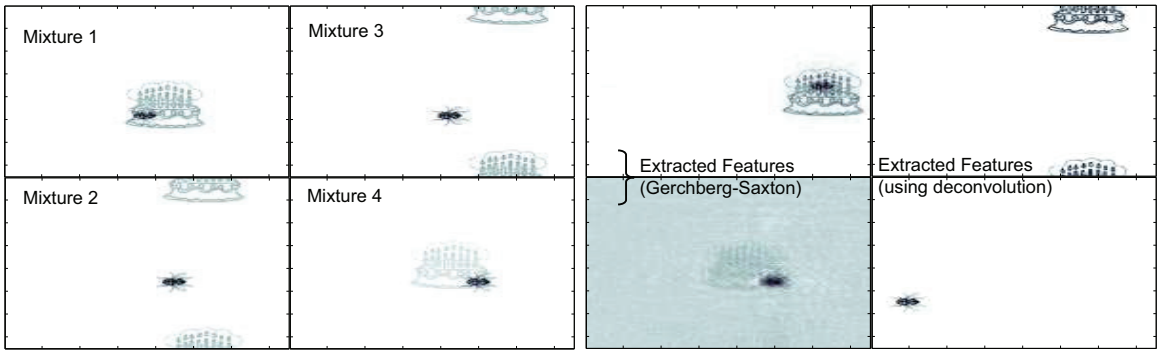


Figure 4: Left: synthetic example images defining an (over-determined) anechoic mixture in two dimensions. Right: extracted features from the image set on the left using GR phase retrieval or deconvolution.



Figure 5: Left: Real Images. Right: Extracted Features from pre-whitened images

5.4 Scale and Rotation Invariant Shape Separation

Automatic identification of image content is an important challenge for many applications such as medical imaging, surveillance or the search in image databases. Often the shape of objects in images has to be recognized independently of object size or orientation. Many supervised and unsupervised learning approaches have been proposed for the classification of images using object shapes (see, e.g., Veltkamp and Hagedoorn, 1999; Mohanty et al., 2005; Ling and Jacobs, 2007; Ahmad and Ibrahim, 2006). Essential for the performance of such algorithms is the underlying shape representation (Zhang and Lu, 2004).

Since an object should be identifiable independent of it is relative size and orientation, scaling and rotation are desired invariances for two-dimensional contour recognition. In order to apply the anechoic mixture model (2) to this problem, a planar contour can be transformed into a log-polar image. Assuming that the center of the coordinate system is given by $[t_1, t_2] = [0, 0]$, the nonlinear transformation is given by:

$$\rho = \log\left(\sqrt{t_1^2 + t_2^2}\right),$$
$$\vartheta = \arctan\left(\frac{t_2}{t_1}\right).$$

Translations in the new coordinates (ρ, ϑ) permit to model scalings and rotations in the original coordinates. A shift of the angle ϑ corresponds to a rotation of the object in the image plane. Shifts of the logarithmic radius ρ can be used to model objects with different sizes. This is just one example for a transformation. In the general case, the independent variable $t \in \mathbb{R}^n$ can be replaced by the nonlinearly transformed coordinates $\phi(t)$. The mixture model (2) then transforms into:

$$\tilde{x}_i(t) = x_i(\phi(t)) = \sum_{j=1}^{n} \alpha_{ij} s_j(\phi(t - \tau_{ij})) = \sum_{j=1}^{n} \alpha_{ij} \tilde{s}_j(t - \tau_{ij}) \quad i = 1, \cdots, m.$$

Thus a shift in the new independent variable t corresponds to a transformation of the source functions $\tilde{s}_j(.) = s_j(\phi(.))$, that depends on the coordinate change ϕ.

5.4.1 APPLICATION TO TWO-DIMENSIONAL SHAPES

To illustrate this application of anechoic mixtures to model rotation and scale invariance, we tested the scale and rotation invariant learning on a small sample of shapes taken from the MPEG-7 test database (Sebastian et al., 2001) (depicted in Figure 6a). After the transformation into log-polar coordinates (with coordinate centers chosen as the xy-mean of the shape), algorithm 4.2 was applied for feature extraction.

Both the extracted features (figure 6b) as well as the corresponding weights α_{ij} (figure 6c) show that in principle, the anechoic mixing model is capable of correctly classifying different objects, independent of their size and rotation. Due to the fact that the one-dimensional contours are treated as 2D images, this approach has a high computational cost, limiting its applicability to large databases. Besides complex objects would need more than one source for an adequate description. This is obvious for the example of the elephant in Figure 6b. An increase in the necessary number of sources would result in decrease of classification performance. There are better approaches for the parametrization of line shapes, such as level sets (Osher and Fedkiw, 2002), which might improve the results of our algorithm in this case.

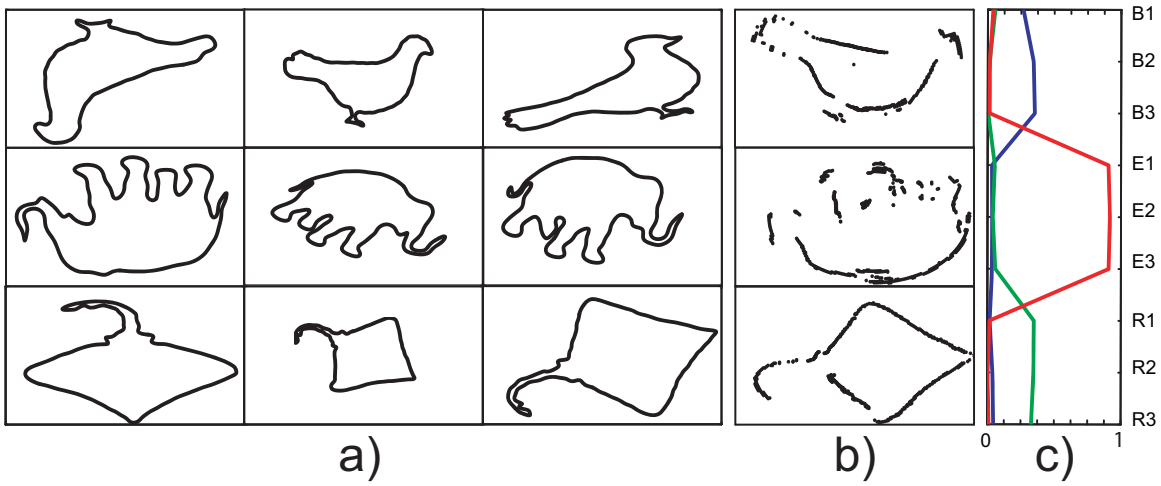


Figure 6: a) Sample object contours taken from the MPEG-7 database. b) Extracted features using algorithm 4.2. c) Amplitude normalized weights α_{ij} corresponding to the three shapes Bird (B1-3), Elephant (E1-3) and Ray (R1-3).

6. Conclusions

We have presented a novel class of algorithms for the solution of over- and under-determined anechoic mixture problems, which extend to an arbitrary number of dimensions of the time argument. The developed method exploits the marginal properties of the stochastic Wigner-Ville distribution. Proper application of this bilinear time-frequency distribution to the delayed mixture model transforms the anechoic problem into a simpler delayed mixture problem in the domain of a linear canonical transform and a phase recovery problem. Appropriate choice of this transformation enhances the separability of the signals and allows the projection of high-dimensional problems onto a system of one-dimensional problems. This results in implementations with a computational complexity that grows linearly in the number of dimensions.

The efficiency of this approach was demonstrated by a series of applications including both synthetic and real world data, such as music streams, natural 2D images, human motion trajectories as well as two dimensional shapes. These examples represent only a small subset of the many possible applications for the learning of invariant features. Due to its modularity, the proposed framework is very flexible and can be easily adjusted and optimized for special applications.

In order to optimize the computational performance of the proposed approach, future work will investigate the use of marginals of time-frequency distributions transcending the set of distributions from the convenient Cohen class. Another important direction is the inclusion of additional constraints in the optimization problem, for example, for the weights of the time delays. Such constraints will allow to model certain topological structures which are given by the data. One particular example for this are the neighborhood relations of certain muscles (d'Avella and Bizzi, 2005), for example, in the context of face movements. In addition, some aspects of the discussed time-frequency framework may be applicable to full convolutive mixtures or mixture models with other invariance properties. Finally, some steps of the proposed algorithms seems to be well-suited

for a fast implementation on Graphics Processing Units (GPUs) (David B. Kirk and Hwu, 2010). An example are the discussed variants of NMF algorithms.

Acknowledgments

Supported by EC FP6 project COBOL, the EC FP7 projects SEARISE, AMARSI and TANGO, the Volkswagenstiftung, DFG Forschergruppe Perceptual Graphics, and the Human Frontiers Science Program (HFSP). Additional support was provided by the Hermann and Lilly-Schilling-Stiftung. We thank C.L. Roether for help with the trajectory acquisition and W. Ilg for support with the motion capturing.

Appendix A. Delay Estimation

Time delay estimation (TDE) problems are common in many technical applications like telecommunications (Takeuchi et al., 1990), radar (Raja Rajeswari and Rani, 1998), sonar (Carter, 1981) or seismology (Du et al., 2004). The corruption of the signal with noise, degradation of the signal shape, moving signal sources, and the multiple overlaps of reflected copies of the signals (multipath), make TDE a very challenging problem. Consequently, a variety of algorithmic approaches have been developed (see Chen et al., 2006 and Bjorklund and Ljung, 2003 for review). If the observed signal $x(t)$ is the superposition of multiple delayed signals s_j:

$$x(t) = \sum_{j=1}^{n} \alpha_j s_j(t - \tau_{ij}) + n(t).$$

then (in case of gaussian noise n) the maximum likelihood (ML) delay estimator takes the form of the least squares problem:

$$\arg \min_{\alpha_j,(D_j)_j} E\left\{ \|x(t) - \sum_{j=1}^{n} \alpha_j s_j(t - D_j))\|^2 \right\}. \tag{36}$$

Though the ML estimator is statistically efficient (i.e., achieves the Cramér Rao bound (CRB)) it requires in general a n-dimensional search and is thus computationally inefficient (Swindelhurst, 1998.). If the signals are uncorrelated $r_{s_i,s_j}(t,t') = 0$ then the ML estimation simplifies to n 1-dimensional estimation problems ($r_{x,s_j}(t)$ specifying the cross-correlation function of the signals x and s_j):

$$\arg \min_{(D_j)_j} E\left\{ \|x(t) - \sum_{j=1}^{n} \alpha_j s_j(t - D_j))\|^2 \right\} = \left(\arg \max_{D_j} r_{x,s_j}(D_j)\right)_j. \tag{37}$$

In this case the corresponding optimal weights α_j can be obtained by linear regression. Though computational very efficient, this approximation is only accurate if cross-talk between the signals is minimal. It is straight-forward, to change both estimators to include the constraint $\alpha_j \geq 0$.

For the solution of the delay estimation problem in the anechoic NMF algorithm (instead of Equation (29)), the ML-cost function (36) was minimized by a nonlinear optimization based on the nonlinear Gauss-Seidel algorithm.

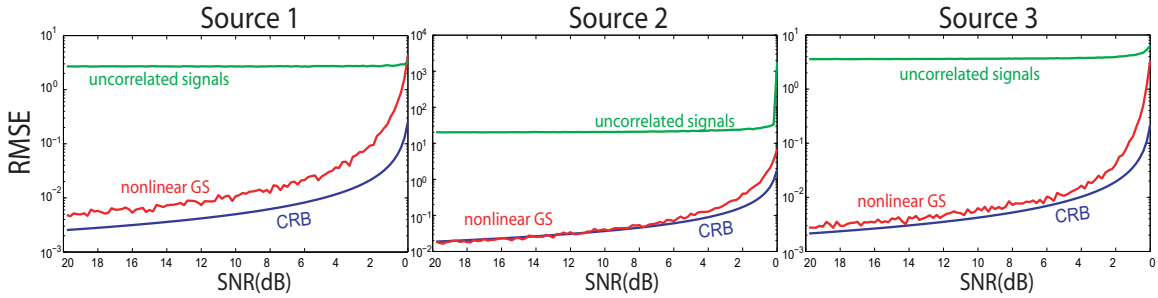


Figure 7: Comparison of different multi-source time-delay estimation methods. The red line shows the RMSE of the nonlinear Gauss-Seidel algorithm, the blue line indicates the Cramér Rao bound (CRB) and the green line shows the performance of the simple estimator (37).

The performance of this estimator was tested in several simulations, and compared to the Cramér Rao bound. The test sources were part of the music data-base used for the sound mixtures described in Section 5.1. The delays were estimated from mixtures with three randomly selected sources and constant weight A and delay matrices $T = (\tau_{ij})_{ij}$:

$$A = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \; T = (\tau_{ij})_{ij} = \begin{pmatrix} 20.2 \\ 7.9 \\ 120.2 \end{pmatrix}.$$

Different amounts of noise were added in order to vary the signal to noise ratio (SNR). The root mean square error $\sqrt{E\{(\hat{\tau}_{ij} - \tau_{ij})^2\}}$, which is compared with the Cramér Rao bound, is based on 1000 simulated trials.

The results for the different estimators can be found in Figure 7. Clearly, the nonlinear Gauss-Seidel iteration outperforms the simple estimator (37). In addition, the Gauss-Seidel iteration results in estimation errors close to the Cramér Rao bound. This implies that the described method for phase retrieval performs close to the theoretically possible optimum.

References

I. Ahmad and M. T. Ibrahim. Image classification and retrieval using correlation. In *CRV '06: Proceedings of the The 3rd Canadian Conference on Computer and Robot Vision*, page 60, Washington, DC, USA, 2006. IEEE Computer Society.

F. Aires, A. Chedin, and J.-P. Nadal. Independent component analysis of multivariate time series: Application to the tropical sst variability. *Journal of Geophysical Research*, 105:1743717455, 2000.

M.S. Akbar and L.J. Douglas. Optimal kernels for nonstationary spectral estimation. *IEEE Transactions on Signal Processing*, 43(2):478–491, 1995.

T. Alieva and M. J. Bastiaans. Properties of the linear canonical integral transformation. *Journal of the Optical Society America A*, 24(11):3658–3665, 2007.

J. B. Anthony and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

S. Arberet, R. Gribonval, and F. Bimbot. A robust method to count and locate audio sources in a stereophonic linear anechoic mixture. In *IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP 2007.*, volume 3, pages III–745 –III–748, 15-20 2007.

L. Bar, N. Sochen, and N. Kiryati. Image deblurring in the presence of salt-and-pepper noise. *Scale Space and PDE Methods in Computer Vision*, pages 107–118, 2005.

R.G. Baraniuk. Wigner-ville spectrum estimation via wavelet soft-thresholding. In *Time-Frequency and Time-Scale Analysis, 1994., Proceedings of the IEEE-SP International Symposium on*, pages 452 –455, October 1994. doi: 10.1109/TFSA.1994.467316.

A. Barliya, L. Omlor, M.A. Giese, and T. Flash. An analytical formulation of the law of intersegmental coordination during human locomotion. *Experimental Brain Research*, 193(3):371–385, 2009.

M. J. Bastiaans and K. B. Wolf. Phase reconstruction from intensity measurements in linear systems. *Journal of the Optical Society America A*, 20(6):1046–1049, 2003.

E. Be'ery and A. Yeredor. Blind separation of superimposed shifted images using parameterized joint diagonalization. *IEEE Transactions on Image Processing*, 17(3):340–353, 2008.

R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ., 1957.

A. Belouchrani and M.G. Amin. Blind source separation based on time-frequency signal representations. *IEEE Transactions on Signal Processing*, 46(11):2888–2897, 1998.

S. Bjorklund and L. Ljung. A review of time-delay estimation techniques. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 3, pages 2502 – 2507, 2003. doi: 10.1109/CDC.2003.1272997.

P. Bofill. Underdetermined blind separation of delayed sound sources in the frequency domain. *Neurocomputing*, 55:627–641, 2003.

S. Bounkong, B. Toch, D. Saad, and D. Lowe. Ica for watermarking digital images. *Journal of Machine Learning Research*, 4(7-8):1471–1498, 2004.

A. Bultheel and H. Martínez-Sulbaran. Recent developments in the theory of the fractional Fourier transforms and linear canonical transforms. *Bulletin of the Belgian Mathematical Society- Simon Stevin*, 13:971–1005, 2006.

J.F. Cardoso. Blind singal separation: statistical principles. *Proceedings of the IEEE*, 9(10):2009–2025, 1998.

G. Carter. Time delay estimation for passive sonar signal processing. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(3):463–470, 1981.

T. Chau. A review of analytical techniques for gait data. part 1: fuzzy, statistical and fractal methods. *Gait & Posture*, 13(1):49–66, 2001.

J. Chen and X.Z. Wang. A new approach to near-infrared spectral data analysis using independent component analysis. *Journal of Chemical Information and Computer Sciences*, 41(4):992–1001, 2001.

J. Chen, J. Benesty, and Y. Huang. Time delay estimation in room acoustic environments: an overview. *EURASIP Journal of Applied Signal Processing*, pages 170–170, 2006.

Z. Chen and A. Cichocki. Nonnegative matrix factorization with temporal smoothness and/or spatial decorrelation constraints. In *Laboratory for Advanced Brain Signal Processing, RIKEN, Tech. Rep*, 2005.

E. C. Cherry. Some experiments on the recognition of speech, with one and with two ears. *Journal of Acoustical Society of America*, 25(5):975–979, 1953.

S. Choi, A. Cichocki, H.H. Park, and S.Y. Lee. Blind signal separation and independent component analysis: A review. *Neural Information Processing – Letters and Reviews*, 5:1–57, 2005.

A. Cichocki and S. Amari. Adaptive blind signal and image processing. *John Wiley, Chichester*, 2002.

A. Cichocki, H. Lee, Y.D. Kim, and S. Choi. Non-negative matrix factorization with alpha-divergence. *Pattern Recognition Letters*, 29(9):1433–1440, 2008.

L. Cohen. Time-frequency distributions-a review. *Proceedings of the IEEE*, 77(7):941–981, 1989.

P. Comon and C. Jutten. *Handbook of Blind Source Separation, Independent Component Analysis and Applications*. Academic Press, 2010.

W. X. Cong, N. X. Chen, and B. Y. Gu. Recursive algorithm for phase retrieval in the fractional fourier transform domain. *Applied Optics*, 37(29):6906–6910, 1998.

A. d'Avella and E. Bizzi. Shared and specific muscle synergies in natural motor behaviors. *Proc Natl Acad Sci U S A*, 102(8):3076–81, 2005.

A. d'Avella, L. Fernandez, A. Portone, and F. Lacquaniti. Modulation of Phasic and Tonic Muscle Synergies With Reaching Direction and Speed. *J Neurophysiol*, 100(3):1433–1454, 2008.

D.B. David B. Kirk and W. W. Hwu. *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufman Publishers, San Fransisco, CA, 2010.

C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix tri-factorizations for clustering. In *Proceedings of the Twelfth ACM SIGKDD international conference on knowledge discovery and data mining*, pages 126–135, 2006.

B.A. Draper, K. Baek, M.S. Bartlett, and J.R. Beveridge. Recognizing faces with PCA and ICA. *Computer Vision and Image Understanding*, 91(1-2):115–137, 2003.

W.X. Du, C.H. Thurber, and D. Eberhart-Phillips. Earthquake relocation using cross-correlation time delay estimates verified with the bispectrum method. *Bulletin of the Seismological Society of America*, 94(3):856–866, 2004.

B. Emile and P. Comon. Estimation of time delays between unknown colored signals. *Signal Processing*, 69(1):93–100, 1998.

T. Flash and B. Hochner. Motor primitives in vertebrates and invertebrates. *Current Opinion in Neurobiology*, 15 (6):660–666, 2005.

P. Georgiev, F. Theis, and A. Cichocki. Sparse component analysis and blind source separation of underdetermined mixtures. *IEEE Transactions on Neural Networks*, 16(4):992 –996, 2005.

R. W. Gerchberg and W. O. Saxton. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35:237–246, 1972.

J. Gluckman. Higher order whitening of natural images. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:354–360, 2005.

R. A. Harshman, S. Hong, and M. E. Lundy. Shifted factor analysis–part i: Models and properties. *Journal of Chemometrics*, 17:363–378, 2003.

J.J. Healy, P. O'Grady, and J.T. Sheridan. Simulating paraxial optical systems using the linear canonical transform: properties, issues and applications. *Optics and Photonics for Information Processing II*, 7072(1):70720E, 2008.

F. Hlawatsch and W. Kozek. The Wigner distribution of a linear signal space. *IEEE Transactions on Signal Processing*, 41(3):1248–1258, 1993.

P. Hoyer and E. Oja. Image denoising by sparse code shrinkage. In *Intelligent Signal Processing*. IEEE Press, 2000.

W. Hu and L.M. Collins. Classification of closely spaced subsurface objects using electromagnetic induction data and blind source separation algorithms. *Radio Science*, 39:1–13, 2004.

W. Ilg, G. H. Bakir, J. Mezger, and M.A. Giese. On the representation, learning and transfer of spatio-temporal movement characteristics. *International Journal of Humanoid Robotics*, 1:613–636, 2004.

A. J. E. M. Janssen. *Application of the Wigner Distribution to Harmonic Analysis of Generalized Stochastic Processes*. Mathematical Centre Tracts 114, Amsterdam, 1979.

O. C. Jenkins and M. J. Mataric. Deriving action and behavior primitives from human motion data. *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, 3:2551–2556, 2002.

S. Karako-Eilon, A. Yeredor, and D. Mendlovic. Blind source separation based on the fractional fourier transform. *Proc. 4th Int. Symp. on Independent Component Analysis*, 46(11):615–620, 2003.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

C.-T. Leung and W.-C. Siu. A general contrast function based blind source separation method for convolutively mixed independent sources. *Signal Processing*, 87(1):107–123, 2007.

Y. Li, A. Cichocki, and S. Amari. Analysis of sparse representation and blind source separation. *Neural Computation*, 16(6):1193–1234, 2004.

Y. Lin and D.D. Lee. Bayesian regularization and nonnegative deconvolution for time delay estimation. In *in Advances in Neural Information Processing Systems 17*, pages 809–816. MIT Press, 2005.

H. Ling and D.W. Jacobs. Shape classification using the inner-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):286–299, 2007.

W. Martin. Time-frequency analysis of random signals. In *Proc. IEEE ICASSP*, pages 1325–1328, Paris, 1982.

W. Martin and P. Flandrin. Wigner-Ville spectral analysis of nonstationary processes. *IEEE Transactions on Acoustics, Speech, and. Signal Processing*, 33(6):1461–1470, 1985.

K. Matsuoka. Minimal distortion principle for blind source separation. In *SICE 2002. Proceedings of the 41st SICE Annual Conference*, volume 4, pages 2138 – 2143, 2002. doi: 10.1109/SICE.2002.1195729.

G. Matz and F. Hlawatsch. Wigner distributions (nearly) everywhere: time-frequency analysis of signals, systems, random processes, signal spaces, and frames. *Signal Processing*, 83(7):1355–1378, 2003.

W. Mecklenbruker and F. Hlawatsch. *The Wigner Distribution*. Elsevier Science, 1997.

N. Mohanty, T. Rath, A. Lee, and R. Manmatha. Learning shapes for image classification and retrieval. In W.-K. Leow, M. Lew, T. S. Chua, W.-Y. Ma, L. Chaisorn, and E. Bakker, editors, *Image and Video Retrieval*, volume 3568 of *Lecture Notes in Computer Science*, pages 589–598. Springer Berlin / Heidelberg, 2005.

M. Mørup, K. H. Madsen, and L. K. Hansen. Shifted independent component analysis. In M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, editors, *ICA2007*, pages 89–96. Springer, Berlin, 2007.

M. Mørup, L.K. Hansen, S.M. Arnfred, L. Lim, and K.H. Madsen. Shift invariant multilinear decomposition of neuroimaging data. *NeuroImage*, 42(4):1439–50, 2008.

A. Mukovskiy, A. Park, L. Omlor, and M.A. Giese. Self-organization of character behavior by mixing of learned movement primitives. In *Proceedings of the 13th Fall Workshop on Vision, Modeling, and Visualization (VMV)*, Konstanz, Germany, 2008.

C. Namgook and C.-C. Jay Kuo. Underdetermined audio source separation from anechoic mixtures with long time delay. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP '09, pages 1557–1560, Washington, DC, USA, 2009. IEEE Computer Society.

D. Nuzillard and A. Bijaoui. Blind source separation and analysis of multispectral astronomical images. *Astronomy and Astrophysics Suppl. Series*, 147:129–138, 2000.

L. Omlor and M. A. Giese. Learning of translation-invariant independent components: Multivariate anechoic mixtures. In M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, editors, *ICA 2007*, pages 762–769. Springer, Berlin, 2007a.

L. Omlor and M. A. Giese. Blind source separation for over-determined delayed mixtures. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1049–1056. MIT Press, Cambridge, MA, 2007b.

S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2002.

H. M. Ozaktas, Z. Zalevsky, and M. A. Kutay. *The Fractional Fourier Transform with Applications in Optics and Signal Processing*. John Wiley & Sons, 2001.

P. O. Ogrady, B. A. Pearlmutter, and S. T. Rickard. Survey of sparse and non-sparse methods in source separation. *International Journal of Imaging Systems and Technology*, 15:2005, 2005.

A. Papoulis, P.S. Unnikrishna, and S. Unnikrishna. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill Higher Education, 2001.

A. Park, A. Mukovskiy, L. Omlor, and M.A. Giese. Self organized character animation based on learned synergies from full-body motion capture data. In *International Conference on Cognitive Systems (CogSys)*, Karlsruhe, Germany, 2008.

M. S. Pedersen, J. Larsen, U. Kjems, and L. C. Parra. A survey of convolutive blind source separation methods. In *Springer Handbook of Speech Processing*, pages 1065–1094. Springer Verlag, Berlin, 2007.

J.C. Platt and F. Faggin. Networks for the separation of sources that are superimposed and delayed. *Advances in Neural Information Processing Systems*, 4:730–737, 1992.

M.D. Plumbley and E. Oja. A "nonnegative pca" algorithm for independent component analysis. *IEEE Transactions on Neural Networks*, 15(1):66–76, 2004.

L. R. Rabiner and B. Gold. *Theory and Application of Digital Signal P'rocessing*. Englewood Cliffs, NJ : Prentice-Hall, 1975.

K. Raja Rajeswari and D.E. Rani. Time-delay estimation using mle approach for wide-band radar systems. *Fourth International Conference on Signal Processing,Proceedings,ICSP '98.*, 2:1493–1496, 1998.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 5500(290):2268–2269, 2000.

A. Safonova, J. K. Hodgins, and N. S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Transactions on Graphics*, 23(3):514–521, 2004.

T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing shock graphs. In *Proceedings of ICCV*, pages 550 – 571, 2001.

K. Seki, M. Narusawa, and P. Smaragdis. Blind separation of convolved mixtures in the frequency domain - signal processing. *Neurocomputing*, 22(14):21–34, 1998.

K. K. Sharmaa and S. D. Joshi. Signal separation using linear canonical and fractional fourier transforms. *Optics Communications*, 265:454–460, 2006.

P. Smaragdis, B. Raj, and M. V. S. Shashanka. Supervised and semi-supervised separation of sounds from single-channel mixtures. In M. E Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, editors, *ICA*, volume 4666 of *Lecture Notes in Computer Science*, pages 414–421. Springer, 2007.

A. Swindelhurst. Time delay and spatial signature estimation using known asynchronous signals. *IEEE Transactions on Signal Processing*, ASSP-33,no. 6:1461–1470, 1998.

T. Takeuchi, M. Sako, and S. Yoshida. Multipath delay estimation for indoor wireless communication. *Vehicular Technology Conference, 1990 IEEE 40th*, pages 401–406, 1990.

K. Torkkola. Blind separation of convolved sources based on information maximization. In *In IEEE Workshop on Neural Networks for Signal Processing*, pages 423–432, 1996a.

K. Torkkola. Blind separation of delayed sources based on information maximization. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 6:3509–3512, 1996b.

M.C. Tresch, V.C. Cheung, and A. d'Avella. Matrix factorization algorithms for the identification of muscle synergies: evaluation on simulated and experimental data sets. *Journal of Neurophysiology*, 95:2199–2212, 2006.

R. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching. Technical Report UU-CS-1999-27, Utrecht University, the Netherlands, 1999.

J. Ville. Théorie et applications de la notion de signal analytique. *Câbles et Trans.*, 2:61–74, 1948.

R. Vollgraf, M. Stetter, and K. Obermayer. Convolutive decorrelation procedures for blind source separation. In *In Proceedings of the 2. International Workshop on*, pages 515–520, 2000.

N. Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. MIT Press, Cambridge, MA, 1964.

E. Wigner. On the quantum correction for thermodynamic equilibrium. *Physical Review*, 40(5): 749–759, 1932.

J. Xiao and P. Flandrin. Multitaper time-frequency reassignment for nonstationary spectrum estimation and chirp enhancement. *IEEE Transactions on Signal Processing*, 55(6):2851–2860, 2007.

S. Yang and Z. Yi. Nonnegative matrix factorization for independent component analysis. In *International Conference on Communications, Circuits and Systems.*, pages 769–771, 2007.

A. Yeredor. Time-delay estimation in mixtures. *Acoustics, Speech, and Signal Processing*, 5:237–40, 2003.

O. Yilmaz and S. Rickard. Blind separation of speech mixtures via time-frequency masking. *IEEE Transactions on Signal Processing*, 52(7):1830–1847, 2004.

Z. Zalevsky, D. Mendlovic, and R. G. Dorsch. Gerchberg-Saxton algorithm applied in the fractional Fourier or the Fresnel domain. *Optics Letters*, 21:842–844, 1996.

D. S. Zhang and G. Lu. Review of shape. representation and description techniques. *Pattern Recognition*, 37(1):1–19, 2004.